Altova XMLSpy 2023 Enterprise Edition



Benutzer- und Referenzhandbuch

Altova XMLSpy 2023 Enterprise Edition
Benutzer- und Referenzhandbuch

All rights reserved. No parts of this work may be reproduced in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems - without the written permission of the publisher.

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.

While every precaution has been taken in the preparation of this document, the publisher and the author assume no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document or from the use of programs and source code that may accompany it. In no event shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

Published: 2022

© 2016-2022 Altova GmbH

Inhaltsverzeichnis

1 XMLSpy und diese Dokumentation 28

1.1 Neue Funktionen 2023 ... 29
1.1.1 Version 2022 ... 29
1.1.2 Version 2021 ... 30
1.1.3 Version 2020 ... 31
1.1.4 Version 2019 ... 33
1.2 Windows Dateipfade ... 34
1.3 RaptorXML Server ... 35

2 XMLSpy Tutorial 36

2.1 XMLSpy-Benutzeroberfläche ... 37
2.1.1 Die Ansichten ... 38
2.1.2 Die Fenster ... 39
2.1.3 Menüs und Symbolleisten ... 41
2.1.4 Einstellungen für die Textansicht ... 44
2.1.5 Applikationsoptionen ... 46
2.2 XML-Schemas: Grundlagen ... 48
2.2.1 Anlegen einer neuen XML-Schema-Datei ... 48
2.2.2 Definieren von Namespaces ... 50
2.2.3 Definieren eines Content Model ... 51
2.2.4 Hinzufügen von Elementen mit Drag & Drop ... 56
2.2.5 Konfigurieren der Content Model-Ansicht ... 57
2.2.6 Fertigstellen des Schemas ... 59
2.3 XML-Schemas: komplexere Vorgänge ... 63
2.3.1 Arbeiten mit ComplexTypes und SimpleTypes ... 63
2.3.2 Referenzieren von globalen Elementen ... 71
2.3.3 Attribute und Attribut-Enumerationen ... 73
2.4 XML-Schemas: XMLSpy-Funktionen ... 76
2.4.1 Schemanavigation ... 76

2.4.2 Schema-Dokumentation ... 78
2.5 XML Dokumente ... 83
2.5.1 Erstellen eines neuen XML-Dokuments ... 84
2.5.2 Festlegen des Typs eines Elements ... 85
2.5.3 Eingabe von Daten in der Grid-Ansicht ... 88
2.5.4 Eingabe von Daten in der Textansicht ... 89
2.5.5 Validierung des Dokuments ... 93
2.5.6 Hinzufügen von Elementen und Attributen ... 97
2.5.7 Bearbeiten in der Tabellenansicht ... 99
2.5.8 Ändern des Schemas ... 103
2.6 XSLT-Transformationen ... 105
2.6.1 Zuweisen einer XSLT-Datei ... 105
2.6.2 Transformieren der XML-Datei ... 106
2.6.3 Ändern der XSL-Datei ... 107
2.7 Projektverwaltung ... 110
2.7.1 Vorteile von Projekten ... 110
2.7.2 Erstellen eines Projekts ... 110
2.8 Fertig! ... 113

3 Benutzeroberfläche und Umgebung 114

3.1 Die grafische Benutzeroberfläche ... 115
3.1.1 Hauptfenster ... 116
3.1.2 Projektfenster ... 118
3.1.3 Info-Fenster ... 120
3.1.4 Eingabehilfen ... 121
3.1.5 Ausgabefenster "Meldungen" ... 122
3.1.6 Ausgabefenster: "XPath/XQuery" ... 124
3.1.7 Ausgabefenster "XSL-Übersicht" ... 125
3.1.8 Ausgabefenster: HTTP ... 126
3.1.9 Ausgabefenster "In Dateien suchen" ... 127
3.1.10 Ausgabefenster "In Schemas suchen" ... 129
3.1.11 Ausgabefenster "In XBRL suchen" ... 129
3.1.12 Ausgabefenster "Diagramme" ... 130
3.1.13 Ausgabefenster: XULE ... 132

3.1.14 Menüleiste, Symbolleisten und Statusleiste ... 133
3.2 Die Applikationsumgebung ... 135
3.2.1 Einstellungen und Anpassung ... 135
3.2.2 Tutorials, Projekte, Beispiele ... 138
3.2.3 XMLSpy Features, Hilfe und Altova-Produkte ... 138

4 Bearbeitungsansichten 140

4.1 Automatisches Backup von Dateien ... 142
4.2 Textansicht ... 144
4.2.1 Formatieren in der Textansicht ... 145
4.2.2 Anzeige des Dokuments ... 148
4.2.3 Bearbeiten in der Textansicht ... 151
4.2.4 Navigation im Dokument ... 154
4.2.5 Suchen und Ersetzen ... 157
4.2.6 Eingabehilfen in der Textansicht ... 165
4.2.7 Geteilte Ansicht ... 166
4.2.8 Tastaturkürzel für die Textansicht ... 167
4.3 Grid-Ansicht ... 169
4.4 Schema-Ansicht ... 171
4.4.1 XSD-Modus: XSD 1.0 oder 1.1 ... 173
4.4.2 Schema-Übersicht ... 177
4.4.3 Content Model-Ansicht ... 190
4.4.4 Attribute, Assertions und Identity Constraints ... 212
4.4.5 Eingabehilfen in der Schema-Ansicht ... 229
4.4.6 Validierung und Smart Fixes ... 239
4.4.7 Assertion-Meldungen ... 241
4.4.8 Modifizieren des Base Type ... 243
4.4.9 Smart Restrictions ... 244
4.4.10 xml:base, xml:id, xml:lang, xml:space ... 249
4.4.11 Zurück und Vorwärts: Navigieren von einer Position zu einer anderen ... 251
4.5 WSDL-Ansicht ... 253
4.5.1 Hauptfenster ... 254
4.5.2 Eingabehilfe Übersicht ... 258
4.5.3 Eingabehilfe Details ... 265

4.6 XBRL-Ansicht ... 266
4.6.1 Hauptfenster: Register "Elemente" ... 266
4.6.2 Hauptfenster: Die Register "Definitionen", "Presentation", "Calculation", "Formula", "Tabelle" ... 271
4.6.3 Eingabehilfen in der XBRL-Ansicht ... 274
4.6.4 Einstellungen für die XBRL-Ansicht ... 280
4.7 Authentic-Ansicht ... 281
4.8 Browser-Ansicht ... 282
4.9 Archivansicht ... 284
4.10 Gebräuchliche Tastaturkürzel ... 286

5 XML 288

5.1 Erstellen, Öffnen und Speichern von XML-Dokumenten ... 289
5.2 Zuweisen von Schemas und Validierung ... 291
5.3 XML in der Textansicht ... 293
5.4 XML in der Grid-Ansicht ... 296
5.4.1 Dokumentanzeige ... 298
5.4.2 Dokumentstruktur ... 302
5.4.3 Dokumentinhalt ... 305
5.4.4 Drag-and-Drop in der Grid-Ansicht ... 309
5.4.5 Tabellenansicht ... 312
5.4.6 Filter ... 317
5.4.7 Formeln ... 319
5.4.8 Bilder ... 322
5.4.9 Diagramme ... 323
5.4.10 Kontextmenü in der Grid-Ansicht ... 326
5.4.11 Einstellungen der Grid-Ansicht ... 329
5.5 XML in der Authentic-Ansicht ... 335
5.6 Eingabehilfen (Textansicht, Authentic-Ansicht) ... 337
5.7 Validieren eines XML-Dokuments ... 339
5.8 Whitespace ... 341
5.9 Einfügen von XML-Fragmenten ... 343
5.10 Verarbeitung mit XSLT und XQuery ... 345
5.11 PDF-Schriftarten ... 347

5.12 Diagramme ... 350
5.12.1 Erstellen eines Diagramms ... 353
5.12.2 Quell-XPath ... 357
5.12.3 Auswahl für die X-Achse ... 361
5.12.4 Auswahl für die Y-Achse ... 366
5.12.5 Diagrammdaten ... 370
5.12.6 Overlays ... 372
5.12.7 Diagrammeinstellungen: Kurzanleitung ... 373
5.12.8 Diagrammeinstellungen und Diagrammdarstellung ... 376
5.12.9 Exportieren ... 403
5.12.10 Diagrammbeispiel: Einfach ... 404
5.12.11 Diagrammbeispiel: Komplex ... 406
5.12.12 Diagrammbeispiel: Kerzendiagramm ... 413
5.13 XML-Signaturen ... 417
5.13.1 Erstellen von XML-Signaturen ... 419
5.13.2 Überprüfen von XML-Signaturen ... 422
5.13.3 Arbeiten mit Zertifikaten ... 425
5.14 Zusätzliche Funktionen ... 430

6 DTDs und XML Schemas 432

6.1 Schema-Manager ... 433
6.1.1 Ausführen des Schema-Managers ... 435
6.1.2 Statuskategorien ... 437
6.1.3 Anwenden eines Patch oder Installation eines Schemas ... 439
6.1.4 Deinstallieren eines Schemas, Zurücksetzen ... 440
6.1.5 Befehlszeilenschnittstelle (CLI) ... 441
6.2 DTDs ... 448
6.3 XML-Schemas ... 451
6.4 Schemamodule ... 452
6.5 Schemaregeln ... 456
6.5.1 Verwalten von Schemaregeln ... 456
6.5.2 Definieren von Schemaregeln ... 458
6.6 Kataloge in XMLSpy ... 463
6.6.1 Funktionsweise von Katalogen ... 463

6.6.2 Katalogstruktur in XMLSpy ... 464
6.6.3 Anpassen von Katalogen ... 466
6.6.4 Umgebungsvariablen ... 467
6.7 Arbeiten mit SchemaAgent ... 470
6.7.1 Herstellen einer Verbindung mit SchemaAgent Server ... 471
6.7.2 Öffnen von Schemas aus dem Suchpfad ... 474
6.7.3 Verwenden von IIR-Beziehungen ... 475
6.7.4 Anzeigen von Schemas in SchemaAgent ... 479
6.7.5 SchemAgent-Validierung ... 479
6.8 In Schemas suchen ... 482
6.8.1 Suchbegriff ... 484
6.8.2 Komponenten ... 485
6.8.3 Eigenschaften ... 488
6.8.4 Bereich ... 491
6.8.5 Die Befehle "Suchen" und "Ersetzen" ... 492
6.8.6 Ergebnisse und Informationen ... 494
6.8.7 Suchen und Umbenennen von globalen Komponenten ... 495

7 XSLT 498

7.1 XSLT-Dokumente ... 499
7.2 XSLT-Verarbeitung ... 501
7.3 XSL-Übersicht ... 504
7.3.1 Das Fenster "XSL-Übersicht" ... 505
7.3.2 Das Fenster "Info" ... 508
7.4 XSL-Geschwindigkeitsoptimierung ... 512

8 XQuery 514

8.1 Bearbeiten von XQuery-Dokumenten ... 516
8.1.1 XQuery-Dokumente ... 517
8.1.2 XQuery Eingabehilfen ... 518
8.1.3 XQuery Syntaxfärbung ... 519
8.1.4 Intelligentes Editieren von XQuery ... 520
8.2 XQuery-Auswertung ... 523

8.3 XQuery-Validierung ... 524
8.4 XQuery/Update-Ausführung ... 525
8.5 XQuery Update Facility ... 528
8.5.1 Vorschau auf und Anwendung von Updates ... 529
8.5.2 Update-Operationen und Syntax ... 531
8.6 XQuery und XML-Datenbanken ... 535

9 XSLT/XQuery Debugger und Profiler 539

9.1 XSLT und XQuery Debugger ... 540
9.1.1 Funktionsweise und Benutzeroberfläche ... 541
9.1.2 Befehle und Symbolleistenschaltflächen ... 543
9.1.3 Debugeinstellungen ... 546
9.1.4 Starten einer Debugger-Sitzung ... 548
9.1.5 Informationsfenster ... 550
9.1.6 Breakpoints ... 557
9.1.7 Tracepoints ... 560
9.1.8 Debugger-Tastaturkürzel ... 566
9.2 XSLT/XQuery Profiler ... 567
9.2.1 Erstellen von XSLT-Profilen ... 572
9.2.2 Erstellen von XQuery-Profilen ... 576
9.2.3 Profiler-Ergebnisse: Exortieren und Diagramme ... 579

10 XPath/XQuery-Ausdrücke 582

10.1 Das Fenster "XPath/XQuery" ... 583
10.2 Auswerten des Ausdrucks ... 585
10.3 Debuggen des Ausdrucks ... 591
10.4 Ausdruckserstellung ... 598
10.5 XQuery-Ausdrücke für JSON ... 601
10.6 Anmerkungen ... 604

11 Authentic 606

11.1 Tutorial Authentic-Ansicht ... 608
11.1.1 Öffnen eines XML-Dokuments in der Authentic-Ansicht ... 609

11.1.2 Die Benutzeroberfläche der Authentic-Ansicht ... 610
11.1.3 Node-Operationen ... 613
11.1.4 Eingabe von Daten in der Authentic-Ansicht ... 616
11.1.5 Eingabe von Attributwerten ... 618
11.1.6 Hinzufügen von Entities ... 619
11.1.7 Drucken des Dokuments ... 620
11.2 Benutzeroberfläche der Authentic-Ansicht ... 622
11.2.1 Übersicht über die Benutzeroberfläche ... 622
11.2.2 Authentic-Ansicht Symbolleistenschaltflächen ... 624
11.2.3 Authentic-Ansicht Hauptfenster ... 627
11.2.4 Authentic-Ansicht Eingabehilfen ... 630
11.2.5 Authentic-Ansicht Kontextmenüs ... 634
11.3 Bearbeitung in der Authentic-Ansicht ... 637
11.3.1 Grundlegendes zur Bearbeitung ... 637
11.3.2 Tabellen in der Authentic-Ansicht ... 642
11.3.3 Bearbeiten einer DB ... 650
11.3.4 Arbeiten mit Datumsangaben ... 657
11.3.5 Definieren von Entities ... 659
11.3.6 XML-Signaturen ... 661
11.3.7 Grafiken in der Authentic-Ansicht ... 662
11.3.8 Verwendung von Tasten in der Authentic-Ansicht ... 663
11.4 Authentic Skripterstellung ... 664

12 HTML und CSS 667

12.1 HTML ... 668
12.2 CSS ... 670

13 JSON, JSON-Schema 676

13.1 JSON-Daten ... 679
13.2 JSON-Schema ... 682
13.3 JSON Lines und JSON-Kommentare ... 684
13.4 JSON-Textansicht ... 685
13.5 JSON-Grid-Ansicht ... 690

13.5.1 Dokumentanzeige in der Grid-Ansicht ... 693
13.5.2 Bearbeiten der Dokumentstruktur ... 697
13.5.3 Bearbeiten des Dokumentinhalts ... 699
13.5.4 Drag-and-Drop in der Grid-Ansicht ... 702
13.5.5 Tabellenansicht ... 704
13.5.6 Filter ... 709
13.5.7 Formeln ... 710
13.5.8 Bilder ... 715
13.5.9 Diagramme ... 716
13.5.10 Kontextmenü in der Grid-Ansicht ... 718
13.5.11 Einstellungen der Grid-Ansicht ... 721
13.6 JSON-Schema-Ansicht ... 726
13.6.1 JSON-Schema-Version ... 727
13.6.2 Hinzufügen von globalen Definitionen ... 729
13.6.3 Eingabehilfen: Übersicht, Details, Constraints ... 730
13.6.4 Globale und lokale Definitionen ... 733
13.6.5 Design-Ansicht ... 735
13.6.6 Objekte und Eigenschaften ... 737
13.6.7 Nicht definierte Eigenschaften ... 741
13.6.8 Objekte und Abhängigkeiten ... 744
13.6.9 Arrays ... 748
13.6.10 Atomare Typen ... 750
13.6.11 Typ-Selektoren (Beliebig, Mehrere, usw.) ... 752
13.6.12 BSON (Binary JSON) für MongoDB ... 754
13.6.13 Operatoren ... 758
13.6.14 Bedingungen ... 760
13.6.15 Konfigurieren der Design-Ansicht ... 761
13.6.16 Generieren von JSON-Schema-Dokumentation ... 762
13.7 Validieren von JSON-Dokumenten ... 765
13.8 Einfügen von JSON-Fragmenten ... 767
13.9 JSON-Transformationen mit XSLT/XQuery ... 769
13.10 XQuery-Ausdrücke für JSON ... 772
13.11 Generieren eines JSON-Schemas anhand einer JSON-Instanz ... 774
13.12 Generieren einer JSON-Instanz anhand eines JSON-Schemas ... 777
13.13 Konvertierung zwischen JSON und XML ... 778

14 Avro, Avro-Schema 779

14.1 Avro-Schema ... 782
14.2 Avro-Daten im JSON-Format ... 785
14.3 Avro-Ansicht: eine Grid-Ansicht von Avro-Binärdateien ... 786

15 WSDL und SOAP 788

15.1 WSDL Tutorial ... 789
15.1.1 Erstellen eines neuen Dokuments ... 789
15.1.2 Erstellen eines PortType ... 790
15.1.3 Erstellen eines Binding ... 792
15.1.4 Erstellen eines Service und von Ports ... 794
15.1.5 Validieren des WSDL-Dokuments ... 795
15.1.6 Herstellen einer Verbindung zu einem Webservice und Öffnen von Dateien ... 796
15.1.7 Senden eines SOAP Request von der WSDL-Datei ... 797
15.1.8 Erstellen von WSDL-Dokumentation ... 798
15.1.9 Konvertieren in WSDL 2.0 ... 800
15.2 SOAP ... 801
15.2.1 SOAP-Validierung ... 801
15.2.2 SOAP Debugger ... 802

16 HTTP 817

16.1 Senden des Request ... 818
16.2 Importieren eines Requests zu senden ... 823
16.3 Empfangen der Response ... 825

17 XBRL 829

17.1 Taxonomie-Manager ... 831
17.1.1 Ausführen des Taxonomie-Managers ... 834
17.1.2 Statuskategorien ... 836
17.1.3 Anwenden eines Patch oder Installation einer Taxonomie ... 837
17.1.4 Deinstallieren einer Taxonomie, Zurücksetzen ... 838

17.1.5 Befehlszeilenschnittstelle (CLI) ... 839
17.2 Grundlegende Arbeitsweise ... 847
17.2.1 Taxonomien: neue und bestehende ... 847
17.2.2 Taxonomiedateien Übersicht ... 848
17.2.3 Erstellen einer neuen Taxonomie ... 851
17.2.4 Importieren einer Basis-Taxonomie ... 852
17.2.5 Namespaces ... 856
17.2.6 Taxonomiedateien ... 859
17.2.7 Hinzufügen von Elementen zu einer Taxonomie ... 861
17.2.8 Beziehungen und Linkroles ... 865
17.2.9 Erstellen von Beziehungen : Teil 1 ... 866
17.2.10 Erstellen von Beziehungen: Teil 2 ... 869
17.3 Zusätzliche Verfahren ... 873
17.3.1 Bevorzugte Labels ... 873
17.3.2 Typed Domains ... 875
17.3.3 Ermittlung und Eliminierung von Duplikaten ... 876
17.3.4 Inline XBRL ... 876
17.4 XBRL Formula Editor ... 878
17.4.1 Formula Linkbases und Link Roles ... 879
17.4.2 Formula-Komponenten ... 880
17.4.3 Bearbeiten der Eigenschaften von Formula-Komponenten ... 897
17.4.4 Formula-Komponentenbeziehungen ... 899
17.4.5 Formula-Parameter ... 901
17.4.6 Suchen von Formula-Komponenten ... 902
17.5 XBRL-Tabellendefinitionseditor ... 904
17.5.1 Tabellen-Linkbases und Link Roles ... 906
17.5.2 Tabellenstruktur ... 908
17.5.3 Tabellenkomponenten ... 922
17.5.4 Bearbeiten von Komponenteneigenschaften und -inhalt ... 927
17.5.5 Tabellenkomponentenbeziehungen ... 928
17.5.6 Tabellenparameter ... 929
17.5.7 Tabellenlayoutvorschau ... 933
17.5.8 Suchen von Tabellenkomponenten ... 936
17.6 XULE ... 938
17.6.1 XULE-Dokumente ... 939

17.6.2 Das Fenster "XULE" 941
17.6.3 XULE-Ausführung 945
17.7 In XBRL suchen 947
17.7.1 Suchbegriff 947
17.7.2 Befehlsausführung 949
17.7.3 Ergebnisse und Informationen 951
17.8 OIM 953
17.9 Validieren von XBRL-Instanzen und -Taxonomien 954

18 Office Open XML, ZIP, EPUB 955

18.1 Arbeiten mit OOXML-Dateien 957
18.2 OOXML-Beispieldateien 959
18.3 ZIP-Dateien 961
18.4 EPUB-Dateien 963

19 Datenbanken 967

19.1 Herstellen einer Verbindung zu einer Datenquelle 969
19.1.1 Starten des Verbindungsassistenten 970
19.1.2 Übersicht über Datenbanktreiber 972
19.1.3 ADO-Verbindung 975
19.1.4 ADO.NET-Verbindung 982
19.1.5 ODBC-Verbindung 989
19.1.6 JDBC-Verbindung 992
19.1.7 PostgreSQL-Verbindung 997
19.1.8 SQLite-Verbindung 998
19.1.9 Globale Ressourcen 1002
19.1.10 Beispiele für Datenbankverbindungen 1003
19.2 Unterstützte Datenbanken 1059

20 Globale Altova-Ressourcen 1061

20.1 Definieren globaler Ressourcen 1062
20.1.1 Dateien 1064
20.1.2 Ordner 1069

20.1.3 Datenbanken ... 1071
20.2 Verwendung globaler Ressourcen ... 1074
20.2.1 Zuweisen von Dateien und Ordnern ... 1074
20.2.2 Zuweisen von Datenbanken ... 1077
20.2.3 Wechseln der aktiven Konfiguration ... 1078

21 Projekte 1080

21.1 Erstellen eines Projekts ... 1081
21.2 Verwenden von Projekten ... 1086

22 RaptorXML(+XBRL) Server 1088

22.1 Hinzufügen von Servern und Server-Konfigurationen ... 1089
22.2 Validierung mit RaptorXML Server ... 1093
22.3 Validierungsoptionen ... 1094
22.3.1 Allgemeine Optionen ... 1094
22.3.2 XML with DTD ... 1095
22.3.3 DTD ... 1095
22.3.4 XML with W3C Schema ... 1096
22.3.5 W3C Schema ... 1096
22.3.6 Inline XBRL-Instanz ... 1097
22.3.7 XBRL-Instanz ... 1099
22.3.8 XBRL-Taxonomie ... 1101
22.3.9 XBRL-Taxonomiepaket ... 1102
22.3.10 XBRL Versioning Report ... 1102
22.3.11 XSLT ... 1103
22.3.12 XQuery ... 1104
22.3.13 JSON ... 1104
22.3.14 JSON Schema ... 1105
22.3.15 AVRO ... 1105
22.3.16 AVRO JSON ... 1106
22.3.17 AVRO Schema ... 1106
22.3.18 EDGAR ... 1106
22.4 XSLT und XQuery mit RaptorXML Server ... 1109

23 Datei-/Verzeichnisvergleiche 1111

23.1 Dateivergleiche ... 1112
23.2 Verzeichnisvergleiche ... 1113

24 Versionskontrolle 1115

24.1 Einrichten der Versionskontrolle ... 1117
24.2 Unterstützte Versionskontrollsysteme ... 1118
24.3 Lokaler Arbeitsbereichordner ... 1120
24.4 Applikationsprojekt ... 1121
24.5 Zu Versionskontrolle hinzufügen ... 1123
24.6 Arbeiten mit Versionskontrolle ... 1126
24.6.1 Hinzufügen zur, Entfernen aus der Versionskontrolle ... 1126
24.6.2 Auschecken, Einchecken ... 1127
24.6.3 Dateien schreibgeschützt abrufen ... 1130
24.6.4 Kopieren und Freigeben über die Versionskontrolle ... 1131
24.6.5 Wechseln der Versionskontrolle ... 1134
24.7 Versionskontrolle mit Git ... 1136
24.7.1 Aktivieren der Git-Versionskontrolle mit dem Git SCC Plug-in ... 1137
24.7.2 Hinzufügen eines Projekts zur Git-Versionskontrolle ... 1137
24.7.3 Klonen eines Projekts anhand der Git-Versionskontrolle ... 1139

25 XMLSpy in MS Visual Studio 1141

25.1 Installieren des XMLSpy Plugin ... 1142
25.2 Unterschiede zur XMLSpy-Standalone-Version ... 1143
25.3 Die XMLSpy Debugger in Visual Studio ... 1146

26 XMLSpy in Eclipse 1147

26.1 Installation des XMLSpy Plugin für Eclipse ... 1148
26.2 XMLSpy-Perspektive in Eclipse ... 1150
26.3 XMLSpy-Einstiegspunkte in Eclipse ... 1153
26.4 Die XMLSpy Debugger-Perspektiven ... 1155

27 Code Generator 1156

27.1 Was ist neu ... 1158
27.2 Generieren von Code anhand von XML-Schemas oder DTDs ... 1160
27.2.1 Informationen zu Schema Wrapper-Bibliotheken (C++) ... 1163
27.2.2 Informationen zu Schema Wrapper-Bibliotheken (C#) ... 1166
27.2.3 Informationen zu Schema Wrapper-Bibliotheken (Java) ... 1168
27.2.4 Integrieren von Altova-Bibliotheken in vorhandene Projekte ... 1170
27.2.5 Beispiel: Verwendung der Schema Wrapper-Bibliotheken ... 1173
27.2.6 Beispiel: Bestellung ... 1197
27.3 Referenz zu generierten Klassen (C++) ... 1205
27.3.1 altova::DateTime ... 1205
27.3.2 altova::Duration ... 1208
27.3.3 altova::DayTimeDuration ... 1210
27.3.4 altova::YearMonthDuration ... 1211
27.3.5 altova::meta::Attribute ... 1212
27.3.6 altova::meta::ComplexType ... 1212
27.3.7 altova::meta::Element ... 1213
27.3.8 altova::meta::SimpleType ... 1214
27.3.9 [YourSchema]::[CDoc] ... 1215
27.3.10 [YourSchema]::[ElementType] ... 1217
27.3.11 [YourSchema]::MemberAttribute ... 1218
27.3.12 [YourSchema]::MemberElement ... 1219
27.4 Referenz zu generierten Klassen (C#) ... 1221
27.4.1 Altova.Types.DateTime ... 1221
27.4.2 Altova.Types.DateTimeFormat ... 1224
27.4.3 Altova.Types.Duration ... 1226
27.4.4 Altova.Xml.Meta.Attribute ... 1228
27.4.5 Altova.Xml.Meta.ComplexType ... 1229
27.4.6 Altova.Xml.Meta.Element ... 1230
27.4.7 Altova.Xml.Meta.SimpleType ... 1230
27.4.8 [YourSchema].[Doc] ... 1231
27.4.9 [YourSchema].[ElementType] ... 1233
27.4.10 [YourSchemaType].MemberAttribute ... 1234

27.4.11 [YourSchemaType].MemberElement ... 1235
27.5 Referenz zu generierten Klassen (Java) ... 1237
27.5.1 com.altova.types.DateTime ... 1237
27.5.2 com.altova.types.Duration ... 1241
27.5.3 com.altova.xml.meta.Attribute ... 1245
27.5.4 com.altova.xml.meta.ComplexType ... 1246
27.5.5 com.altova.xml.meta.Element ... 1246
27.5.6 com.altova.xml.meta.SimpleType ... 1247
27.5.7 com.[YourSchema].[Doc] ... 1248
27.5.8 com.[YourSchema].[ElementType] ... 1249
27.5.9 com.[YourSchema].[YourSchemaType].MemberAttribute ... 1251
27.5.10 com.[YourSchema].[YourSchemaType].MemberElement ... 1251
27.6 Codegenerierung Tipps ... 1253
27.7 Code Generator-Optionen ... 1254
27.8 SPL-Referenz ... 1256
27.8.1 Grundlegende SPL-Struktur ... 1256
27.8.2 Deklarationen ... 1257
27.8.3 Variablen ... 1259
27.8.4 Vordefinierte Variablen ... 1260
27.8.5 Erstellen von Ausgabedateien ... 1262
27.8.6 Operatoren ... 1263
27.8.7 Bedingungen ... 1264
27.8.8 Collections und foreach ... 1265
27.8.9 Subroutinen ... 1266
27.8.10 Vordefinierte Typen ... 1269

28 Menübefehle 1275

28.1 Menü "Datei" ... 1276
28.1.1 Neu ... 1276
28.1.2 Öffnen ... 1280
28.1.3 Neu laden ... 1285
28.1.4 Kodierung ... 1285
28.1.5 Schließen, Alle schließen, Alle inaktiven schließen ... 1286
28.1.6 Speichern, Speichern unter, Alles speichern ... 1286

28.1.7 Als Mail senden ... 1292
28.1.8 Drucken ... 1293
28.1.9 Druckvorschau, Druckereinrichtung ... 1295
28.1.10 Zuletzt verwendete Dateien, Beenden ... 1296
28.2 Menü "Bearbeiten" ... 1297
28.2.1 Rückgängig, Wiederherstellen ... 1298
28.2.2 Ausschneiden, Kopieren, Einfügen, Löschen ... 1298
28.2.3 Als XML/JSON-Text kopieren ... 1299
28.2.4 Als tabulatorgetrennten Text kopieren ... 1300
28.2.5 Als Bild kopieren ... 1300
28.2.6 XPath kopieren ... 1301
28.2.7 XPointer/JSON-Pointer kopieren ... 1301
28.2.8 Einfügen ... 1301
28.2.9 Als Bild speichern ... 1305
28.2.10 Pretty-Print ... 1306
28.2.11 Whitespaces entfernen ... 1306
28.2.12 Alles markieren ... 1306
28.2.13 Suchen, Weitersuchen ... 1307
28.2.14 Ersetzen ... 1310
28.2.15 In Dateien suchen ... 1312
28.2.16 Lesezeichen-Befehle ... 1313
28.2.17 Ein-/Auskommentieren ... 1314
28.3 Menü "Projekt" ... 1316
28.3.1 Neues Projekt ... 1319
28.3.2 Projekt öffnen ... 1319
28.3.3 Projekt neu laden ... 1319
28.3.4 Projekt schließen ... 1319
28.3.5 Projekt speichern, Projekt speichern unter ... 1320
28.3.6 Versionskontrolle ... 1320
28.3.7 Dateien zu Projekt hinzufügen ... 1335
28.3.8 Globale Ressource zu Projekt hinzufügen ... 1336
28.3.9 URL zu Projekt hinzufügen ... 1336
28.3.10 Aktive Datei zu Projekt hinzufügen ... 1336
28.3.11 Aktive und verwandte Dateien zu Projekt hinzufügen ... 1336
28.3.12 Projektordner zu Projekt hinzufügen ... 1337

28.3.13 Externes Verzeichnis zum Projekt hinzufügen ... 1337
28.3.14 Externes Webverzeichnis zum Projekt hinzufügen ... 1340
28.3.15 Skript-Einstellungen ... 1344
28.3.16 Eigenschaften ... 1344
28.3.17 Zuletzt verwendete Projekte ... 1347
28.4 Menü "XML" ... 1348
28.4.1 Typ ... 1349
28.4.2 Einfügen nach/vor ... 1349
28.4.3 Anhängen, Child hinzufügen ... 1349
28.4.4 In Element verpacken ... 1349
28.4.5 Als reinen Text bearbeiten ... 1350
28.4.6 Nach oben/nach unten/höher stufen/tiefer stufen ... 1350
28.4.7 Als Tabelle anzeigen ... 1350
28.4.8 Aufsteigend/Absteigend sortieren ... 1350
28.4.9 Zeilen/Spalten vertauschen ... 1350
28.4.10 XPath auswerten ... 1351
28.4.11 Wohlgeformtheit prüfen ... 1351
28.4.12 XML validieren ... 1353
28.4.13 XML am Server validieren (Hochleistungsvalidierung) ... 1357
28.4.14 Validieren von WSDL-Dateien ... 1358
28.4.15 Bei Bearbeitung validieren ... 1359
28.4.16 Eingabehilfen aktualisieren ... 1359
28.4.17 Namespace Präfix ... 1360
28.4.18 XML-Signatur erstellen ... 1360
28.4.19 XML-Signatur überprüfen ... 1363
28.5 Menü "JSON" ... 1367
28.5.1 Typ ... 1367
28.5.2 Einfügen nach/vor, Anhängen, Child hinzufügen ... 1368
28.5.3 In Array/Objekt verpacken ... 1368
28.5.4 Verschieben ... 1369
28.5.5 Als Tabelle anzeigen ... 1369
28.5.6 Aufsteigend/Absteigend sortieren ... 1369
28.5.7 Zeilen/Spalten vertauschen ... 1369
28.5.8 Kommentare entfernen, Alle erneut auswerten ... 1370
28.6 Menü "DTD/Schema" ... 1371

28.6.1 DTD zuweisen ... 1371
28.6.2 Schema zuweisen ... 1372
28.6.3 Weitere DTD einfügen ... 1372
28.6.4 Gehe zu DTD ... 1373
28.6.5 Gehe zu Schema ... 1373
28.6.6 Gehe zu Definition ... 1373
28.6.7 DTD/Schema generieren ... 1374
28.6.8 DTD nivellieren ... 1376
28.6.9 DTD in Schema konvertieren ... 1376
28.6.10 Schema nivellieren ... 1378
28.6.11 Schema in DTD konvertieren ... 1379
28.6.12 Im UML konvertieren ... 1379
28.6.13 XML anhand von DB, Excel, EDI in MapForce generieren ... 1381
28.6.14 HTML/PDF/Word-Ausgabe mit StyleVision erstellen ... 1381
28.6.15 XML/JSON-Beispieldatei generieren ... 1381
28.6.16 Programmcode generieren ... 1385
28.6.17 Cache-Speicher leeren ... 1385
28.7 Menü "Schema-Design" ... 1386
28.7.1 Schema-Einstellungen ... 1386
28.7.2 Diagramm speichern ... 1389
28.7.3 Dokumentation generieren ... 1389
28.7.4 Ansicht konfigurieren ... 1395
28.7.5 Vergrößern/Verkleinern ... 1398
28.7.6 Globale Elemente anzeigen ... 1398
28.7.7 Diagramm anzeigen ... 1398
28.7.8 Schemaerweiterungen für Datenbanken ... 1399
28.7.9 Verbindung zu SchemaAgent Server herstellen ... 1403
28.7.10 Verbindung mit SchemaAgent Server trennen ... 1404
28.7.11 In SchemaAgent anzeigen ... 1404
28.7.12 SchemaAgent-Validierung ... 1405
28.7.13 Schemamodul erstellen ... 1405
28.7.14 Schema nivellieren ... 1407
28.8 Menü "XSL/XQuery" ... 1409
28.8.1 XSL-Transformation ... 1411
28.8.2 XSL-Geschwindigkeitsoptimierung ... 1412

28.8.3 XSL-FO-Transformation ... 1412
28.8.4 XSL-Parameter / XQuery-Variablen ... 1414
28.8.5 XQuery/Update-Ausführung ... 1417
28.8.6 Rückwärts-Mapping aktivieren ... 1418
28.8.7 XSLT / XQuery-Profilerstellung aktivieren ... 1420
28.8.8 XSL zuweisen ... 1420
28.8.9 XSL-FO zuweisen ... 1421
28.8.10 XML-Beispieldatei zuweisen ... 1421
28.8.11 Gehe zu XSL ... 1422
28.8.12 Gehe zu Quellanweisung ... 1422
28.8.13 Gehe zu Kontext-Node ... 1422
28.8.14 Debugger starten ... 1422
28.8.15 Debugger anhalten ... 1423
28.8.16 Debugger neu starten ... 1423
28.8.17 Debugger-Sitzung beenden ... 1423
28.8.18 Einsteigen ... 1423
28.8.19 Aussteigen ... 1424
28.8.20 Überspringen ... 1424
28.8.21 Aktuellen Execution Node einblenden ... 1424
28.8.22 Breakpoint einfügen/löschen ... 1424
28.8.23 Tracepoint einfügen/löschen ... 1425
28.8.24 Breakpoint aktivieren/deaktivieren ... 1425
28.8.25 Tracepoint aktivieren/deaktivieren ... 1425
28.8.26 Breakpoints/Tracepoints ... 1426
28.8.27 Debug-Fenster ... 1427
28.8.28 Debugeinstellungen ... 1427
28.9 Menü "Authentic" ... 1428
28.9.1 Neues Dokument ... 1429
28.9.2 Datenbankdaten bearbeiten ... 1429
28.9.3 StyleVision Stylesheet zuweisen ... 1430
28.9.4 StyleVision Stylesheet bearbeiten ... 1431
28.9.5 Neue Zeile mit XML-Daten für Bearbeitung auswählen ... 1431
28.9.6 XML-Signatur ... 1432
28.9.7 XML Entities definieren ... 1434
28.9.8 Markup-Code anzeigen ... 1436

28.9.9 RichEdit ... 1437
28.9.10 Zeile anhängen/einfügen/kopieren/löschen ... 1437
28.9.11 Zeile verschieben, Zeile löschen ... 1438
28.9.12 HTML-, RTF-, PDF- Word 2007+-, Text-Dokument generieren ... 1438
28.9.13 Vertrauenswürdige Pfade ... 1438
28.10 Menü "DB" ... 1440
28.10.1 Datenbank abfragen ... 1440
28.10.2 IBM DB2 ... 1456
28.10.3 SQL Server ... 1462
28.10.4 Oracle XML DB ... 1465
28.11 Menü "Konvertieren" ... 1470
28.11.1 Textdatei importieren ... 1470
28.11.2 Datenbankdaten importieren ... 1473
28.11.3 MS Word Dokument Importieren ... 1478
28.11.4 XML-Schema von DB-Struktur erzeugen ... 1478
28.11.5 DB-Import auf Basis von XML-Schema ... 1483
28.11.6 DB-Struktur von XML-Schema erzeugen ... 1484
28.11.7 Export in Textdateien ... 1487
28.11.8 Export in Datenbank ... 1490
28.11.9 XML-Instanz von/in JSON konvertieren ... 1493
28.11.10 XML-Schema von/in JSON-Schema konvertieren ... 1497
28.11.11 In OIM xBRL-XML konvertieren ... 1499
28.11.12 In OIM xBRL-JSON konvertieren ... 1499
28.11.13 In OIM xBRL-CSV konvertieren ... 1500
28.12 Menü "Ansicht" ... 1501
28.12.1 Text-Ansicht ... 1501
28.12.2 Grid Ansicht ... 1502
28.12.3 Schema-Ansicht ... 1502
28.12.4 WSDL-Design-Ansicht ... 1502
28.12.5 XBRL-Taxonomieansicht ... 1503
28.12.6 Authentic-Ansicht ... 1503
28.12.7 Browser-Ansicht ... 1503
28.12.8 Erweitern ... 1504
28.12.9 Reduzieren ... 1504
28.12.10 Alles erweitern ... 1504

28.12.11 Unselektierte reduzieren ... 1504
28.12.12 Optimale Breite ... 1505
28.12.13 Zeilenumbruch ... 1505
28.12.14 Gehe zu Zeile/Zeichen ... 1505
28.12.15 Gehe zu Datei ... 1506
28.12.16 Einstellungen für Textansicht ... 1506
28.13 Menü "Browser" ... 1508
28.14 Menü "WSDL" ... 1509
28.14.1 WSDL 1.1-Komponenten ... 1509
28.14.2 WSDL 2.0-Komponenten ... 1512
28.14.3 Types, Diagramm speichern ... 1516
28.14.4 Dokumentation generieren ... 1517
28.14.5 WSDL-Dokument neu parsen ... 1521
28.14.6 In WSDL 2.0 konvertieren ... 1521
28.14.7 WSDL-Programmcode mit MapForce generieren ... 1521
28.15 Menü "Soap" ... 1522
28.15.1 Neuen SOAP Request erzeugen ... 1522
28.15.2 Request an Server senden ... 1524
28.15.3 SOAP Request-Einstellungen ... 1525
28.15.4 SOAP Debugger-Sitzung ... 1529
28.15.5 Starten ... 1530
28.15.6 Einzelschritt ... 1530
28.15.7 Beim nächsten Request anhalten ... 1531
28.15.8 Bei nächstem Response anhalten ... 1531
28.15.9 Proxyserver anhalten ... 1531
28.15.10 Soap Debugger-Optionen ... 1531
28.16 Menü "XBRL" ... 1533
28.16.1 Arcroles ... 1533
28.16.2 Linkroles ... 1535
28.16.3 Namespace-Präfixe ... 1537
28.16.4 Target Namespace definieren ... 1538
28.16.5 Parameterwerte ... 1538
28.16.6 Importieren/Referenzieren ... 1539
28.16.7 Komponente nach ID suchen ... 1541
28.16.8 Dokumentation generieren ... 1541

28.16.9 Ansichtseinstellungen ... 1545
28.16.10 Mit MapForce XBRL anhand von DB, Excel, CSV generieren ... 1546
28.16.11 XBRL mit StyleVision anhand von HTML/PDF/Word darstellen ... 1546
28.16.12 Formula (am Server) ausführen ... 1546
28.16.13 Tabelle (am Server) generieren ... 1549
28.16.14 Duplikate (am Server) ermitteln ... 1551
28.16.15 XULE ausführen ... 1552
28.16.16 Inline XBRL transformieren ... 1553
28.16.17 EDGAR am Server validieren ... 1554
28.16.18 Verarbeitungsoptionen ... 1555
28.17 Menü "Extras" ... 1557
28.17.1 Rechtschreibung ... 1558
28.17.2 Rechtschreiboptionen ... 1562
28.17.3 Skript-Editor ... 1566
28.17.4 Makros ... 1567
28.17.5 Vergleiche ... 1567
28.17.6 Benutzerdefinierte Tools ... 1578
28.17.7 Globale Ressoucen ... 1578
28.17.8 Aktive Konfiguration ... 1579
28.17.9 Raptor-Server verwalten ... 1579
28.17.10 Raptor-Server und Konfigurationen ... 1583
28.17.11 XBRL-Taxonomie-Manager ... 1583
28.17.12 XML-Schema-Manager ... 1584
28.17.13 Anpassen ... 1584
28.17.14 Symbolleisten und Fenster wiederherstellen ... 1602
28.17.15 Optionen ... 1602
28.18 Menü "Fenster" ... 1650
28.19 Menü "Hilfe" ... 1652
28.19.1 Inhaltsverzeichnis, Index, Suche ... 1652
28.19.2 Tastaturzuordnungen ... 1653
28.19.3 Aktivierung, Bestellformular, Registrieren, Updates ... 1654
28.19.4 Weitere Befehle ... 1657
28.20 Befehlszeile ... 1659

29 Referenz für Programmierer 1660

29.1 Skript-Editor ... 1662
29.1.1 Erstellen eines Skripting-Projekts ... 1663
29.1.2 Vordefinierte Befehle ... 1677
29.1.3 Aktivieren von Skripts und Makros ... 1687
29.2 IDE Plugins ... 1690
29.2.1 Registrierung von IDE Plug-Ins ... 1690
29.2.2 ActiveX Controls ... 1691
29.2.3 Konfigurations-XML ... 1691
29.2.4 ATL-Beispieldateien ... 1694
29.2.5 IXMLSpyPlugIn ... 1700
29.3 Applikations-API ... 1706
29.3.1 Übersicht ... 1707
29.3.2 Schnittstellen ... 1745
29.3.3 Schnittstellen (nicht mehr verwendet) ... 1981
29.3.4 Enumerationen ... 2020
29.3.5 Applikations-API für Java (nicht mehr verwendet) ... 2034
29.4 ActiveX Integration ... 2073
29.4.1 Voraussetzungen ... 2073
29.4.2 Hinzufügen der ActiveX Controls zur Toolbox ... 2075
29.4.3 Integration auf Applikationsebene ... 2076
29.4.4 Integration auf Dokumentebene ... 2078
29.4.5 Beispiele zur ActiveX-Integration ... 2081
29.4.6 Befehlsreferenz ... 2094
29.4.7 Objektreferenz ... 2113

30 Anhänge 2136

30.1 Informationen zum XSLT- und XQuery-Prozessor ... 2137
30.1.1 XSLT 1.0 ... 2137
30.1.2 XSLT 2.0 ... 2137
30.1.3 XSLT 3.0 ... 2139
30.1.4 XQuery 1.0 ... 2140

30.1.5 XQuery 3.1 2142
30.2 XSLT- und XPath/XQuery-Funktionen 2143
30.2.1 Altova-Erweiterungsfunktionen 2144
30.2.2 Diverse Erweiterungsfunktionen 2236
30.3 Datentypen in DB-generierten XML-Schemas 2256
30.3.1 ADO 2256
30.3.2 MS Access 2257
30.3.3 MS SQL Server 2258
30.3.4 MySQL 2258
30.3.5 ODBC 2259
30.3.6 Oracle 2260
30.3.7 Sybase 2261
30.4 Datentypen in DB, die anhand von XML-Schemas generiert wurden 2262
30.4.1 MS Access 2262
30.4.2 MS SQL Server 2263
30.4.3 MySQL 2265
30.4.4 Oracle 2267
30.5 Technische Daten 2270
30.5.1 OS- und Arbeitsspeicheranforderungen 2270
30.5.2 Altova-Prozessoren 2270
30.5.3 Unicode-Unterstützung 2271
30.5.4 Internet-Verwendung 2271
30.6 Lizenzinformationen 2273
30.6.1 Electronic Software Distribution 2273
30.6.2 Software-Aktivierung und Lizenzüberwachung 2274
30.6.3 Altova Endbenutzer-Lizenzvereinbarung 2275

Index 2276

1 XMLSpy und diese Dokumentation

Die **Altova XMLSpy 2023 Enterprise Edition** ist der umfangreichste XML- und JSON-Editor zum Erstellen, Editieren und Debuggen von Unternehmensapplikationen, die auf JSON, XML, XML-Schema, XSLT XQuery, SOAP, WSDL und Webservices, OOXML- und XBRL-Technologien basieren. XMLSpy läuft auf Windows 7 SP1 mit Plattformupdate, Windows 8, Windows 10, Windows 11 und Windows Server 2008 R2 SP1 mit Plattformupdate oder höher. XMLSpy steht in einer 64-Bit und einer 32-Bit-Version und zu einem Sonderpreis im Rahmen des Altova MissionKit-Pakets zur Verfügung.



Diese Dokumentation ist in die folgenden Hauptabschnitte gegliedert:

- ein Einführungs-Tutorial 36
- Beschreibungen der Funktionalitäten von XMLSpy, geordnet nach Technologie oder XMLSpy-spezifischer Funktionalität
- Beschreibungen von Menübefehlen 1275
- eine Referenz für Programmierer 1660
- Anhänge 2136 mit Informationen über (i) den XSLT- und XQuery-Prozessor von XMLSpy und (ii) die XPath/XQuery-Erweiterungsfunktionen von Altova

Letzte Aktualisierung: 12.10.2022

© 2016-2022 Altova GmbH

1.1 Neue Funktionen 2023

- Sie können für die Applikation ein klassisches, helles oder dunkles Design[1650] auswählen. Wenn ein Design aktiv ist, kann seine Textanzeige für Dokumenttyp und Ansicht im Dialogfeld Optionen[1602] separat formatiert werden.
- Auch bei der Integration von XMLSpy in Eclipse[1147] oder Visual Studio[1141] kann ein XMLSpy-Design auswählt werden.
- Mit Hilfe der neuen Komponente Altova-Schema-Manager[433] können Sie gebräuchliche DTDs und XML-Schemas installieren und für die Verwendung mit XMLSpy und anderen Altova-Produkten integrieren. Das Dialogfeld Schema-Manager kann über den Menübefehl **Extras | XML-Schema-Manager**[1584] aufgerufen werden.
- Sie können nun als Browser für die Browser-Ansicht[1617] Microsoft Edge WebView2 auswählen. Dies ist eine Alternative zu Internet Explorer als Browser für die Anzeige in der Browser-Ansicht[282].
- Bei Verwendung des neueren Microsoft Edge WebView2 Browsers für die Browser-Ansicht[282] stehen die Developer Tools[282] des Browsers zum Debuggen und Testen Ihres HTML-Codes zur Verfügung.
- Bei der Inline XBRL-Validierung steht eine Option zur Überprüfung des Dokuments anhand des European Single Electronic Format (ESEF)-Berichterstattungshandbuchs zur Verfügung. Bei der Option handelt es sich um eine Inline XBRL-Einstellung im Dialogfeld "Optionen"[1641].
- Das HTTP-Ausgabefenster[126] verfügt nun über ein Log-Fenster[817] und enthält Informationen über die Proxy-Einstellungen[818].
- Die Datenbankunterstützung[1059] wurde auf SQLite 3.38.5 ausgeweitet.
- Die Eclipse-Unterstützung[1147] wurde aktualisiert und auf die folgenden Versionen ausgeweitet: 2022-09; 2022-06; 2022-03; 2021-12.

1.1.1 Version 2022

Version 2022r2

- Mit Hilfe einer applikationsweiten Einstellung kann die Pretty-Print-Darstellung von XML- und JSON-Dokumenten[1611] automatisch aktiviert werden, wenn ein solches Dokument in der Textansicht geladen wird.
- Sie können im Dialogfeld "Optionen"[1606] festlegen, wie Dateien standardmäßig behandelt werden sollen, wenn Sie in XMLSpy geöffnet werden. Diese Behandlung wird auf Dateien mit einer Erweiterung, angewendet, für die keine Behandlung definiert wurde.
- Sie können bei der Generierung von XML-Beispieldokumenten anhand eines Schemas die Verschachtelungstiefe definieren, bis zu der nicht obligatorische Elemente generiert werden sollen[1381].
- Bei der Konvertierung eines XML-Schemas in ein JSON-Schema[1497] wird nun die Definition von Typen in einem JSON-Objekt unterstützt. Dies ist eine Alternative zur Definition des Typs durch Referenzierung des Typs.
- Bilder, die als Base64-kodierte Strings in XML[288] und JSON[676] gespeichert werden, können als Bilder gespeichert werden. Diese Funktion steht sowohl in der Textansicht (*siehe für XML*[293], *für JSON*[685]) als auch der Grid-Ansicht (*siehe für XML*[322], *für JSON*[715]) dieser Dokumente zur Verfügung.
- Das Projektfenster[118] hat nun eine Symbolleiste, über die häufig verwendete Projektbefehle schnell aufgerufen werden können.
- Wenn Sie den Mauszeiger über eine Bilddatei in einem Projektordner (des Projektfensters[118]) platzieren, wird eine Vorschau auf das Bild angezeigt.

- Die Eclipse-Unterstützung[1147] wurde aktualisiert und auf die neuesten Versionen ausgeweitet: 2021-12, 2021-09; 2021-06; 2021-03.
- Die Visual Studio-Unterstützung[1141] wurde auf Version 2022 (64-Bit) erweitert.
- Die Unterstützung für die Generierung von C++- und C#-Code[1156] wurde auf 2022 und .NET 6 ausgeweitet.
- Die Datenbankunterstützung[1059] wurde auf: PostGreSQL 14.1, SQLite 3.37.2, MariaDB 10.6.5, MySQL 8.0.28 ausgedehnt.

Version 2022

- Unterstützung für BSON-Schemabearbeitung[754].
- Unterstützung für die Konvertierung von XBRL-Daten in OIM-Formate[953] und Validierung von OIM xBRL-Dokumenten[953] als XBRL-Dokumente.
- In JSON-Schemas ab Version draft-2019-09 kann eine Schemakomponente sowohl eine andere Schema-Definition referenzieren als auch gleichzeitig ihre eigenen lokalen Definitionen enthalten. Dies unterscheidet sich von früheren Definitionen, in denen entweder eine Referenz oder lokale Definitionen erlaubt waren, nicht aber beides. Die JSON-Schema-Ansicht[726] wurde aktualisiert und unterstützt nun solche erweiterten Referenzen[733].
- Verbesserte Whitespace-Behandlung[341]
- Verbesserte Grid-Ansicht[169]: Wenn Sie in einem Dokument einen Bildlauf nach unten durchführen, wird eine Überschriftsleiste angezeigt, die die Vorfahren-Nodes des gerade am obersten Rand der Grid-Ansicht angezeigten Node enthält.
- Es können separate Limits für die Anzahl der anzuzeigenden Validierungsmeldungen[1615] für Fehler, Warnungen und XBRL-Inkonsistenzen definiert werden.
- Mit Hilfe von Beendungsmodusoptionen[1603] können Sie auswählen, wie nicht gespeicherte Änderungen und offene Dokumente bei Beendung von XMLSpy behandelt werden sollen.
- Sie können im XPath/XQuery-Fenster XPath/XQuery-Codefragmente aus einer XQuery-Datei laden bzw. in einer XQuery-Datei speichern[598]. Dadurch können XPath/XQuery-Ausdrücke und -Fragmente flexibel wiederverwendet werden.
- Neue Altova-Erweiterungsfunktionen zum Aufrufen von Schemainformationen[2186]: Unix-Zeit- (oder Epochen-Zeit) Funktionen[2165].
- Die Datenbankunterstützung[1059] wurde auf: PostGreSQL 13, IBM DB2 11.5 und MySQL 8.0.25 ausgedehnt.
- Die Eclipse-Unterstützung[1147] wurde aktualisiert und auf die folgenden Versionen ausgeweitet: 2021-09; 2021-06; 2021-03; 2020-12.

1.1.2 Version 2021

Version 2021r3

- Die Unterstützung für JSON-Schemas[682] wurde auf die Versionen 2020-12 und 2019-09 ausgedehnt.
- Die Grid-Ansicht von XML-Dokumenten[296] und JSON-Dokumenten[690] wurde durch weitere Funktionalitäten ergänzt und verbessert.

Version 2021r2

- Die Grid-Ansicht von XML-Dokumenten[296] wurde um eine Reihe neuer Funktionalitäten ergänzt.

© 2016-2022 Altova GmbH

- Auch die Befehle und Einstellungen im Zusammenhang mit der Grid-Ansicht von XML-Dokumenten[296] wurden zur Verbesserung der Benutzerfreundlichkeit neu strukturiert: siehe Einstellungen der Grid-Ansicht[329] und das Dialogfeld "Optionen"[1611].
- Die Einstellungen der Grid-Ansicht von JSON-Dokumenten[690] wurden zur Verbesserung der Benutzerfreundlichkeit umstrukturiert: siehe Einstellungen der Grid-Ansicht[721] und das Dialogfeld "Optionen"[1611].
- Die Grid-Ansicht von DTD-Dokumenten[448] wurde um neue Funktionalitäten ergänzt.
- Die neue Drag Overlay-Funktion in der XML-[309] und der JSON-Grid-Ansicht[702] bietet wertvolle Informationen, die die Verwendung von Drag-and-Drop in der Grid-Ansicht erleichtern.
- Neue Funktionalität, mit der XML-[343] und JSON-Fragmente[767] aus externen Quellen einfach und schnell hinzugefügt werden können.
- Zusätzlichen Komponenten (wie XML-Kommentaren) kann in der Grid-Ansicht eine komponentenspezifische Formatierung[1626] zugewiesen werden. Die einzelnen Komponententypen werden daraufhin in der Grid-Ansicht mit der jeweils zugewiesenen Formatierung angezeigt. So können Kommentare etwa in der Grid-Anscihit gelb markiert angezeigt werden.
- Eine neue Funktion zum Validieren von Dokumenten während der Bearbeitung[1359] kann ein- und ausgeschaltet werden.
- Die Eclipse-Unterstützung[1147] wurde erweitert und auf die folgenden Versionen ausgedehnt: 2020-12; 2020-09; 2020-06; 2020-03.

Version 2021

- Ein Taxonomie-Manager-[831]Tool, mit Hilfe dessen Sie Taxonomien einfach für die Verwendung mit XMLSpy installieren, aktualisieren und verwalten können.
- Das JSON-Menü[1367] bietet die folgenden Verbesserungen an der JSON-Grid-Ansicht: Auswahl einer Grid-Ansichtskomponente und Einfügen eines Elements vor oder nach dieser Komponente; Verschieben der ausgewählten Grid-Ansichtskomponente nach oben/nach unten oder nach links/nach rechts; Entfernen aller Kommentare aus dem Dokument.
- Neue Funktionalitäten in der JSON-Grid-Ansicht: (i) das Kontextmenü[718] wurde vereinfacht; (ii) das Raster kann verkleinert und vergrößert werden[693]; (iii) mehrere gleichrangige Komponenten können gemeinsam erweitert/reduziert angezeigt werden[693]; (iv) für Zellen, die Strings enthalten, kann die Leserichtung auf "von rechts nach links" gesetzt werden (nützlich bei Sprachen mit dieser Leserichtung, wie Arabisch und Hebräisch); der Befehl "Kopieren"[718] wurde verbessert.
- Bei der JSON-Schema-Validierung[726] wird die Eigenschaft `$id` berücksichtigt.
- Neue Altova-Erweiterungsfunktionen zum Aufrufen von Schemainformationen[2186].
- Die Eclipse-Unterstützung[1147] wurde erweitert und erstreckt sich nun auf 2020.06, 2020.03, 2019.12, 2019.09.

1.1.3 Version 2020

Version 2020 Release 2

- Dateien, die gerade bearbeitet werden, werden automatisch gesichert[142] und können im Fall einer unerwarteten Beendung des Programms wiederhergestellt werden. Sie können einstellen, ob und wie oft automatische Backups[1603] erstellt werden sollen.
- Mit der Funktion Bei Bearbeitung validieren[339] werden Validierungs- und Wohlgeformtheitsfehler während der Eingabe in der Text-[339] und der JSON-Grid-Ansicht[699] gekennzeichnet. Diese Funktion

kann in den Programmoptionen[1603] sowie über eine ymbolleisten-Schaltfläche aktiviert bzw. deaktiviert werden.
- In der Schema-Ansicht[171] können Sie mit Hilfe des Kontextmenübefehls Gehe zu Typdefinition[199] zur Typdefinition einer Komponente (simple oder complex) springen.
- Neu: Mit Hilfe der neuen JSON-Grid-Ansichtsoptionen können Sie: (i) festlegen, ob Filter und Formeln[709] automatisch, auf eine Aufforderung hin oder nie in einer applikationsweiten JSON-Metadatendatei gespeichert werden sollen und (ii) Sie können den Pfad dieser JSON-Metadatendatei definieren. Anhand dieser Metadatendatei können die gespeicherten Filter und Formeln in der Folge in mehreren Instanzen von XMLSpy auf die damit verknüpften JSON-Dateien angewendet werden, wodurch Filter und Formeln übertragbar gemacht werden. (*Anmerkung: Wurde in Version 2021r2 entfernt.)*
- JSON-Grid-Ansicht[718]: Das Kontextmenü bietet nun Befehle, um (i) alle nicht ausgewählten Komponenten zu reduzieren und (ii) alle Kommentare, optional einschließlich Formeln (welche als Kommentare gespeichert sind) aus dem Dokument zu entfernen.
- JSON-Grid-Ansicht[690]: Verbesserte Suchfunktion.
- Einstellung für die Behandlung von Leerzeichen in Inline XBRL[1641].
- Neue Methoden für die Codegenerierung[1158].

Version 2020

- Die JSON-Grid-Ansicht[690] bietet nun weitere Bearbeitungsfunktionen: (i) eine automatische Typerkennung, (ii) zelleninterne Befehle, (iii) XQuery-Filter zur Bearbeitung der Ansicht und (iv) XQuery-Formeln zum Generieren zusätzlicher Ausgabedateien anhand von JSON-Daten.
- Unterstützung für die Validierung und Bearbeitung von JSON Lines und JSON-Kommentaren[684].
- In der JSON Grid-Ansicht können Bilder angezeigt werden[715] und Diagramme erstellt und angezeigt werden[716].
- Das Dialogfeld "Optionen"[1602] (**Extras | Optionen**) enthält Einstellungen für die JSON-Grid-Ansicht und für die Pretty-Print-Darstellung von JSON-Dokumenten in der Textansicht. (*Anmerkung: Wurde in das Dialogfeld "Einstellungen für Grid-Ansicht" und den Pretty-Print-Abschnitt des Dialogfelds "Optionen" verschoben).*
- Es wurde ein XULE-Validator und -Prozessor hinzugefügt. XULE ist eine Abfragesprache für XBRL-Instanzdokumente. Eine Übersicht über die neuen XULE-Funktionen finden Sie im Abschnitt XBRL > XULE[938].
- Im Fenster "XULE"[941] können Sie XBRL-Instanzdokumente interaktiv abfragen.
- Bearbeiungshilfe für XULE-Dokumente[939].
- Im Gegensatz zu früher, als jedes Dokument nur einzeln verarbeitet werden konnte, können nun mehrere Inline XBRL-Dokumente[876] als Dokumentenmappe verarbeitet werden.
- Der Auswertungs- und Erstellungsmodus des XPath/XQuery-Fensters[124] wurde neu überarbeitet.
- Die XPath Debugger[591]-Funktionalität des XPath/XQuery-Fensters[124] wurde um ein Reihe neuer Funktionen erweitert: (i) Überwachungsausdrücke, (ii) übersichtlichere Benutzeroberfläche und (iii) umfangreichere Analysefunktionen. Aufgrund dieser neuen Funktionalitäten lassen sich XPath/XQuery-Ausdrücke nun noch leichter anhand von XML- und JSON-Dateien testen und debuggen.
- Die Standardbearbeitungsansicht von Dateien mit unterschiedlichem Dateityp wurde zuvor im Dialogfeld "Optionen"[1606] ausgewählt. Jetzt kann die Standardbearbeitungsansicht zusätzlich dazu auch direkt in der Bearbeitungsansicht selbst[140] ausgewählt werden.
- Die Unterstützung für die Integration von XMLSpy in Visual Studio[1141] wurde nun auf Visual Studio 2019 ausgedehnt.
- Die Unterstützung für die Integration von XMLSpy in Eclipse[1147] wurde auf Eclipse 4.11 und 4.12 ausgeweitet.

© 2016-2022 Altova GmbH

1.1.4 Version 2019

Version 2019 Release 3

Im Folgenden finden Sie eine Liste der neuen Funktionen in XMLSpy **Version 2019r3.**

- Die Textansicht verfügt nun über eine geteilte Ansicht[166]. Dadurch stehen Ihnen für ein Dokument, das Sie gerade bearbeiten, zwei Ansichten zur Verfügung, sodass Sie verschiedene Bereiche eines Dokuments Seite an Seite nebeneinander anzeigen können, anstatt durch eine einzige Ansicht scrollen zu müssen, um einen anderen Teil des Dokuments zu sehen.
- Unterstützung für die neueren JSON-Schema-Versionen; draft-06 und draft-07. Nähere Informationen dazu finden Sie im JSON-Abschnitt dieser Dokumentation[676].
- Das Dialogfeld "Optionen"[1602] (**Extras | Optionen**) bietet eine Option, um den Pfad zur Java Virtual Machine zu definieren, wodurch der automatisch ermittelte Pfad automatisch außer Kraft gesetzt wird.
- Unterstützung für Windows Server 2019.
- Die Unterstützung für Eclipse[1147] wurde auf Eclipse 4.9 und 4.10 ausgedehnt.

Version 2019

Im Folgenden finden Sie eine Liste der neuen Funktionen in XMLSpy **Version 2019.**

- Die Datenbankunterstützung wurde erweitert und umfasst nun auch: Firebird 3.0, IBM Informix 12.10, MariaDB 10.3, Microsoft SQL Server 2017, MySQL 8.0, PostgreSQL 10.
- Integration mit Eclipse 4.8.
- Aufgrund der Unterstützung der Generic Preferred Label 1.0 Recommendation können bevorzugte Labels für XBRL-Ressourcen wie z.B. Calculation-Beziehungen definiert werden.[873].
- Die Definitionen von Typed XBRL-Domains[875] werden auf einem separaten Typed Domains-Register der Eingabehilfe "Details" angezeigt.
- Neue XBRL-Menübefehle zum Ausfindigmachen von doppelt vorhandenen Facts in XBRL-Instanzdokumenten[1551].
- Das Dialogfeld "XBRL-Verarbeitungsoptionen"[1555] enthält Optionen zum (i) Ignorieren von doppelt vorhandenen Facts und (ii) zur Generierung von Tabellen.
- Der XBRL-Abschnitt des Dialogfelds "Optionen" (**Extras | Optionen**) verfügt über ein Register mit Validierungsoptionen für XBRL-Instanzdokumente[1639]. Sie können festlegen, ob Dokumente auf (i) die Konformität mit Dimensions 1.0 und Units Registry 1.0, (ii) doppelt vorhandene Facts und (iii) Inkonsistenzen, die sich aus Berechnungen ergeben, überprüft werden sollen.
- Benutzerdefinierte Attribute von XBRL-Elementen werden auch in der Eingabehilfe "Details"[278] angezeigt.

1.2 Windows Dateipfade

Dateipfade unter Windows

Die in dieser Dokumentation angegebenen Dateipfade sind nicht für alle Betriebssysteme gleich. Sie sollten die folgenden Übereinstimmungen beachten:

- Verzeichnis (Eigene) Dateien: Das Verzeichnis (Eigene) Dateien befindet sich standardmäßig an den nachstehend angeführten Orten. Die Beispieldateien befinden sich in einem Unterverzeichnis dieses Verzeichnisses.

Windows 7/8/10/11	`C:\Benutzer\<Benutzername>\Dokumente`

- *Anwendungsverzeichnis:* Das Anwendungsverzeichnis ist jener Ordner, in dem sich Ihre Altova Anwendung befindet. Der Pfad zum Anwendungsverzeichnis ist standardmäßig folgender:

Windows 7/8/10/11	`C:\Programme\Altova\`
32-Bit-Version auf 64-Bit OS	`C:\Programme (x86)\Altova\`

Anmerkung: XMLSpy läuft auch unter Windows Server 2008 R2 SP1 mit Plattformupdate oder höher.

© 2016-2022 Altova GmbH

1.3 RaptorXML Server

Altova RaptorXML(+XBRL) Server (kurz Raptor oder RaptorXML) ist der ultraschnelle XML- und XBRL-Prozessor der dritten Generation und wurde speziell für die neuesten Standards und parallele Rechnerumgebungen optimiert. RaptorXML lässt sich plattformübergreifend einsetzen und verarbeitet XML- und XBRL-Daten dank Nutzung moderner Multi-Core Computer ultraschnell.

RaptorXML ist in zwei Editionen erhältlich:

- RaptorXML Server, der über ein Netzwerk aufgerufen werden und mehrere Dateien gleichzeitig transformieren kann.
- der RaptorXML+XBRL Server Edition, die über ein Netzwerk aufgerufen werden kann und die mehrere Dateien gleichzeitig transformieren kann und die XBRL-Validierung unterstützt.

Die RaptorXML kann über die Befehlszeile ausgeführt werden und hat Schnittstellen für COM, Java, .NET und Python. Sie können auch von der Benutzeroberfläche von XMLSpy aus einen Raptor-Server ausführen 1088.

Altova Website: Server für die XML-Valdierung, XML-Validierung

2 XMLSpy Tutorial

Mit der Ihnen vorliegenden Anleitung erhalten Sie einen umfassenden Einblick in XML. Sie finden darin Anleitungen zu den wichtigsten Aufgaben im Zusammenhang mit XML. Schritt für Schritt werden Sie mit einigen der wichtigsten Funktionen von XMLSpy vertraut.

Das Tutorial besteht aus den folgenden Teilen:

- XMLSpy-Benutzeroberfläche 37. Hier wird die grafische Benutzeroberfläche (GUI) der Applikation erläutert.
- Erstellen eines XML-Schemas 48. Sie lernen, wie man in der intuitiven Schema-Ansicht von XMLSpy ein XML-Schema erstellt, wie man mit Drag-and-Drop komplexe Inhaltsmodelle erstellt und wie man die Schema-Ansicht konfiguriert.
- Erstellen von complex und simpleTypes, globalen Elementreferenzen und Attributenumerationen mit Hilfe von Funktionen der Schema-Ansicht 63.
- Sie lernen, wie man in der Schema-Ansicht in Schemas navigiert 76 und wie man Dokumentation zu Schemas generiert 76.
- Erstellen eines XML-Dokuments 83. Sie lernen, wie man einem XML-Dokument ein Schema zuweist, wie man ein XML-Dokument in der Grid- und der Text-Ansicht bearbeitet und XML-Dokumente mit Hilfe des integrierten Validators von XMLSpy validiert.
- Transformieren einer XML-Datei mittels XSLT-Stylesheets 105. Dabei müssen Sie eine XSLT-Datei zuweisen und die Transformation mit Hilfe des in XMLSpy integrierten XSLT-Prozessors durchführen.
- Arbeiten mit XMLSpy Projekten 110, mit denen Sie Ihre XML-Dokumente ganz einfach organisieren können.

Installation und Konfiguration

In dieser Anleitung gehen wir davon aus, dass Sie XMLSpy erfolgreich auf Ihrem PC installiert haben, dass Sie einen kostenlosen Evaluierungs-Key erhalten haben oder dass Sie sich als User bereits registriert haben. Die Evaluierungsversion von XMLSpy ist voll funktionsfähig. Allerdings ist diese auf dreißig Tage begrenzt. Sie können danach eine reguläre Lizenz über unseren sicheren Webserver oder über einen Vertriebspartner erwerben.

Anleitung: Beispieldateien

Beispieldateien, die in der vorliegenden Anleitung angesprochen werden, finden Sie im Applikationsordner unter:

```
 C:\Dokumente und Einstellungen\<Benutzer>\Eigene
Dokumente\Altova\XMLSpy2023\Examples\Tutorial
```

Im Verzeichnis `Examples` finden Sie viele XML-Dateien, mit denen Sie experimentieren können. Im Verzeichnis `Tutorial` sind alle Dateien abgelegt, die in dieser Anleitung verwendet werden.

Das Verzeichnis `Template` **(Vorlagen)** im Applikationsordner (normalerweise in `C:\Program Files\Altova`) enthält sämtliche XML-Vorlagen, die verwendet werden, wenn Sie die Menüoption **Datei | Neu** aufrufen. Diese Dateien enthalten alle nötigen Daten (Namespace und XML-Deklarationen) die Sie benötigen, um mit dem XML-Dokument arbeiten zu können.

© 2016-2022 Altova GmbH

2.1 XMLSpy-Benutzeroberfläche

In diesem Abschnitt des Tutorials starten Sie XMLSpy und erhalten eine Einführung in die Benutzeroberfläche.

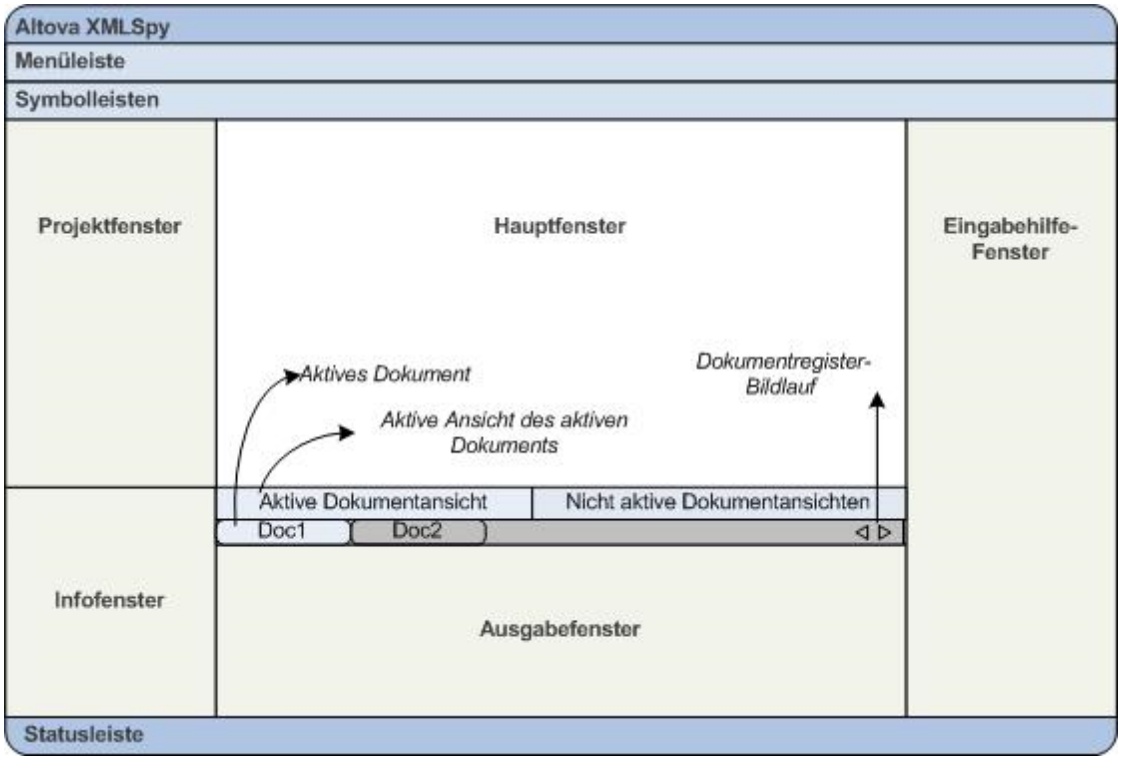
Starten von XMLSpy

Um XMLSpy zu starten, doppelklicken Sie auf das XMLSpy-Symbol auf Ihrem Desktop oder rufen Sie das XMLSpy-Programm über das Menü **Start | Alle Programme** auf. Daraufhin wird XMLSpy gestartet, ohne dass Dokumente auf der Benutzeroberfläche geöffnet werden. Öffnen Sie XMLSpy jetzt.

Übersicht über die Benutzeroberfläche

Standardmäßig ist die Benutzeroberfläche von XMLSpy in drei vertikale Bereiche gegliedert (*Abbildung unten*). Diese drei Bereiche bestehen von links nach rechts gesehen aus: (i) dem Projekt- und dem Info-Fenster; (ii) dem Hauptfenster und den Ausgabefenstern; und (iii) den Eingabehilfefenstern. Im Projektfenster sehen Sie das Beispielprojekt, das beim ersten Start von XMLSpy standardmäßig geöffnet wird.

Im Folgenden werden die wichtigsten Elemente der Benutzeroberfläche und die Funktionen der verschiedenen Komponenten übersichtsmäßig beschrieben. In den Unterabschnitten dieses ersten Teils des Tutorials wird die Benutzeroberfläche näher beschrieben.



Dokumentleiste im Hauptfenster: Wenn mehrere Dokumente geöffnet sind, werden die einzelnen Dokumente im Hauptfenster auf Registern der Dokumentleiste angezeigt (*siehe Abbildung*). Durch Klicken auf ein Register machen Sie das jeweilige Dokument zum aktiven Dokument. Durch Klicken auf die Pfeile auf der rechten Seite der Dokumentleiste können Sie einen Bildlauf durch die Dokumentregister durchführen. Öffnen Sie zwei oder mehr Dateien (z.B. aus dem Projekt "Examples"), um zu sehen, wie die Register verwendet werden.

Dokumentbearbeitungsansichten: Das aktive Dokument kann in einer von mehreren möglichen Bearbeitungsansichten angezeigt werden. Zum Beispiel:

- Ein XML (`.xml`)-Dokument kann in der Text-, Grid-, Authentic- und Browser-Ansicht, nicht aber in anderen Ansichten wie der Schema-Ansicht angezeigt werden.
- Ein XM-Schema (`.xsd`)-Dokument kann dagegen in der Text-, Grid-, Schema- und Browser-Ansicht, nicht aber in der Authentic-Ansicht angezeigt werden.

Es stehen die folgenden Ansichten zur Verfügung: Textansicht[144], Grid-Ansicht[169], Schema-Ansicht[171], WSDL-Ansicht[253], XBRL-Ansicht[266], Authentic-Ansicht[622], Archivansicht[955] und Browser-Ansicht[955].

Eingabehilfen: Die Eingabehilfen ändern sich je nach Art des gerade aktiven Dokuments (z.B. XML oder XSD oder CSS oder WSDL) und je nach der gerade aktiven Dokumentansicht (z.B. Textansicht oder Schema-Ansicht). Über die Eingabehilfen stehen kontextsensitive Bearbeitungsoptionen zur Verfügung, sodass Sie das aktive Dokument schnell und korrekt bearbeiten können.

2.1.1 Die Ansichten

In diesem Teil des Tutorials erfahren Sie, (i) wie man zwischen den Dokumentbearbeitungsansichten wechselt und (ii) wie man die Standardbearbeitungsansicht eines bestimmten Dokumenttyps ändert.

Wechseln zwischen Dokumentansichten

Wenn Sie ein Dokument öffnen, wird es in der für diesen Dokumenttyp definierten Standardansicht geöffnet. So öffnen Sie ein Dokument:

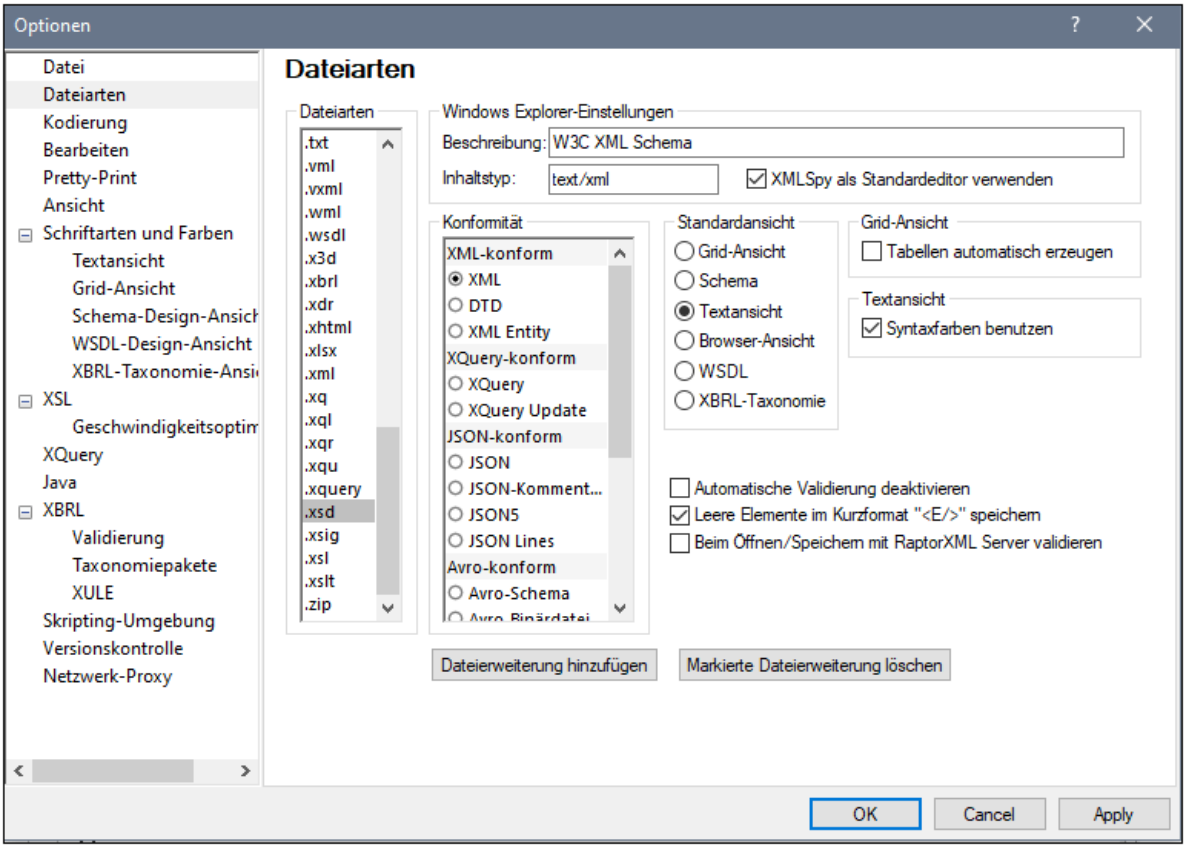
1. Wählen Sie den Befehl **Datei | Öffnen**.
2. Navigieren Sie zur Datei `AddressFirst.xsd` im Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\Tutorial`, wählen Sie sie aus und klicken Sie auf **Öffnen**. Die Datei wird in der Schema-Ansicht geöffnet.
3. Durch Klicken auf die verschiedenen Ansichtsregister (Text, Grid, usw.) am unteren Rand des Hauptfensters können Sie zwischen den verschiedenen Ansichten wechseln. Sie können das XML-Schema-Dokument in der Text-, Grid-, Schema- und Browser-Ansicht anzeigen.
4. Sie können die Ansicht auch durch Auswahl der gewünschten Ansicht im Menü "Ansicht" wechseln. Versuchen Sie, die Ansicht des Dokuments `AddressFirst.xsd` über die Befehle des Menüs "Ansicht" zu wechseln.
5. Schließen Sie das Dokument (mit **Datei | Schließen**).

Ändern der Standardansicht eines Dokumenttyps

Alle Dokumente mit der Erweiterung `.xsd` werden standardmäßig in der Schema-Ansicht geöffnet. Sie können die Standardansicht, in der ein Dokument geöffnet wird, für jeden Dokumenttyp im Dialogfeld "Optionen" ändern. Nehmen Sie diese Einstellung nun für `.xsd`-Dokumente vor.

1. Wählen Sie den Befehl **Extras | Optionen** und wechseln Sie zum Abschnitt *Dateiarten* (*Abbildung unten*) .
2. Scrollen Sie im Bereich *Dateiarten* hinunter zum Eintrag `.xsd` und wählen Sie diesen aus (*siehe Abbildung*).
3. Wählen Sie im Bereich "Standardansicht" die Option "Textansicht" aus.

© 2016-2022 Altova GmbH



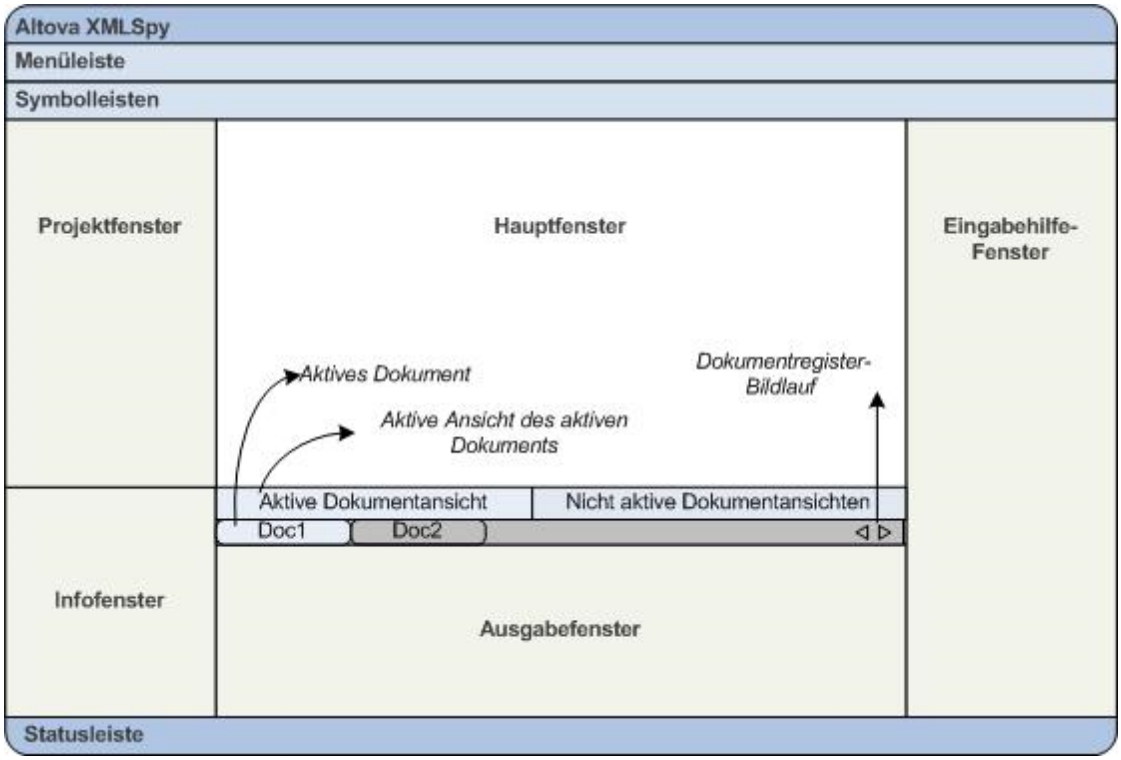
4. Klicken Sie auf **OK**.
5. Wählen Sie den Befehl **Datei | Öffnen** und öffnen Sie die Datei `AddressFirst.xsd`. Die Datei wird in der Textansicht geöffnet.
6. Wechseln Sie zur Schema-Ansicht, um die Datei in dieser Ansicht zu sehen und schließen Sie die Datei anschließend (**Datei | Schließen**).
7. Wechseln Sie zurück in das Dialogfeld "Optionen" (**Extras | Optionen**) und ändern Sie die Standardansicht von `.xsd`-Dateien im Abschnitt *Dateiarten* wieder zurück in Schema.

Anmerkung: Sie können im Dialogfeld "Optionen" im Abschnitt *Dateiarten* (*Abbildung oben*) die Standardansicht jeder der aufgelisteten Dateierweiterungen ändern. Über die Schaltfläche **Neue Dateierweiterung hinzufügen** können Sie eine neue Dateierweiterung zur Liste hinzufügen.

2.1.2 Die Fenster

Standardmäßig sind die Seitenleisten rund um das Hauptfenster angeordnet (*siehe Abbildung unten*) und werden in die folgenden Gruppen eingeteilt:

- Projektfenster
- Infofenster
- Eingabehilfen: (verschiedene. Von der Art der aktiven Dokumentart abhängig)
- Ausgabefenster: Meldungen, Diagramme, XPath, XSL-Übersicht, In Dateien suchen, In Schemas suchen, In XBRL suchen



In diesem Abschnitt erfahren Sie, wie man Fenstergruppen ein- und ausblendet und Fenster am Bildschirm verschiebt. Dies ist nützlich, wenn Sie mehr Platz auf der Benutzeroberfläche benötigen.

Ein- und Ausblenden der Fenstergruppen

Die Fenstergruppen (Projektfenster, Info-Fenster, Eingabehilfen, Ausgabefenster) können über die entsprechenden Befehle im Menü **Fenster** ein- und ausgeblendet werden. Eine Fenstergruppe, die gerade angezeigt wird, kann auch durch Rechtsklick auf die Titelleiste und Auswahl des Befehls **Ausblenden** ausgeblendet werden. Eine ausgeblendete Fenstergruppe kann nur über das Menü **Fenster** wieder eingeblendet werden.

Öffnen Sie eine XML-Datei aus dem Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\Tutorial` und üben Sie die Verwendung dieser einfachen Befehle, um sich damit vertraut zu machen. Nähere Informationen zum Anzeigen und Ausblenden von Fenstergruppen finden Sie im Abschnitt XMLSpy-Benutzeroberfläche 115.

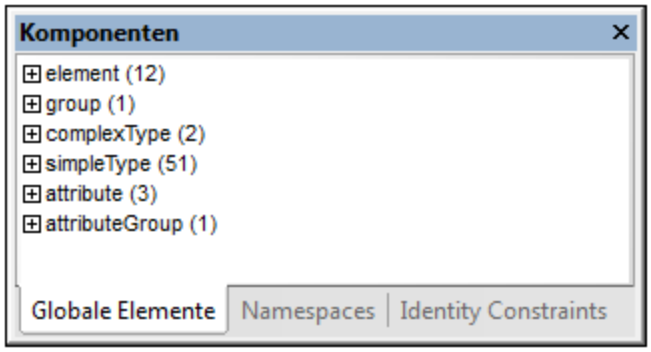
Gespeicherter und gesicherter Status

XMLSpy macht standardmäßig alle fünf Sekunden ein Backup nicht gespeicherter Dokumente. Das Register der jeweiligen Datei am unteren Rand des Hauptfensters gibt mit Hilfe von Symbolen Aufschluss über den gespeicherten/nicht gespeicherten Zustand der Datei und Ihren Backup-Status. Da Ihnen diese Symbolewährend der Arbeit immer wieder unterkommen werden, sollten Sie deren Bedeutung kennen: Informationen zu diesen Symbolen finden Sie im Abschnitt Automatisches Backup von Dateien 142.

 © 2016-2022 Altova GmbH

Verschieben von Fenstern auf dem Bildschirm

Ein einzelnes Fenster kann entweder frei schwebend innerhalb der Benutzeroberfläche oder darin angedockt angezeigt werden. Ein Fenster kann auch als Register in einer anderen Fenstergruppe angedockt werden. (*Erklärung zu Fenstergruppen siehe oben*). So sehen Sie z.B. in der Abbildung unten die Eingabehilfe "Komponenten" in der Schema-Ansicht. Sie hat drei mit Registern versehene Fenster: das Fenster "Gobale Elemente", das Fenster "Namespaces" und das Fenster "Identity Constraints".



Ein Fenster kann auf die folgenden Arten an- oder abgedockt werden:

- Doppelklicken Sie auf die Titelleiste des Fensters. Wenn das Fenster angedockt war, wird es nun frei schwebend angezeigt. Wenn es frei schwebend angezeigt wurde, wird es nun dort angedockt, wo es zuletzt angedockt war.
- Klicken Sie mit der rechten Maustaste auf die Titelleiste eines Fensters und wählen Sie den entsprechenden Befehl aus (**Abgedockt** oder **Angedockt**).
- Ziehen Sie das Fenster (an der Titelleiste als Ziehpunkt) aus seiner angedockten Position bis es frei schwebend angezeigt wird. Ziehen Sie ein frei schwebendes Fenster (an seiner Titelleiste) an die Stelle, an der es angedockt werden soll. Es erscheinen zwei Gruppen blauer Pfeile. Mit den äußeren vier Pfeilen können Sie das Fenster relativ zum Applikationsfenster andocken (am oberen, rechten, unteres oder linken Rand der Benutzeroberfläche). Mit den inneren Pfeilen können Sie das Fenster relativ zu dem Fenster andocken, über dem sich der Cursor gerade befindet. Wenn Sie ein Fenster über den Mittelpunkt der inneren Pfeile ziehen (oder auf die Titelleiste eines Fensters), wird es als Registerkarte in dem Fenster angezeigt, in das es gezogen wurde.

Um ein Fenster, das als Registerkarte angezeigt wird, frei schwebend anzuzeigen, doppelklicken Sie auf sein Register. Um ein als Register angezeigtes Fenster aus einer Gruppe derartiger Fenster herauszuziehen, ziehen Sie es am Register heraus.

Um das Verschieben von Fenstern zu üben, öffnen Sie eine XML-Schema-Datei aus dem Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\Tutorial` und probieren Sie die oben beschriebenen Methoden bei geöffneter Schema-Ansicht aus, um sich damit vertraut zu machen.

2.1.3 Menüs und Symbolleisten

In diesem Abschnitt des Tutorials werden die Hauptfunktionen der Menüs und Symbolleisten von XMLSpy erläutert.

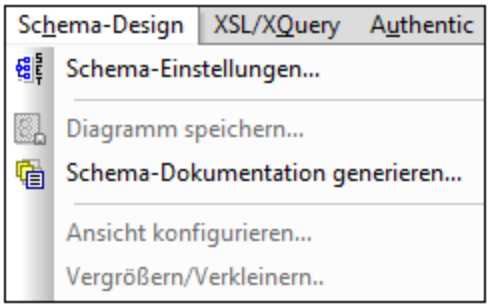
Menüs

Es gibt zwei Hauptleisten: (i) ein Standardmenü, das angezeigt wird, wenn kein Dokument geöffnet ist und (ii) das vollständige XMLSpy Applikationsmenü, das angezeigt wird, sobald ein Dokument geöffnet ist. Gehen Sie folgendermaßen vor:

1. Schließen Sie alle offenen Dokumente mit dem Menübefehl **Datei | Alles schließen**. Daraufhin wird das Standardmenü angezeigt.
2. Öffnen Sie die Datei `AddressFirst.xsd` durch Klicken auf ihren Namen in der Liste der zuletzt geöffneten Dateien ganz unten im Menü **Datei.** Sobald die Datei in der Schema-Ansicht geöffnet ist, ändert sich das Menü und Sie sehen das vollständige XMLSpy-Applikationsmenü.

Die Menüs sind in erster Linie nach Funktion geordnet. Ein Befehl in einem Menü ist nur dann aktiv, wenn er an der Stelle, an der sich der Cursor gerade befindet, ausgeführt werden kann oder, um in der aktuellen Ansicht des aktiven Dokuments eine Auswahl vornehmen zu können. Um zu sehen, unter welchen Umständen ein Menübefehl aktiv oder inaktiv ist, führen Sie die folgenden Schritte durch:

1. Klicken Sie auf das Menü **Schema-Design**. Beachten Sie, dass die Befehle **Diagramm speichern, Ansicht konfigurieren** und **Vergrößern/Verkleinern** deaktiviert sind (*Abbildung unten*).



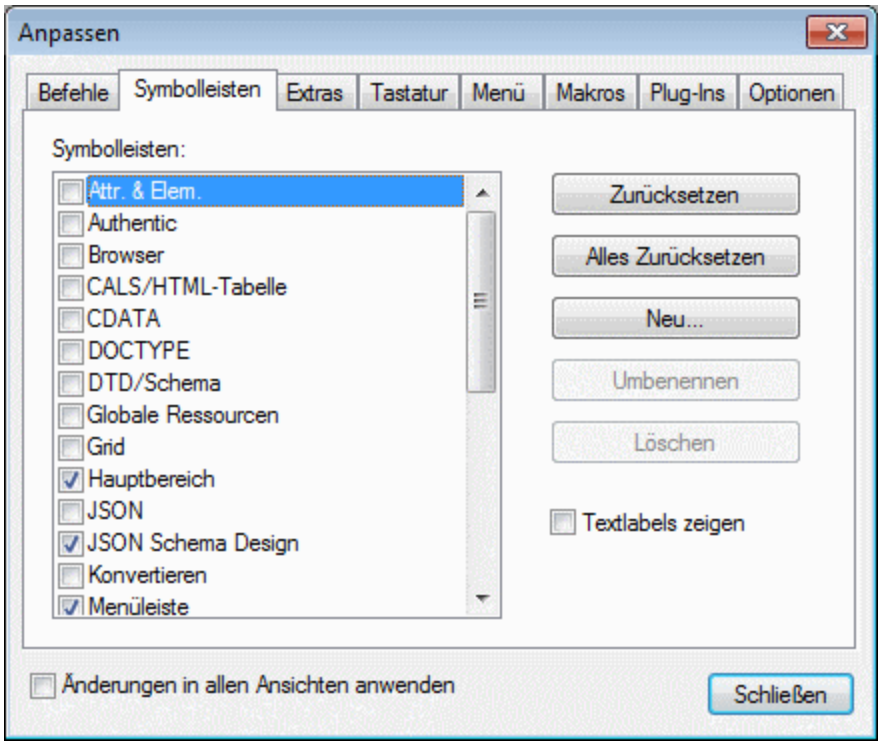
2. Klicken Sie in einen leeren Bereich außerhalb des Menüs, damit das Menü nicht mehr angezeigt wird. Klicken Sie anschließend auf die Schaltfläche **Diagramm anzeigen** links von der Elementkomponente. Daraufhin wechseln Sie in die Content Model-Ansicht der Schema-Ansicht (dies ist die zweite der beiden Schema-Ansichten; die erste ist die Schema-Übersicht). Wenn Sie nun das Menü "Schema-Design" öffnen, sehen Sie, dass die Befehle **Diagramm speichern, Ansicht konfigurieren** und **Vergrößern/Verkleinern** nun aktiv sind. Diese Befehle sind nur in der Content Model-Ansicht der Schema-Ansicht, nicht aber in dedr Schema-Übersicht der Schema-Ansicht aktiv. Beachten Sie außerdem, dass nur XML-Schema-Dateien in der Schema-Ansicht geöffnet werden können.
3. Eine XML-Schema-Datei ist gleichzeitig eine XML-Datei, daher wird sie in der Text- und Grid-Ansicht als XML-Datei angezeigt. In diesen Ansichten sind alle Menübefehle, die sich auf XML-Dateien anwenden lassen, aktiv. Vergleichen Sie die Befehle im Menü **Bearbeiten** in der Schema- und Text-Ansicht miteinander (ob sie aktiv sind oder nicht).
4. Vergleichen Sie als Nächstes die Befehle im Menü **XML | Einfügen** in der Text- und der Grid-Ansicht (ob sie aktiv sind oder nicht). Die Befehle in diesem Menü sind nur in der Grid-Ansicht aktiv.

Eine Beschreibung aller Menübefehle finden Sie im Abschnitt Menübefehle[1275] des Benutzerhandbuchs.

© 2016-2022 Altova GmbH

Symbolleisten

Die Anzeige der Symbolleisten variiert je nachdem, welche Ansicht aktiv ist. In der Standardeinstellung sind die richtigen Symbolleisten für die einzelnen Ansichten definiert. Daher unterscheiden sich die angezeigten Symbolleisten je nach Ansicht. Sie können die Symbolleisten allerdings im Dialogfeld "Anpassen" auf dem Register "Symbolleisten" (**Extras | Anpassen | Symbolleisten**, *Abbildung unten*) auch anpassen.



Üben Sie nun das Verschieben der Symbolleisten auf der Benutzeroberfläche. Klicken Sie auf den Ziehpunkt einer Symbolleiste und ziehen Sie die Symbolleiste an die gewünschte Stelle auf der Benutzeroberfläche. (Der Ziehpunkt wird durch die vertikale gepunktete Linie auf der linken Seite jeder Symbolleiste gekennzeichnet; *siehe Abbildung unten*).

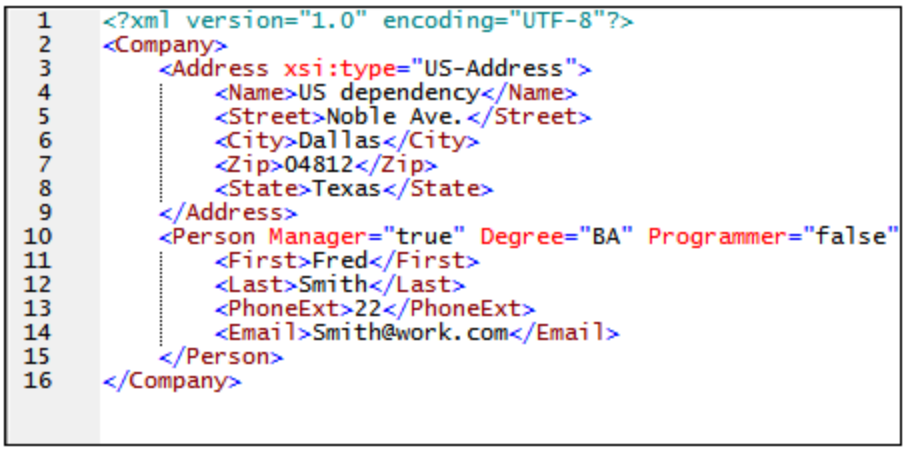


Versuchen Sie, eine Symbolleist an die folgenden Stellen zu ziehen: (i) in eine andere Zeile im Symbolleistenbereich; (ii) nach links oder rechts von einer anderen Symbolleiste; (iii) in die Mitte des Hauptfensters; (iv) angedockt an die linke oder rechte Seite des Applikationsfensters (dazu muss der Ziehpunkt über den linken bzw. rechten Rand des Fensters platziert werden).

Schließen Sie anschließend die Datei `AddressFirst.xsd` wieder.

2.1.4 Einstellungen für die Textansicht

In diesem Abschnitt erfahren Sie, wie Sie die Pretty-Print Anzeige für ein Dokument konfigurieren und Lesezeichen bei der Bearbeitung verwenden. Mit Pretty-Print wird ein Dokument in der Textansicht so angezeigt, dass jede untergeordnete Ebene in der XML-Hierarchie weiter eingerückt ist als die darüberliegende Ebene (*siehe Abbildung unten*). Mit Hilfe von Lesezeichen können Sie schnell zu einer markierten Stelle zurückkehren.

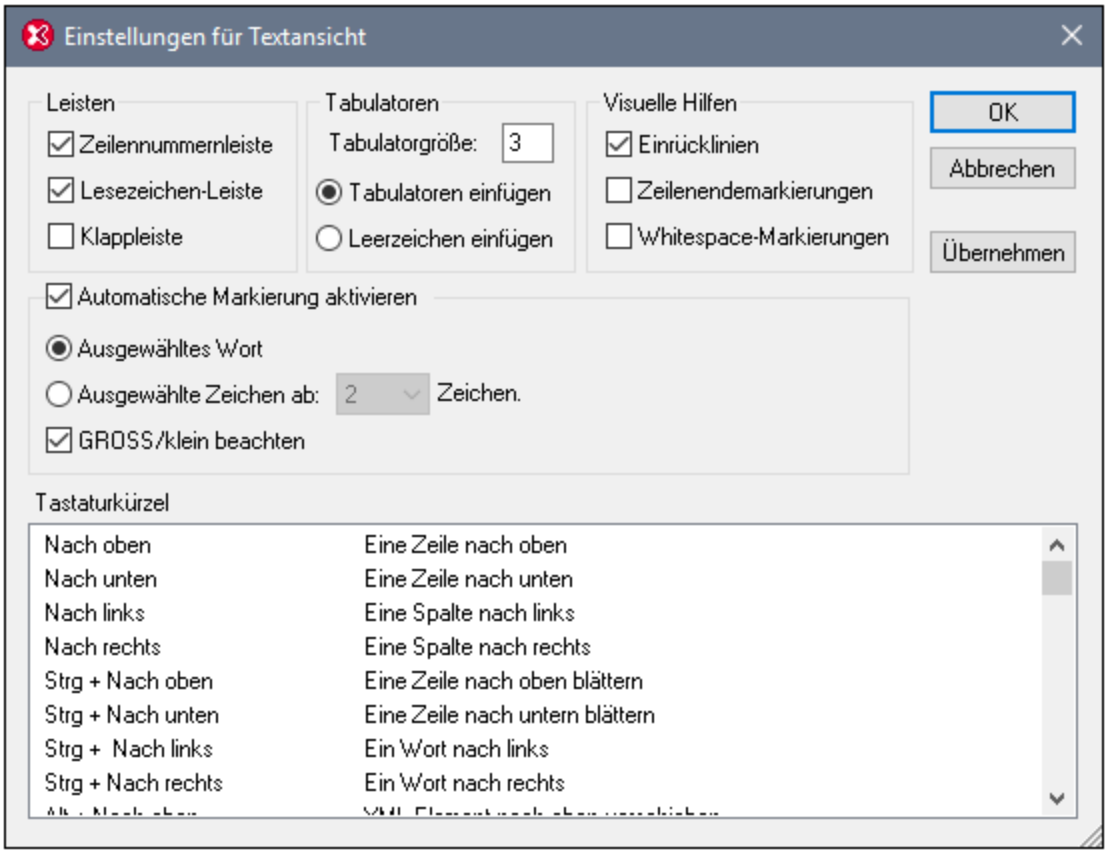


Pretty-Print-Anzeige

Für die Pretty-Print-Anzeige sind zwei Schritte erforderlich: (i) Konfiguration der Pretty-Print-Anzeige, (ii) Anwenden von Pretty-Print.

1. Öffnen Sie die Datei **CompanyFirst.xml**aus dem Ordner **C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\Tutorial**.
2. Wechseln Sie in die Textansicht, wenn diese nicht die Standardansicht von XML-Dokumenten ist.
3. Wählen Sie den Menübefehl **Ansicht | Einstellungen für Textansicht**, um das Dialogfeld "Einstelllungen für Textansicht (*Abbildung unten*) zu öffnen.

© 2016-2022 Altova GmbH



4. Verringern Sie die Tabulatorgröße im Bereich "Tabulatoren" auf 3. Belassen Sie die Standardeinstellung des Optionsfelds *Tabulatoren einfügen* unverändert. Mit dieser Einstellung wird die Pretty-Print-Einrückung[145] als Tabulator (und nicht in Form von Leerzeichen) angezeigt, wobei jeder Tabulator eine Breite von drei Leerzeichen hat. Klicken Sie abschließend auf **OK**.
5. Wählen Sie den Menübefehl **Bearbeiten | Pretty-Print**. Dadurch wird Pretty-Print auf das Dokument angewendet. Daraufhin wird das Dokument mit der neuen Tabulatorbreite angezeigt.
6. Öffnen Sie das Dialogfeld "Einstellungen für Textansicht" (**Ansicht | Einstellungen für Textansicht**) wieder und aktivieren Sie im Bereich *Visuelle Hilfen* die Option "Zeilenendemarkierungen".
7. Gehen Sie in der Textansicht zum Ende einer Zeile und löschen Sie die Zeilenendemarkierung, sodass die nächste Zeile eine Zeile hinaufspringt.
8. Wechseln Sie in die Grid-Ansicht und wieder zurück in die Textansicht.
9. Wählen Sie den Menübefehl **Bearbeiten | Pretty-Print**. Das Dokument wird mit Pretty-Print angezeigt und die gelöschte Zeilenendemarkierung wird wiederhergestellt.

Anmerkung: Wenn Sie im Abschnitt "Pretty-Print" des Dialogfelds "Optionen" (**Extras | Optionen | Pretty-Print**[1611]) das Kontrollkästchen *Einrückungen verwenden* deaktivieren und dann ein Pretty-Print durchführen, beginnen alle Zeilen ohne Einrückung.

Lesezeichen

Lesezeichen werden in eine Lesezeichenleiste links von den zu markierenden Zeilen platziert. Sie können dann schnell von einem Lesezeichen zum nächsten in Ihrem Dokument und wieder zurück springen.

1. Stellen Sie im Dialogfeld "Einstellungen für Textansicht" (**Ansicht | Einstellungen für Textansicht**, *Abbildung oben*) sicher, dass die Option "Lesezeichen-Leiste" im Bereich *Leisten* aktiviert ist. Klicken Sie abschließend auf **OK**.
2. Platzieren sie den Cursor in der Textansicht der Datei `CompanyFirst.xml` an eine beliebige Stelle innerhalb einer zu markierenden Zeile und wählen Sie dann den Menübefehl **Bearbeiten | Lesezeichen einfügen/löschen**. Daraufhin wird die Zeile mit einem blauen Lesezeichen in der Lesezeichenleiste versehen (*siehe Abbildung unten*).
3. Erstellen Sie, wie in Schritt 2 beschrieben, ein weiteres Lesezeichen in eine anderen Zeile.

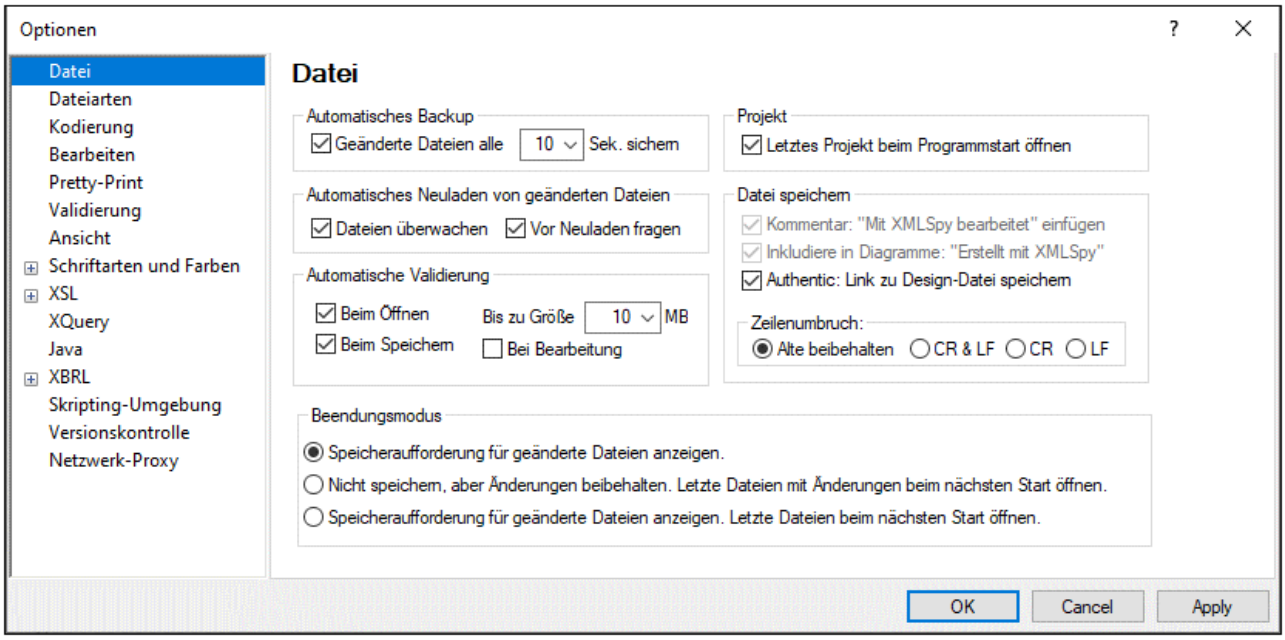
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company>
3     <Address xsi:type="US-Address">
4        <Name>US dependency</Name>
5        <Street>Noble Ave.</Street>
6        <City>Dallas</City>
7        <Zip>04812</Zip>
8        <State>Texas</State>
9     </Address>
10    <Person Manager="true" Degree="BA" Programmer="false">
11       <First>Fred</First>
12       <Last>Smith</Last>
13       <PhoneExt>22</PhoneExt>
14       <Email>Smith@work.com</Email>
15    </Person>
16 </Company>
17
```

4. Drücken Sie **F2** (oder wählen Sie den Befehl **Bearbeiten | Zum nächsten Lesezeichen**), um zum nächsten Lesezeichen im Dokument zu gehen. Drücken Sie **Umschalt + F2** (oder wählen Sie den Befehl **Bearbeiten | Zum nächsten Lesezeichen**), um zum vorherigen Lesezeichen im Dokument zu gelangen. Wiederholen Sie einen oder beide Befehle so oft Sie wollen.
5. Setzen Sie den Cursor in eine der mit einem Lesezeichen versehenen Zeilen und wählen Sie den Menübefehl **Bearbeiten | Lesezeichen einfügen/löschen,** um das Lesezeichen zu löschen. Das Lesezeichen wird entfernt.
6. Speichern und schließen Sie die Datei. In der Datei werden keine Lesezeicheninformationen gespeichert. Öffnen Sie die Datei erneut, um dies zu überprüfen.

2.1.5 Applikationsoptionen

Aufgrund der unzähligen Funktionalitäten von XLMSpy steht eine ganze Reihe von Konfigurationsoptionen zur Verfügung, die sich erheblich auf verschiedene Aspekte Ihrer Arbeit auswirken können. Es ist daher von großem Vorteil, wenn Sie sich mit den Applikationseinstellungen, die Sie je nach Arbeitsanforderung und Arbeitsweise anpassen können, vertraut machen. Diese Einstellungen werden über das Dialogfeld "Optionen" (*Abbildung unten*), das Sie über den Menübefehl **Extras | Optionen** 1602 öffnen können, aufgerufen.

© 2016-2022 Altova GmbH



Die Einstellungen sind in Abschnitte gegliedert, die im Bereich auf der linken Seite aufgelistet werden. Werfen Sie einen Blick in die einzelnen Abschnitte, um einen Überblick über die verfügbaren Einstellungen zu erhalten. In der Liste unten haben wir einige Einstellungen, die sich auf einige häufig verwendete Funktionen beziehen, hervorgehoben. Eine Beschreibung aller Einstellungen finden Sie in der Dokumentation zum Dialogfeld "Optionen" 1602.

Nachdem Sie eine Einstellung geändert haben, klicken Sie auf **OK**, um die Änderung in der Registry zu speichern und das Dialogfeld zu schließen. Bei Klick auf die Schaltfläche **Anwenden** werden die Änderungen in den derzeit geöffneten Dokumenten angezeigt.

Datei
Das *automatische Backup* (von gerade in Bearbeitung befindlichen Dateien) kann aktiviert und deaktiviert werden. *Bei Bearbeitung validieren* führt während der Eingabe eine Wohlgeformtheits- und Validierungsüberprüfung durch. Falls Sie das stört, können Sie die Funktion deaktivieren und die Wohlgeformtheits- und Validierungsüberprüfung jederzeit manuell durchführen.

Dateiarten
Definiert die Standardansicht für Dateiarten. Sie können für jede Dokumentart die für Ihre Bedürfnisse am besten geeignete Ansicht auswählen.

Ansicht
In der XMLSpy-Titelleiste wird entweder nur der Dateiname oder der gesamte Dateipfad angezeigt. Beachten Sie, dass der Dateipfad angezeigt wird, wenn Sie die Maus auf dem Dateiregister am unteren Rand des Hauptfensters über den Namen der Datei platzieren.

Schriftarten und Farben
Sie können die Schriftarten, -größen und die Farbe von Text und anderen Komponenten definieren. Für jede Ansicht gibt es eine eigene Einstellung.

2.2 XML-Schemas: Grundlagen

Ein XML-Schema beschreibt die Struktur eines XML-Dokuments. Ein XML-Dokument kann anhand eines XML-Schemas validiert werden, um zu überprüfen, ob es mit den im Schema definierten Vorgaben übereinstimmt und somit **gültig** ist. Falls dies nicht der Fall ist, ist es **ungültig**. Mit Hilfe von Schemas kann der Verfasser eines Dokuments die Struktur und den zulässigen Inhalt eines XML-Dokuments definieren und überprüfen, ob das XML-Dokument gültig ist.

Ein XML-Schema hat eine komplexe Struktur und Syntax. Da es sich bei einem XML-Schema selbst um ein XML-Dokument handelt, muss es den Regeln der XML-Spezifikation entsprechen. In XMLSpy können Sie in der Schema-Ansicht mit Hilfe von Drag & Drop-Techniken gültige XML-Schemas auf einfache Art erstellen. Das von Ihnen erzeugte XML-Schema-Dokument kann auch in der Text- und der Grid-Ansicht bearbeitet werden, viel einfacher lässt sich dies jedoch in der Schema-Ansicht bewerkstelligen.

Ziel

In diesem Tutorial-Abschnitt lernen Sie, wie man XML-Schemas in der Schema-Ansicht bearbeitet. Dabei lernen Sie die folgenden Schritte:

- Erstellen einer neuen Schema-Datei
- Definieren von Namespaces für das Schema
- Definieren eines einfachen Inhaltsmodells (Content Model)
- Hinzufügen von Elementen zum Inhaltsmodell über Kontextmenüs und mittels Drag-and-Drop
- Konfigurieren der Content Model-Ansicht

Wenn Sie das einfache Schema fertig erstellt haben, können Sie mit dem nächsten Abschnitt des Tutorials 63, in dem Sie erfahren, wie Sie in XMLSpy mit den komplexeren Funktionalitäten von XML-Schema arbeiten, fortfahren. Auf diesen Abschnitt folgt ein Abschnitt über Schemanavigation und Dokumentation 76 in XMLSpy.

In diesem Abschnitt verwendete Befehle

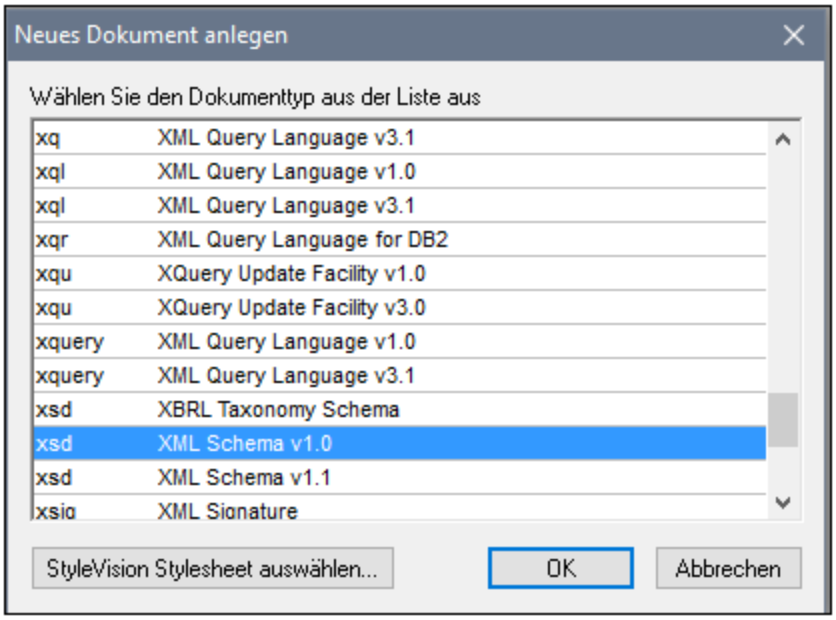
Wir verwenden in diesem Abschnitt des Tutorials ausschließlich die Schema-Ansicht. Dabei werden die folgenden Befehle verwendet:

	Diagramm anzeigen (oder Content Model-Ansicht anzeigen). Dieses Symbol befindet sich links von alle globalen Komponenten in der Schema-Übersicht. Wenn Sie darauf klicken, wird das Content Model der dazugehörigen globalen Komponente angezeigt.

2.2.1 Anlegen einer neuen XML-Schema-Datei

Anlegen einer neuen XML-Schema-Datei

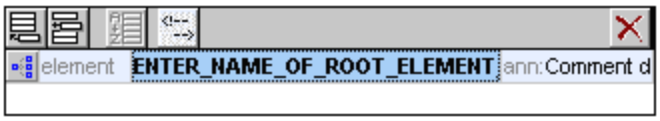
1. Wählen Sie die Menüoption **Datei | Neu**. Das Dialogfeld "Neues Dokument anlegen" wird angezeigt.



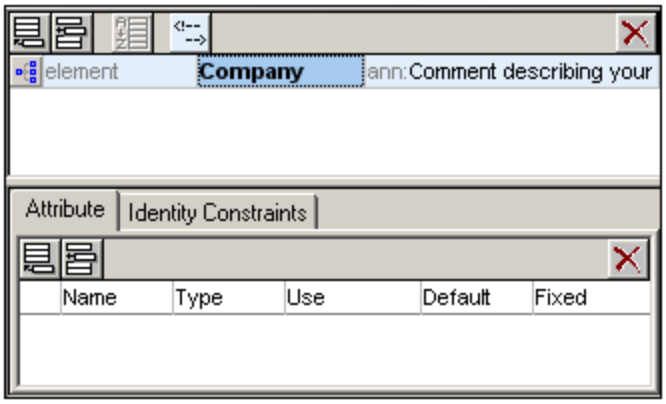
2. Wählen Sie im Dialogfeld den Eintrag *XSD (XML Schema v1.0)* (die Dokumentbeschreibung und die Liste im Fenster sind unter Umständen nicht mit denen in der Abbildung identisch). Bestätigen Sie mit **OK**. Eine leere Schema-Datei wird im Hauptfenster in der Schema-Ansicht angezeigt.
3. Klicken Sie in der Schema-Design-Symbolleiste auf die Schaltfläche **XSD 1.0** (*siehe Abbildung unten*), damit die Schema-Ansicht im XSD 1.0-Bearbeitungsmodus verwendet wird.



4. Sie werden aufgefordert, den Namen des Root-Elements einzugeben.



5. Doppelklicken Sie in das markierte Feld, und geben Sie `Company` ein. Bestätigen Sie mit der **Eingabetaste**. `Company` ist somit das "Root" Element dieses Schemas und gleichzeitig ein globales Element. Die im Hauptfenster angezeigte Ansicht (*siehe Abbildung unten*) nennt sich Schema-Übersicht und enthält eine Liste aller globalen Komponenten im oberen Bereich Fenster; die Attribute und Identity Constraints der ausgewählten globalen Komponente werden im unteren Fenster angezeigt. (Sie können das Content Model einzelner globaler Komponenten ansehen und bearbeiten, indem Sie auf das Symbol "Diagramm anzeigen" klicken, das links von der globalen Komponente zu sehen ist.)



6. Geben Sie im Annotationsfeld (`ann`) des Elements `Company` die Beschreibung für das Element ein, in diesem Fall `Root element`.
7. Klicken Sie auf die Menüoption **Datei | Speichern** und benennen Sie Ihr Schema (z.B. `AddressFirst.xsd`).

Die farbigen Kreissymbole auf dem Register der Datei geben Aufschluss über den Backup-Status der Datei. Informationen zu diesen Symbolen finden Sie im Abschnitt Automatisches Backup von Dateien 142.

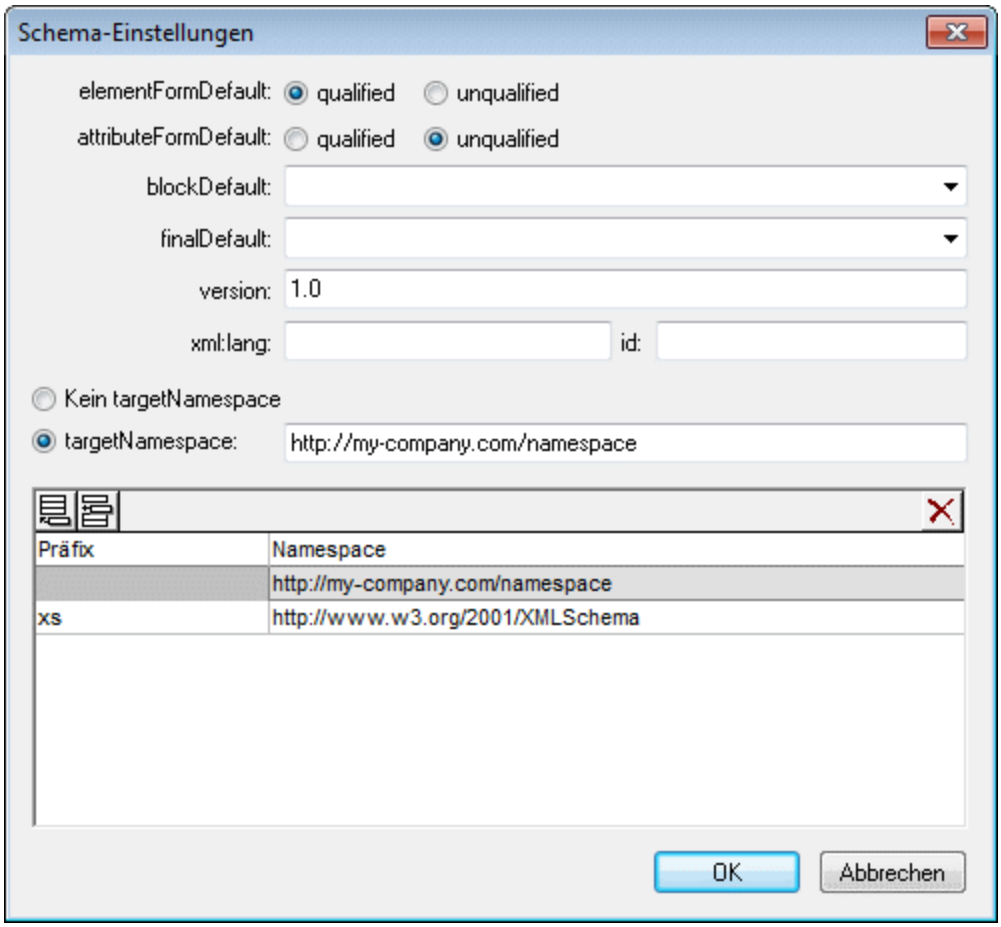
2.2.2 Definieren von Namespaces

XML Namespaces spielen eine wichtige Rolle in XML-Schemas und XML-Dokumenten. Ein XML-Schema muss den XML-Schema-Namespace referenzieren und kann optional auch einen Target Namespace für das XML-Instanzdokument definieren. Als Schema-Autor müssen Sie festlegen, wie Sie diese beiden Namespaces definieren (im Grunde mit Hilfe von Präfixen).

Sie werden nun im XML-Schema, das Sie erstellen, einen Target Namespace für das XML-Instanzdokument definieren. (Die erforderliche Referenz zum XML Schema Namespace wird von XMLSpy automatisch erstellt, wenn Sie ein neues XML Schema-Dokument anlegen.) So erstellen Sie einen Target Namespace:

1. Wählen Sie die Menüoption **Schema-Design | Schema-Einstellungen**, um das Dialogfeld "Schema-Einstellungen" aufzurufen (*Abbildung unten*).

© 2016-2022 Altova GmbH



2. Klicken Sie auf das Target Namespace-Optionsfeld und geben Sie `http://my-company.com/Namespace` ein. In XMLSpy wird der von Ihnen als Target Namespace eingegebene Namespace als Default Namespace des XML-Schema-Dokuments erstellt und in der Liste der Namespaces im unteren Bereich des Dialogfelds angezeigt.
3. Bestätigen Sie mit OK.

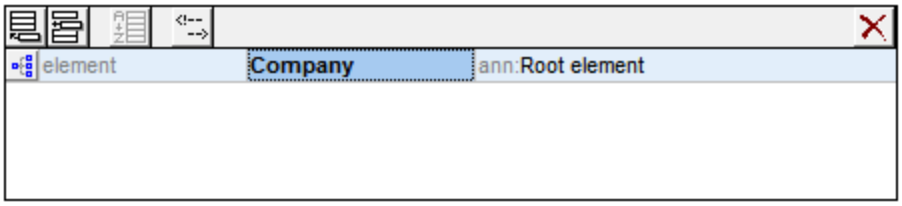
Anmerkung:

- Der XML Schema Namespace wird von XMLSpy automatisch erstellt und erhält das Präfix `xs:`.
- Wenn Sie das XML-Instanzdokument erstellen, muss es den im XML-Schema für das XML-Dokument definierten Target Namespace haben, um gültig zu sein.

2.2.3 Definieren eines Content Model

Sie haben in der Schema-Übersicht bereits ein globales Element mit dem Namen `Company` erstellt. Dieses Element soll ein einziges `Address`-Element und eine unbegrenzte Anzahl an `Person`-Elementen enthalten. Dies ist dann das Content Model von `Company`. Globale Elemente, die Content Models haben können, sind Elemente, complexTypes und Elementgruppen. In XMLSpy wird das Content Model einer globalen Komponente in der Content Model-Ansicht der Schema-Ansicht angezeigt. Um das Content Model einer globalen

Komponente anzeigen und editieren zu können, klicken Sie auf das Symbol "Diagramm anzeigen" links neben der globalen Komponente.

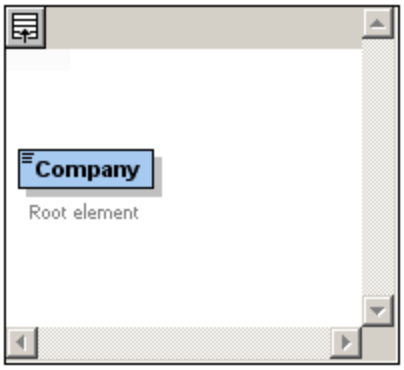


In diesem Abschnitt werden Sie das Content Model des Elements `Company` erstellen.

Erstellen eines einfachen Content Model

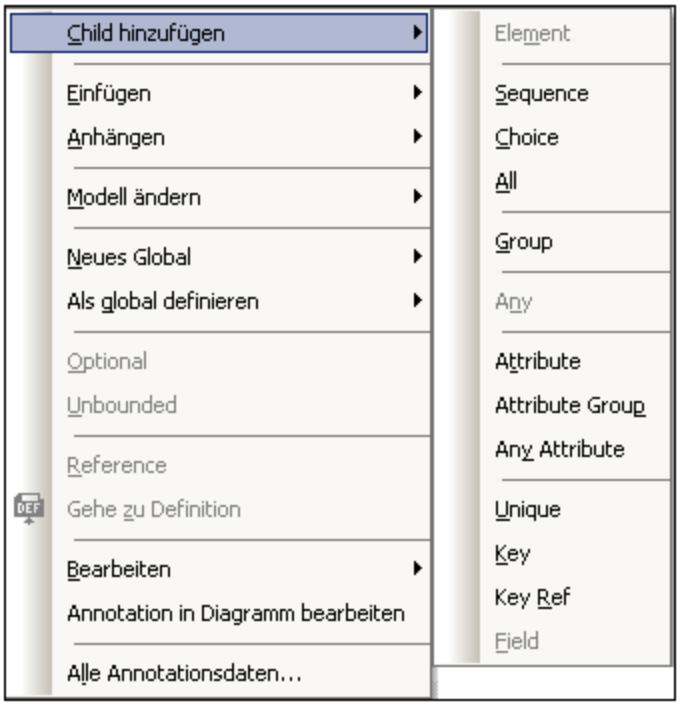
So erstellen Sie das Content Model des Elements `Company`:

1. Klicken Sie in der Schema-Übersicht auf das Symbol "Diagramm anzeigen" links neben dem Element `Company`. Hierdurch wird das Content Model des Elements `Company` angezeigt (*Abbildung unten*), das derzeit noch leer ist. (Alternativ dazu können Sie auch auf den Eintrag `Company` im Komponentenfenster doppelklicken, um das Content Model anzuzeigen.)

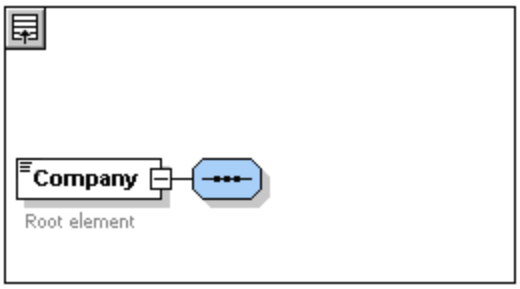


2. Ein Content Model besteht aus Kompositoren und Komponenten. Die Kompositoren definieren die Beziehung zwischen zwei Komponenten. Sie müssen nun im `Company` Content Model einen Child-Kompositor zum Element `Company` hinzufügen, um ein Child-Element hinzufügen zu können. Rechtsklicken Sie dazu auf das Element "Company" und wählen Sie im Kontextmenü die Option **Child hinzufügen | Sequence**. (Sequence, Choice und All sind die drei Kompositoren, die in einem Content Model verwendet werden können.)

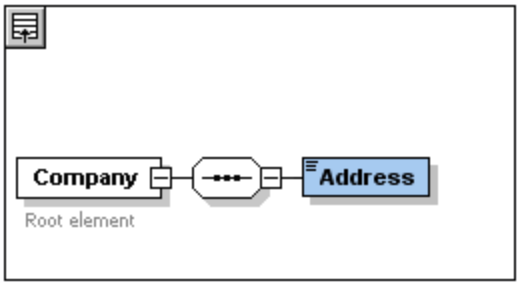
© 2016-2022 Altova GmbH



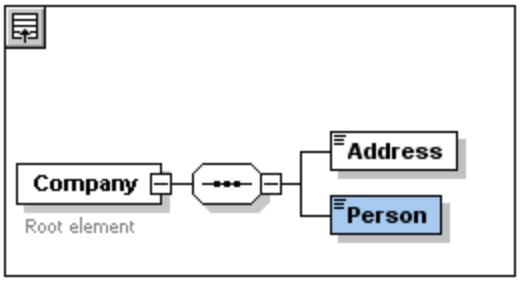
Daraufhin wird der Sequence-Kompositor eingefügt, der definiert, dass die Komponenten, die darauf folgen in der angegebenen Reihenfolge erscheinen müssen.



3. Rechtsklicken Sie auf den Sequence-Kompositor und wählen Sie **Child hinzufügen | Element**. Daraufhin wird ein unbenanntes Element hinzugefügt.
4. Benennen Sie das Feld `Address` und bestätigen Sie mit der Eingabetaste.

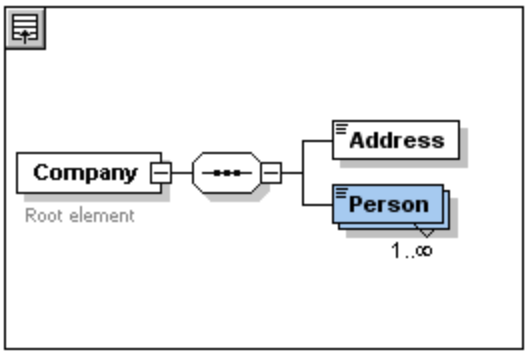


5. Rechtsklicken Sie erneut auf den Sequence-Kompositor und wählen Sie **Child hinzufügen | Element**, benennen Sie die neue Elementkomponente `Person`.



Sie haben nun ein Schema mit einem Eintrag für die Firmenadresse und einem Eintrag für einen Angestellten (`Person`) definiert. Wir müssen nun die Anzahl der `Person`-Elemente erhöhen.

6. Klicken Sie mit der rechten Maustaste auf das Element `Person` und wählen Sie **Unbounded** aus dem Kontextmenü. Unter dem Element `Person` wird nun die Anzahl der zulässigen Instanzen angezeigt: 1 bis unendlich.

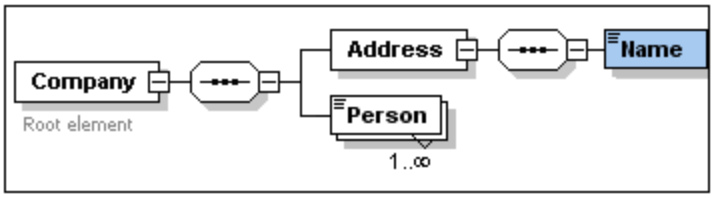


Sie können alternativ dazu in der Details-Eingabehilfe die Felder `minOcc` und `maxOcc` direkt editieren und die Anzahl der zulässigen Instanzen angeben, in diesem Fall 1 bzw. unbounded.

Hinzufügen von zusätzlichen Ebenen zur Struktur des Content Model

Bisher enthält das Content Model, das Sie erstellt haben, eine Ebene: eine Child-Ebene für das Element `company`, das die Elemente `Address` und `Person` enthält. Nun wollen wir den Inhalt des Elements `Address` definieren, und zwar die Subelemente `Name`, `Street` und `City`. Dies ist eine zweite Ebene. Wir müssen also zum Element `Address` einen Child-Kompositor hinzufügen und anschließend die Elementkomponenten selbst. Gehen Sie dabei folgendermaßen vor:

1. Klicken Sie mit der rechten Maustaste auf das Element `Address`, um das Kontextmenü zu öffnen, und wählen Sie **Child hinzufügen | Sequence**.
2. Klicken Sie mit der rechten Maustaste auf den **Sequence**-Kompositor und wählen Sie **Child hinzufügen | Element**. Benennen Sie die neu erstellte Elementkomponente mit "Name".



 © 2016-2022 Altova GmbH

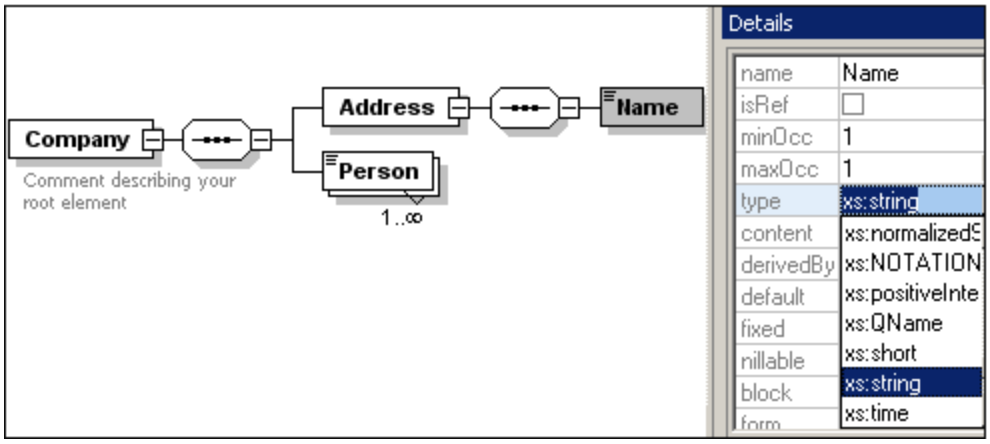
ComplexTypes, simpleTypes und XML Schema-Datentypen

Bisher haben wir noch keinen Elementtyp explizit definiert. Klicken Sie auf das Register **Text** um die Textansicht für Ihr Schema anzuzeigen (*siehe Liste unten*). Sie werden feststellen, dass jedes Mal, wenn ein Sequence-Kompositor eingefügt wurde, das Element `xs:sequence` im Element `xs:complexType` eingefügt wurde. Kurz gesagt, die Elemente `Company` und `Address` sind ComplexTypes, da sie Child-Elemente enthalten. Ein ComplexType ist ein Element, das Attribute oder Elemente enthält.

```
<xs:element name="Company">
  <xs:annotation>
    <xs:documentation>Root element</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Address">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Name"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="Person"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

SimpleType-Elemente dagegen enthalten nur Text und haben keine Attribute. Beim Text kann es sich um Strings, Daten, Zahlen usw. handeln. Wir möchten, dass das Child-Element `Name` von `Address` nur Text enthält. Es ist ein SimpleType, dessen Textinhalt wir durch Restriktion auf einen String einschränken möchten. Dies lässt sich durch Verwendung des XML-Schema-Datentyps `xs:string` bewerkstelligen. Um das Element `Name` als diesen Datentyp zu definieren, gehen Sie folgendermaßen vor.

1. Klicken Sie auf das Register **Schema** um zur Schema/WSLD-Ansicht zurückzukehren.
2. Klicken Sie auf `Name`, um das Element auszuwählen.
3. Klicken Sie in der Details-Eingabehilfe auf das Listenfeld `type` in der mittleren Eingabehilfe und wählen Sie `xs:string` aus der Auswahlliste.



Beachten Sie, dass sowohl `minOcc` als auch `maxOcc` den Wert 1 hat, d.h. dieses Element hat nur eine Instanz.

In der Textdarstellung sieht das Element `Name` folgendermaßen aus:

```
<xs:element name="Name" type="xs:string"/>
```

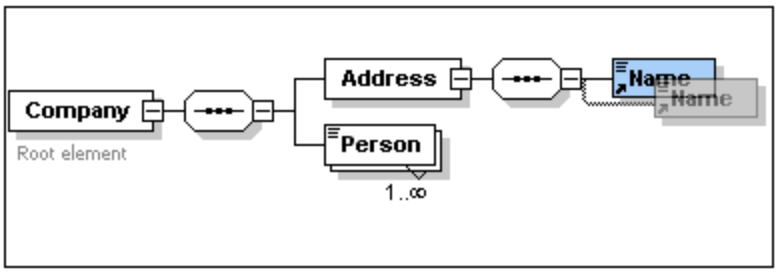
Anmerkung: Ein einfaches Type-Element kann jedem beliebigen von mehreren XML-Schema-Datentypen angehören. In all diesen Fällen erscheint das Symbol für Textinhalt nur im Elementfeld.

2.2.4 Hinzufügen von Elementen mit Drag & Drop

Sie haben Elemente bisher über das Kontextmenü hinzugefügt, das erscheint, wenn Sie mit der rechten Maustaste auf ein Element oder einen Kompositor klicken. Schneller lassen sich Elemente aber mit Drag & Drop erstellen. In diesem Abschnitt lernen Sie, wie man mit Drag & Drop mehr Elemente zur Definition des Elements Address hinzufügt und so diese Definition fertig stellt.

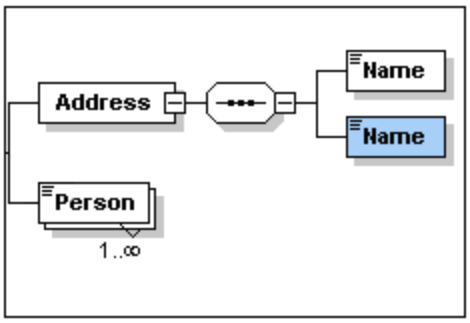
So stellen Sie de Definition des Elements Address mit Hilfe von Drag & Drop fertig:

1. Klicken Sie auf das Element `Name` des Elements `Address`, halten Sie die Maustaste sowie die Strg-Taste gedrückt und ziehen Sie die Maus leicht nach unten. Die Markierung mit einem +Zeichen erscheint. Hierdurch wird angezeigt, dass das Element kopiert werden kann. Es erscheint eine Kopie des Elements sowie eine Konnektorlinie, sodass Sie sehen, wo das Element erstellt wird.

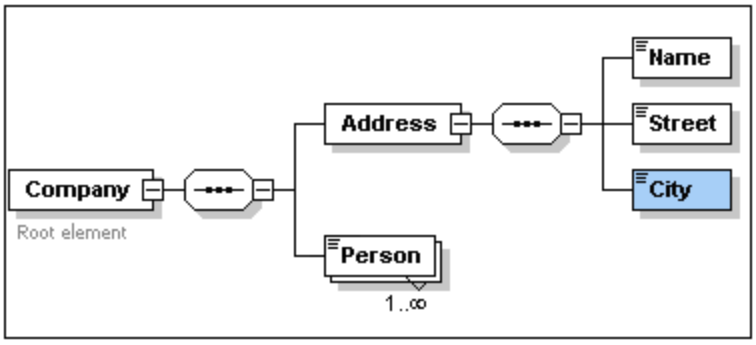


 © 2016-2022 Altova GmbH

2. Lassen Sie nun die Maustaste und die STRG-Taste los um das neue Element in der `Address`-Sequenz anzulegen. Wurde das Element an der falschen Stelle angelegt, so ziehen Sie es einfach mit der Maus unter das Element `Name`.



3. Doppelklicken Sie in das Elementfeld und geben Sie `Street` ein, um den Namen des Elements zu ändern.
4. Wiederholen Sie obige Schritte um ein weiteres Element zu erzeugen. Benennen Sie dieses Feld um und geben Sie ihm den Namen `City`. Das Content Model sollte nun folgenden Inhalt haben:

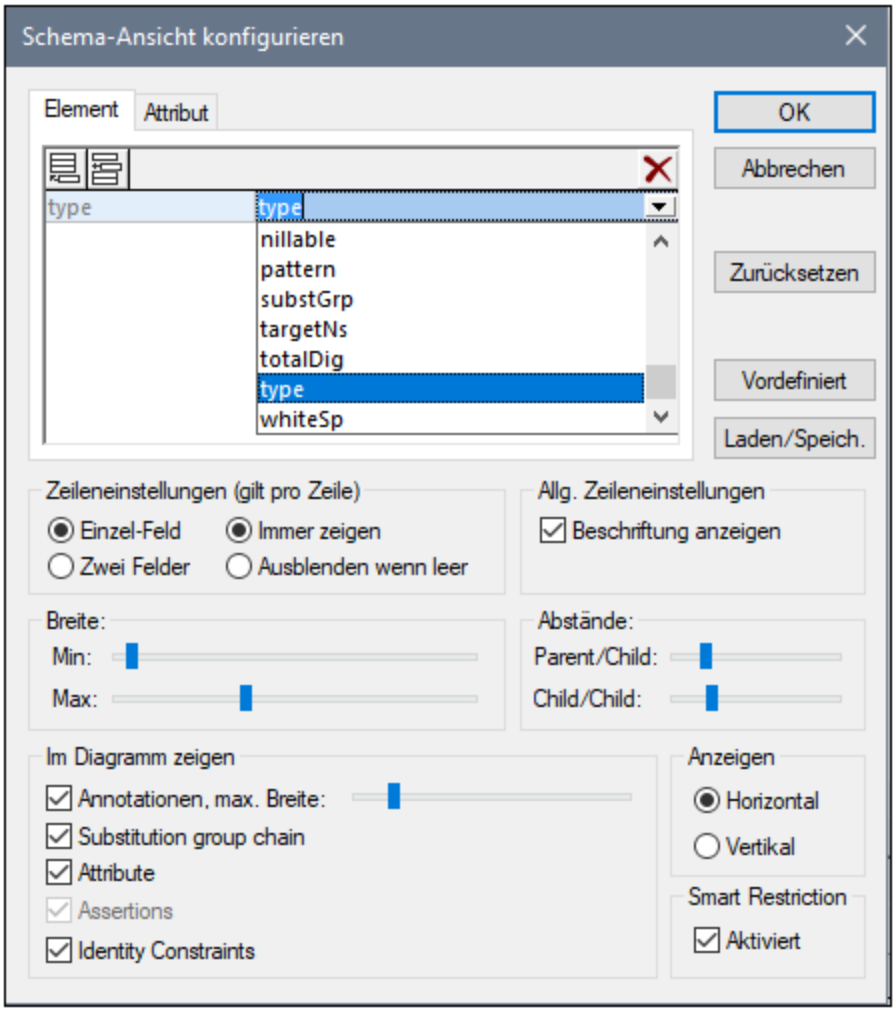


Das Element `Address` enthält eine Sequence mit einem Element `Name`, einem Element `Street` und einem Element `City`, und zwar in der obigen Reihenfolge.

2.2.5 Konfigurieren der Content Model-Ansicht

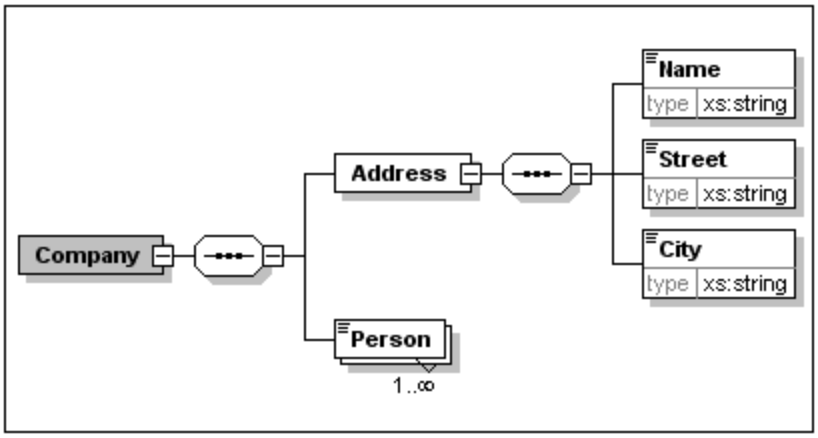
Sie können jetzt das Aussehen der Content Model-Ansicht konfigurieren. Wir wollen die Content Model-Ansicht so konfigurieren, dass der Typ eines Elements im jeweiligen Feld für das Element angezeigt wird. Gehen Sie dabei folgendermaßen vor:

1. Wählen Sie die Content Model-Ansicht einer Komponente aus (klicken Sie auf das Symbol Content Model-Ansicht ).
2. Wenn in der Content Model-Ansicht die Menüoption **Schema-Design | Ansicht konfigurieren** aktiviert ist, wählen Sie den Befehl aus, um das Dialogfeld "Schema-Ansicht konfigurieren" aufzurufen (*Abbildung unten*).

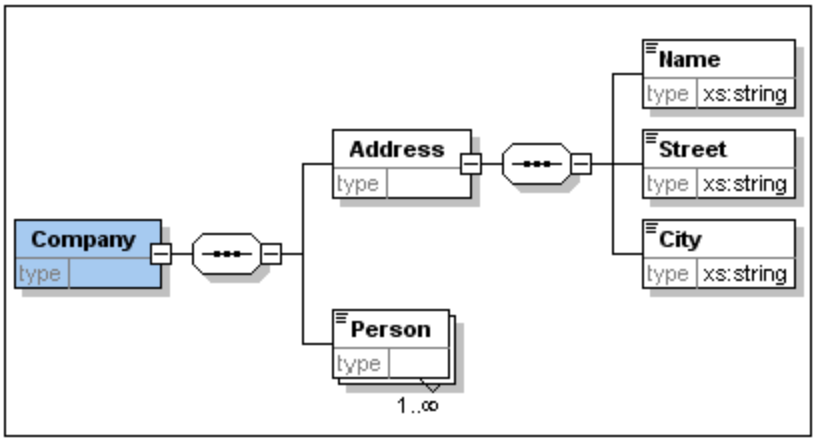


3. Klicken Sie auf dem Register *Element* auf das Symbol **Anhängen** und wählen Sie *type* (*siehe Abbildung*), um diese Eigenschaftsbeschreibungszeilen zu Elementfeldern hinzuzufügen.
4. Wählen Sie im Bereich *Zeileneinstellungen (Gilt pro Zeile)* die Option *Ausblenden wenn leer*, damit die Beschreibung des Datentyps im Elementfeld ausgeblendet wird, wenn das Element keinen Datentyp hat (z.B. wenn das Element ein ComplexType ist). Beachten Sie in der Abbildung unten, dass die Beschreibungszeile für den Typ bei den Elementen Name, Street und City aufscheint, da es sich dabei um SimpleTypes vom Typ xs:string handelt, nicht aber bei den ComplexType-Elementen, da die Option "Ausblenden wenn leer" aktiv ist.

© 2016-2022 Altova GmbH



5. Aktivieren Sie im Bereich *Zeileneinstellungen (Gilt pro Zeile)* die Option *Immer zeigen*.
6. Klicken Sie auf **OK**, um die Änderungen zu speichern.



Beachten Sie, dass die Beschreibungszeile für den Datentyp immer angezeigt wird - selbst in Elementfeldern von ComplexTypes, in denen keine Werte in dieser Zeile angezeigt werden.

Beachten Sie die folgenden Punkte:

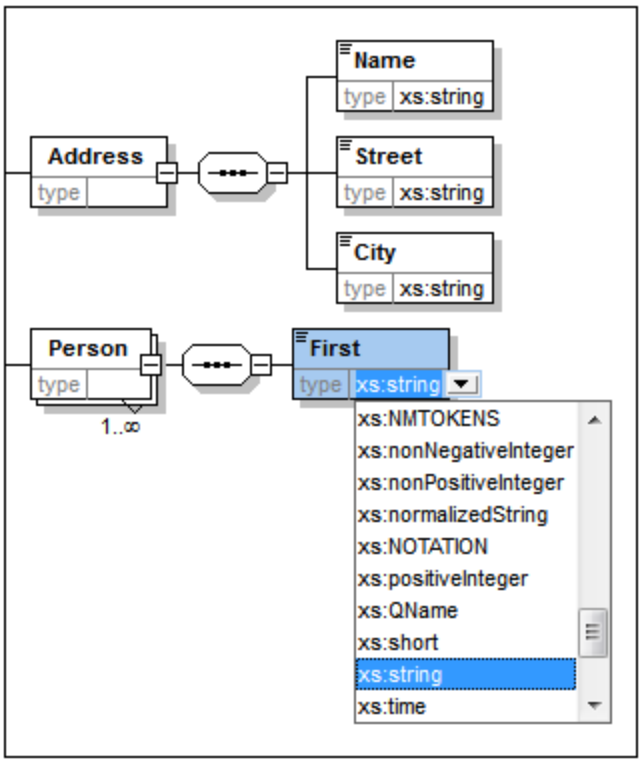
- Die Eigenschaftsbeschreibungszeilen können editiert werden. Werte, die Sie dort eingeben, werden Teil der Elementdefinition.
- Die Einstellungen, die Sie im Dialogfeld "Schema-Ansicht konfigurieren" vornehmen, gelten für die Ausgabe der Schema-Dokumentation ebenso wie für die Druckausgabe.

2.2.6 Fertigstellen des Schemas

Nachdem Sie nun den Inhalt des Elements **`Address`** definiert haben, müssen Sie den Inhalt des Elements `Person` definieren. Das Element **`Person`** soll die folgenden Child-Elemente vom Typ SimpleType enthalten: **`First`**, **`Last`**, **`Title`**, **`PhoneExt`** und **`Email`**. Alle diese Elemente müssen mit Ausnahme des Elements **`Title`** (das optional ist) obligatorisch sein, und alle Elemente müssen in der von Ihnen angegebenen Reihenfolge aufscheinen. Alle Elemente sollen vom Typ **`xs:string`** sein, mit Ausnahme von **`PhoneExt`**, das vom Typ **`xs:integer`** sein muss und maximal 2 Stellen haben darf.

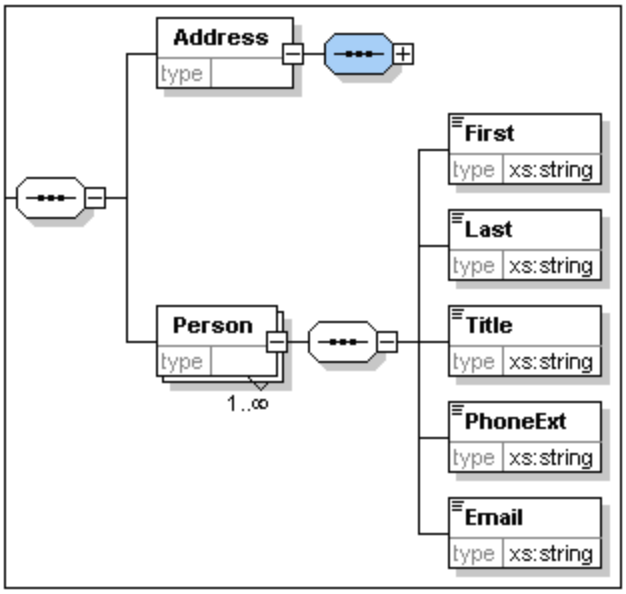
Um das Content Model für **Person** zu erstellen, gehen Sie folgendermaßen vor:

1. Rechtsklicken Sie auf das Element **Person**, um das Kontextmenü aufzurufen und wählen Sie **Child hinzufügen | Sequence**. Der Sequence-Kompositor wird eingefügt.
2. Klicken Sie mit der rechten Maustaste auf den Sequence-Kompositor und wählen Sie **Child hinzufügen | Element**.
3. Nennen Sie das Element **First** und betätigen Sie die **Tabulatortaste** Ihrer Tastatur. Sie gelangen automatisch in das Feld type.



4. Wählen Sie den Eintrag **xs:string** aus der Auswahlliste oder geben Sie ihn in das Feld *type* ein.
5. Erzeugen Sie weitere vier Elemente mittels der Drag & Drop Methode. Benennen Sie diese der Reihe nach **Last, Title, PhoneExt** und **Email**.

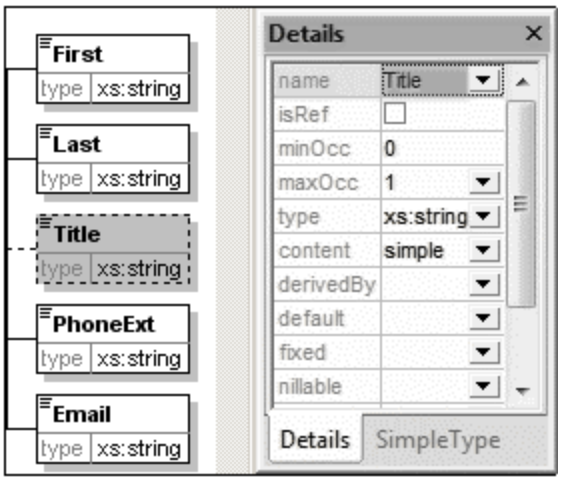
© 2016-2022 Altova GmbH



Anmerkung: Sie können mehrere Elemente auswählen, indem Sie die **Strg**-Taste gedrückt halten, während Sie die Elemente anklicken. Auf diese Art können Sie mehrere Elemente gleichzeitig kopieren.

Ein Element optional machen

Rechtsklicken Sie auf das Element `Title` und wählen Sie **Optional** im Kontextmenü. Die Umrahmung des Elementes wechselt von durchgezogener Linie auf gestrichelt. Hierdurch wird angezeigt, dass dieses Element optional ist.

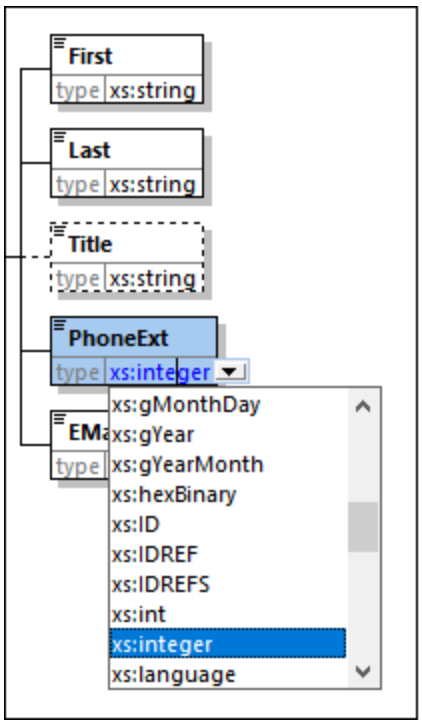


Sie sehen in der Details-Eingabehilfe, dass `minOcc=0` und `maxOcc=1` ist, d.h. das Element ist optional. Alternativ dazu können Sie das Element auch über das Kontextmenü optional machen und für `minOcc=0` einstellen, um das Element zu einem optionalen zu machen.

Den Inhalt eines Elements einschränken

Um das Element `PhoneExt` als Typ **`xs:integer`** mit maximal 2 Stellen zu definieren, gehen Sie folgendermaßen vor:

1. Doppelklicken Sie in das Feld type unter dem Element `PhoneExt` und wählen Sie den Eintrag **`xs:integer`** aus der Auswahlliste oder geben Sie ihn ein (*siehe Abbildung unten*).



2. Änderungen werden in der Facets-Eingabehilfe angezeigt. Doppelklicken Sie in der Facets-Eingabehilfe in das Feld `maxIncl`, geben Sie `99` ein und bestätigen Sie dies durch Drücken der **Eingabetaste**. Hierdurch wird definiert, dass Durchwahlnummern Werte von `0` bis inkl. `99` haben können.
3. Wählen Sie die Menüoption **Datei | Speichern** zum Speichern der Änderungen im Schema.

Anmerkung:

- Beim Auswählen eines XML-Schema-Datentyps vom Typ simpleType (z.B. `xs:string` oder `xs:date`) wird das Content Model automatisch in der Details-Eingabehilfe auf simple (`content = simple`) gesetzt.
- Beim Hinzufügen eines Kompositors zu einem Element (selection, choice oder all), wird das Inhaltsmodell automatisch auf complex (`content = complex`) gesetzt.
- Das verwendete Schema ist als `AddressFirst.xsd` im Ordner Tutorial Ihres XMLSpy-Anwendungsordners abgelegt.

© 2016-2022 Altova GmbH

2.3 XML-Schemas: komplexere Vorgänge

Nachdem Sie nun ein einfaches Schema erstellt haben, können wir uns den komplizierteren Aspekten der Schema-Entwicklung widmen.

Ziel

In diesem Abschnitt werden folgende Punkte behandelt:

- Arbeiten mit complexTypes und simpleTypes 63, die später als die Typen von Schema-Elementen verwendet werden können.
- Erstellen von globalen Elementen 71 und Referenzieren dieser Elemente von anderen Elementen aus
- Erstellen von Attributen 73 und deren Eigenschaften, einschließlich Enumerationswerten.

Wir beginnen diesen Abschnitt mit dem einfachen Schema AddressFirst.xsd, das Sie im ersten Teil dieses Tutorials erstellt haben.

In diesem Abschnitt verwendete Befehle

In diesem Abschnitt des Tutorials wird ausschließlich die Schema-Ansicht verwendet. Dabei kommen die folgenden Befehle zum Einsatz:

	Diagramm anzeigen (oder Content Model-Ansicht anzeigen). Mit diesem Symbol, das in der Schema-Übersicht links von allen globalen Komponenten angezeigt wird, sehen Sie das Content Model der dazugehörigen globalen Komponente.
	Globale Elemente anzeigen. Dieses Symbol befindet sich in der linken oberen Ecke des Content Model. Mit diesem Befehl gelangen Sie zurück zur Schema-Übersicht, in der alle globalen Komponenten angezeigt werden.
	Anhängen. Mit diesem Symbol (in der linken oberen Ecke der Schema-Übersicht) können Sie eine globale Komponente hinzufügen.

2.3.1 Arbeiten mit ComplexTypes und SimpleTypes

Nachdem Sie ein Element definiert haben, möchten Sie es möglicherweise auch an einer anderen Stelle im Schema verwenden. Das könnte z.B. der Fall sein, wenn Sie ein Content Model für Adressen in den USA und Großbritannien definieren. Die beiden Adressformate haben einige Komponenten gemeinsam, wie z.B. die Komponenten Straße (street) und Stadt (city). Andere Komponenten unterscheiden sich wiederum voneinander. Daher empfiehlt es sich, in jedem Adress-Content Model (US und UK) die gemeinsamen Komponenten wiederzuverwenden und die beiden Content Models dann durch die länderspezifschen Komponenten zu ergänzen (wie z.B. ZIP in den USA und "Postal code" in UK). Wir können die gemeinsamen Komponenten zu diesem Zweck als globale complexTypes (oder alternativ dazu jede gemeinsame Komponente als globales Element) erstellen und den globalen complexTyp (oder die globalen Elemente) im Content Model der einzelnen Adresstypen wiederverwenden.

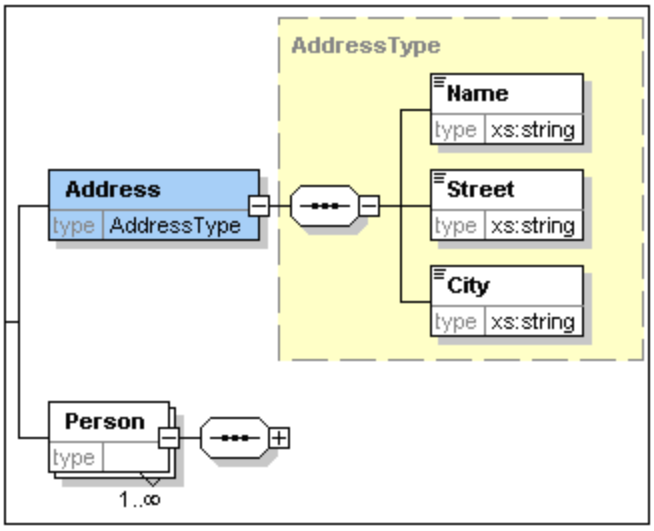
In diesem Abschnitt werden Sie mit globalen ComplexTypes arbeiten. Ein complexType ist eine Typdefinition für Elemente, die andere Elemente und/oder Attribute enthält. Wir werden zuerst einen ComplexType auf globaler Ebene erstellen, diesen anschließend in ein Content Model importieren und dann erweitern. Mehr über globale Elemente erfahren Sie weiter hinten in diesem Tutorial.

Erstellen eines globalen complex type

Das Basiselement Address, das wir zuvor definiert haben (und das die Elemente Name, Street und City enthält), kann in verschiedenen Adressformaten wiederverwendet werden. Wir wollen daher diese Elementdefinition als wiederverwendbaren ComplexType erstellen. Gehen Sie dazu folgendermaßen vor:

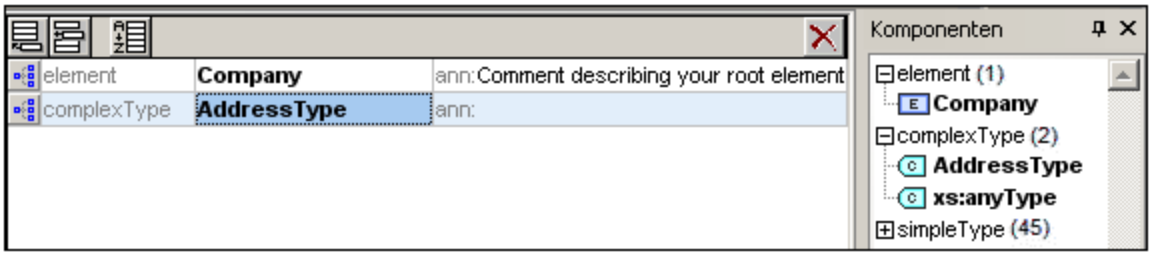
So erstellen Sie einen globales Element vom Typ Complex Type:

1. Klicken Sie in der Content Model-Ansicht mit der rechten Maustaste auf das Element `Address`.
2. Wählen Sie im Kontextmenü nun die Option **Als global definieren | Complex Type**. Es wird ein globaler ComplexType mit dem Namen `AddressType` erstellt, und dem Element `Address` im Content Model `Company` wird dieser Typ zugewiesen. Der Inhalt des Elements `Address` ist das Content Model von `AddressType`, das in einem gelb hinterlegten Kasten angezeigt wird. Beachten Sie, dass der Datentyp des Elements `Address` nun `AddressType` ist.

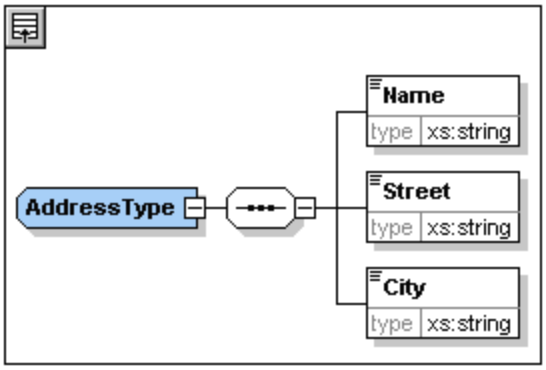


3. Klicken Sie auf das Symbol "Globale Elemente anzeigen"  . Damit kehren Sie zur Schema-Übersicht zurück, in der nun alle globalen Komponenten angezeigt werden.
4. Klicken Sie in der Komponenten-Eingabehilfe auf das Erweiterungssymbol für die **`Element`**- und **`ComplexType`**-Einträge, um die jeweiligen Schema-Konstrukte zu sehen.In der Schema-Übersicht werden nun zwei globale Komponenten angezeigt: das Element `Company` und der ComplexType `AddressType`. Außerdem wird in der Komponenten-Eingabehilfe der ComplexType `AddressType` angezeigt.

© 2016-2022 Altova GmbH



5. Klicken Sie auf das Symbol "Content Model-Ansicht" von `AddressType`, um dessen Content Model anzuzeigen (siehe Abbildung unten). Beachten Sie die Form des ComplexType-Behälters.

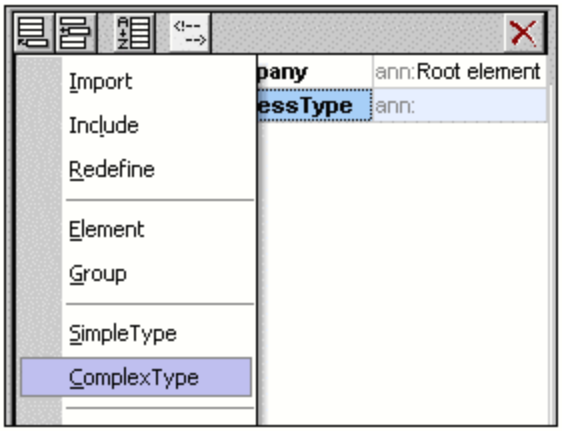


6. Klicken Sie auf das Symbol "Globale Elemente anzeigen", um zur Schema-Übersicht zurückzukehren.

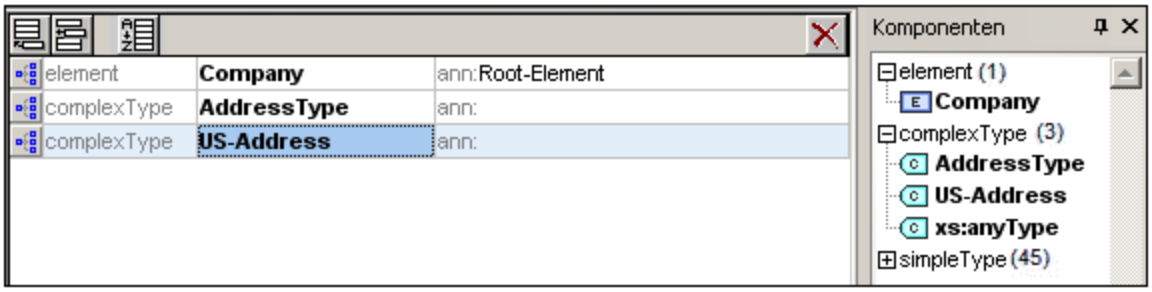
Erweitern einer "complex type" Definition

Nun werden wir die globale Komponente "AddressType" verwenden, um zwei Arten von länderspezifischen Adressen zu erstellen. Hierzu wird zuerst ein neuer complexType erzeugt, welcher auf der Komponente "AddressType" basiert. Anschließend wird diese Definition erweitert. Gehen Sie dazu folgendermaßen vor:

1. Wechseln Sie zur Schema-Übersicht (Wenn Sie sich in der Content Model-Ansicht befinden, klicken Sie auf "Globale Elemente anzeigen").
2. Klicken Sie auf das Symbol **Anhängen** (oben links im Komponentenfenster). Daraufhin wird das folgende Menü angezeigt:

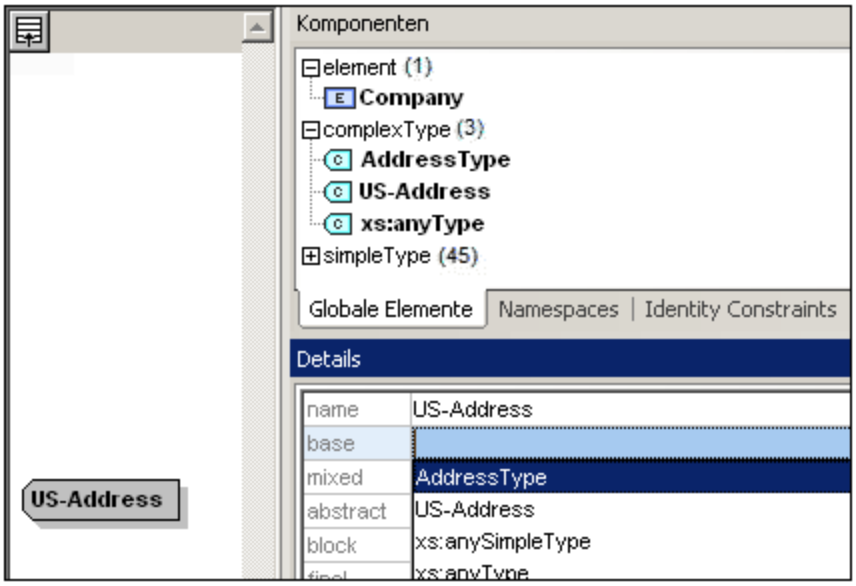


3. Wählen Sie im Menü **ComplexType** aus. Eine neue Leerzeile wird in der Komponentenliste angehängt, in der Sie den Namen der Komponente eingeben können.
4. Geben Sie `US-Address` ein und bestätigen Sie mit der **Eingabetaste**. (Sollten Sie den Bindestrich "-" beim Eingeben vergessen, und stattdessen ein Leerzeichen verwenden, wird der Name in Rot angezeigt. D.h., ein nicht erlaubtes Schriftzeichen wurde verwendet.)

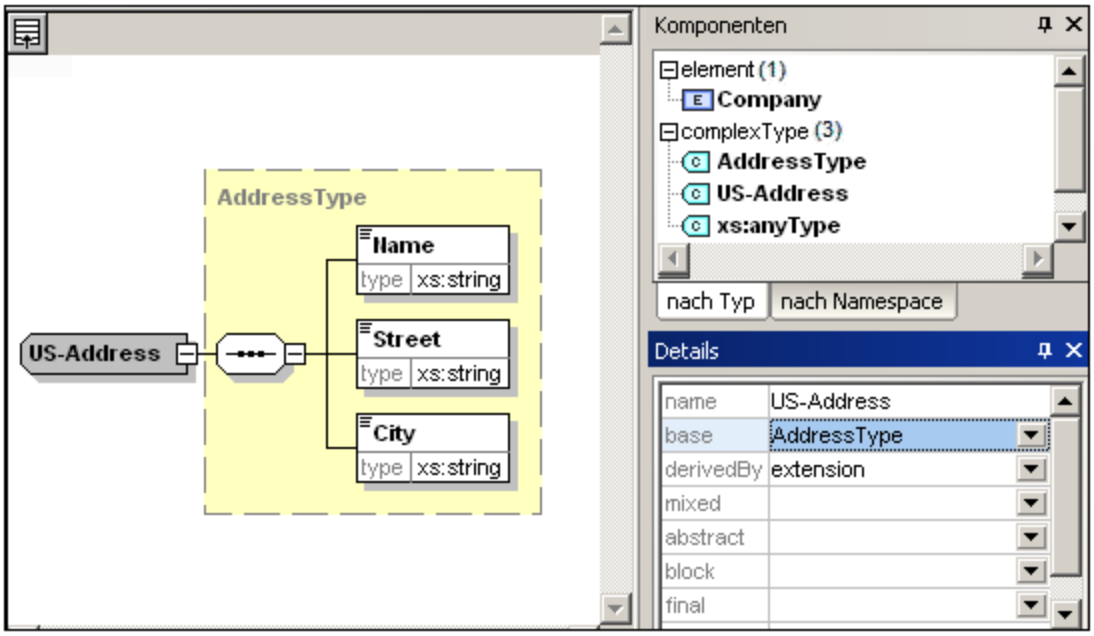


5. Klicken Sie auf das Symbol "Content Model-Ansicht"  in der Zeile `US-Address`, um das Content Model des neuen ComplexType zu sehen. Das Content Model ist leer (*siehe Abbildung unten*).
6. Klicken Sie in der Details-Eingabehilfe auf das Listenfeld base und wählen Sie `AddressType`.

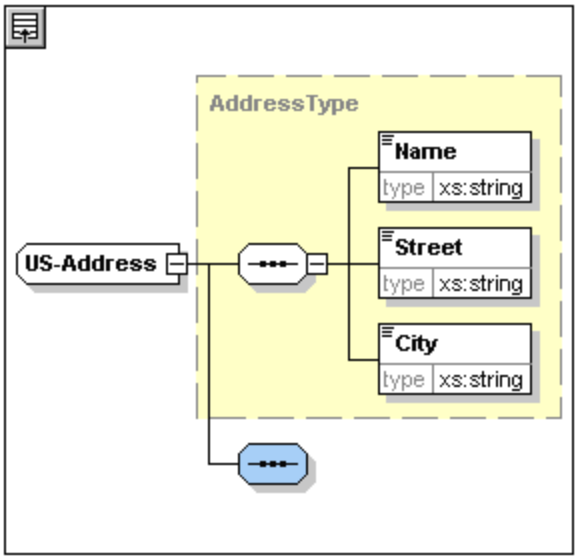
 © 2016-2022 Altova GmbH



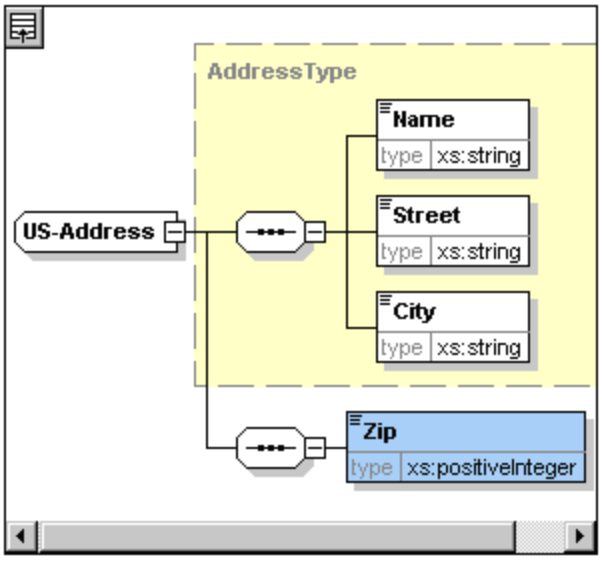
In der Content Model-Ansicht wird nun das Content Model `AddressType` als Content Model von US-Address (*siehe Abbildung unten*) angezeigt.



7. Wir können nun das Content Model des ComplexType `US-Address` um ein ZIP Code-Element erweitern. Klicken Sie dazu mit der rechten Maustaste auf das Element US-Address und wählen Sie **Child hinzufügen | Sequence**. Ein neuer Sequence-Kompositor wird außerhalb des Kastens AddressType (*Abbildung unten*) angezeigt; d.h., es handelt sich um die Erweiterung zu einem Element.



8. Klicken Sie mit der rechten Maustaste auf den neuen Sequence-Kompositor und wählen Sie **Child hinzufügen | Element**.
9. Benennen Sie das neu erstellte Element mit `Zip` und drücken Sie die **Tabulatortaste**. Daraufhin wird der Cursor in das Wertefeld der Typbeschreibungszeile gesetzt.
10. Wählen Sie in der Auswahlliste `xs:positiveInteger` aus und bestätigen Sie mit der **Eingabetaste**.



Sie haben nun einen ComplexType namens `US-Address`, der auf dem ComplexType `AddressType` basiert und diesen um einen ZIP Code (Postleitzahl) erweitert.

Globale SimpleTypes

Genau wie der ComplexType `US-Address` auf dem ComplexType `AddressType` basiert, kann ein Element auch auf einem SimpleType basieren. Der Vorteil ist derselbe wie bei globalen ComplexTypes: Der SimpleType kann wiederverwendet werden. Dazu muss der SimpleType zuerst als globales Element definiert werden. In diesem Tutorial werden wir ein Content Model für US-Bundesstaaten in einem SimpleType definieren. Dieser SimpleType wird als Basis für ein weiteres Element verwendet.

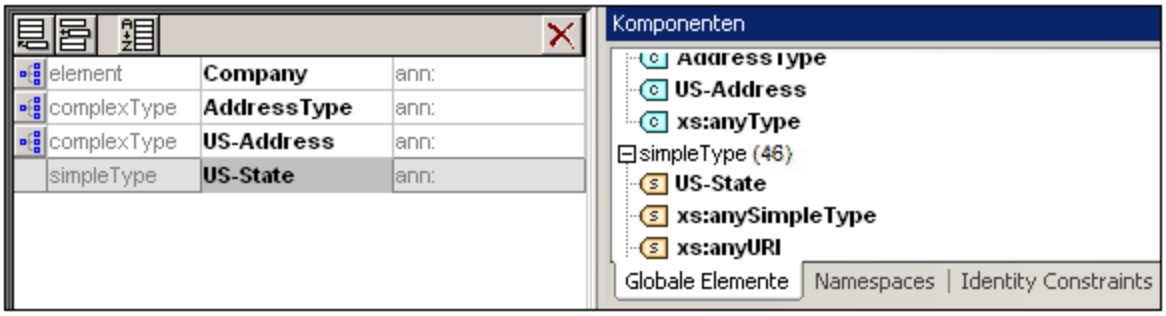
© 2016-2022 Altova GmbH

Erstellen eines globalen SimpleType

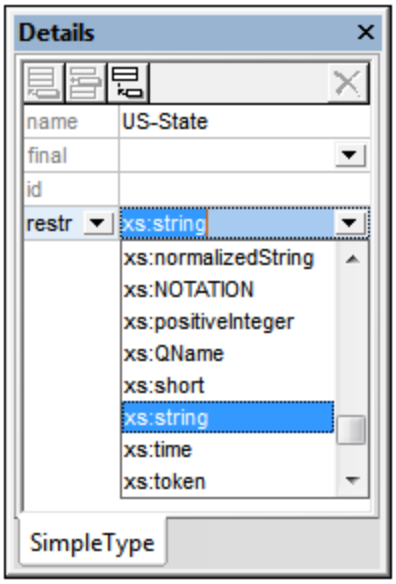
Um einen globalen SimpleType zu erstellen, müssen Sie einen neuen SimpleType an die Liste der globalen Komponenten anhängen, diesen benennen und seinen Datentyp definieren.

So erstellen Sie einen globalen SimpleType:

1. Wechseln Sie in die Schema-Übersicht (Klicken Sie auf "Globale Elemente anzeigen" , wenn Sie sich in der Content Model-Ansicht befinden).
2. Klicken Sie auf das Symbol Anhängen und wählen Sie im Kontextmenü **Simple Type**.
3. Geben Sie `US-State` als Elementname ein.
4. Bestätigen Sie mit der **Eingabetaste**. Der SimpleType `US-State` wird erstellt und in der Komponenten-Eingabehilfe in die Liste der SimpleTypes aufgenommen (Klicken Sie auf das Erweiterungssymbol des SimpleType-Eintrags, um es zu sehen).



5. Platzieren Sie in der Details-Eingabehilfe (*Abbildung unten*) den Cursor in das Wertefeld von `restr` und geben Sie `xs:string` ein oder wählen Sie `xs:string` in der Dropdown-Liste im Wertefeld `restr` aus.

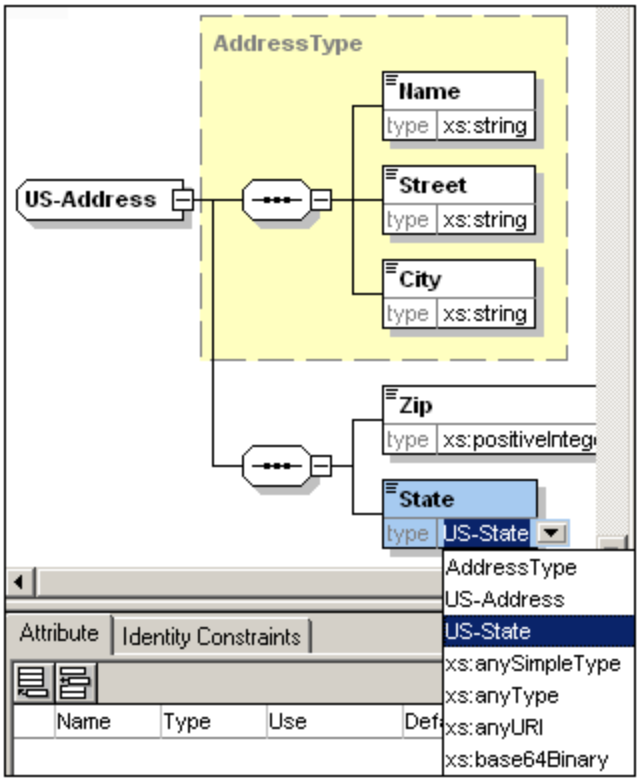


Damit wird ein SimpleType namens `US-State` vom Datentyp `xs:string` erstellt. Diese globale Komponente kann nun im Content Model von `US-Address` verwendet werden.

Verwendung eines globalen SimpleType in einem Content Model

Ein globaler SimpleType kann in einem Content Model dazu verwendet werden, den Typ einer Komponente zu definieren. Wir werden `US-State` verwenden, um im Content Model von `US-Address` ein Element namens `State` zu definieren. Gehen Sie dazu folgendermaßen vor:

1. Klicken Sie in der Schema-Übersicht auf das Symbol "Content Model-Ansicht" von `US-Address`.
2. Klicken Sie mit der rechten Maustaste auf den unteren Sequence-Kompositor und wählen Sie **Child hinzufügen | Element**.
3. Geben Sie `State` als Elementnamen ein.
4. Drücken Sie die **Tabulatortaste**, um den Cursor in das Wertefeld der Typbeschreibungszeile zu setzen.
5. Wählen Sie in der Dropdown-Liste den Eintrag `US-State` aus.



Das Element State basiert nun auf dem SimpleType `US-State`.

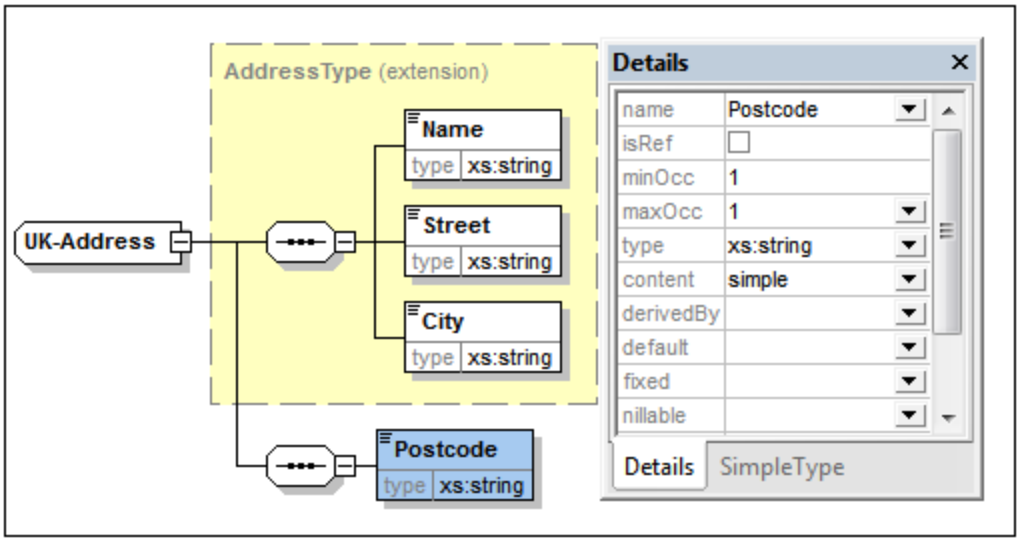
Erstellen eines zweiten ComplexType auf Basis von AddressType

Wir erstellen nun einen globalen ComplexType für Adressen in Großbritannien. Der ComplexType basiert auf dem Element `AddressType` und wird erweitert, bis er dem englischen Adressformat entspricht.

Gehen Sie folgendermaßen vor:

© 2016-2022 Altova GmbH

1. Erstellen Sie in der Schema-Übersicht auf Basis von `AddressType` einen globalen ComplexType mit dem Namen `UK-Address`. (`base=AddressType`).
2. Hängen Sie in der Content Model-Ansicht von `UK-Address` das neue Element `Postcode` an und weisen Sie ihm den Typ `xs:string` zu. bDas Content Model von UK-Address sollte nach erfolgreicher Erstellung folgendermaßen aussehen:

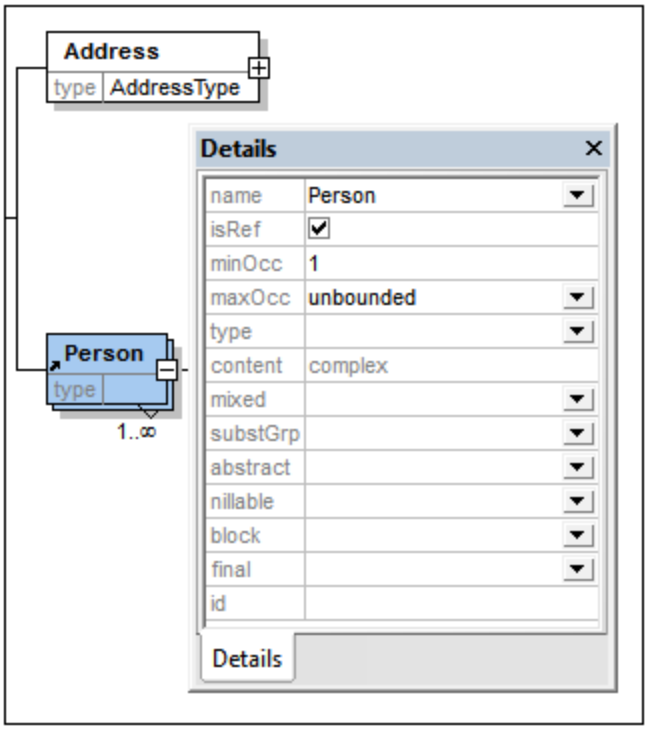


Anmerkung: In diesem Abschnitt haben Sie globale Simple und ComplexTypes erstellt und anschließend in Content Model-Definitionen verwendet. Der Vorteil von globalen Typen liegt darin, dass diese in mehreren Definitionen wiederverwendet werden können.

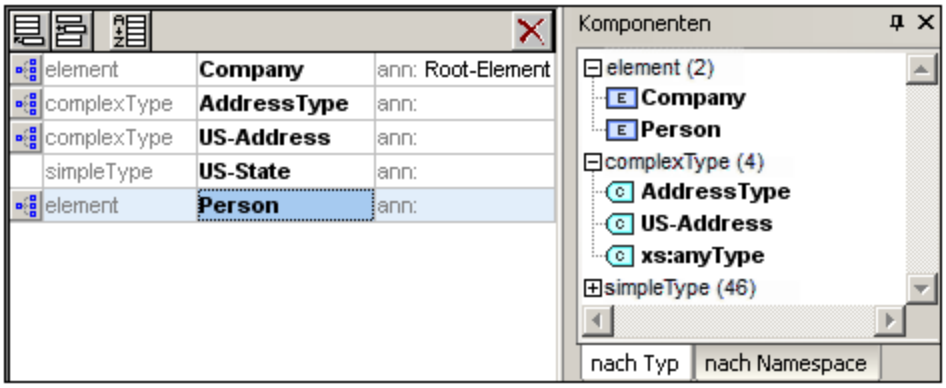
2.3.2 Referenzieren von globalen Elementen

In diesem Abschnitt werden wir das lokal definierte Element Person als globales Element definieren und es dann vom Element Company aus referenzieren.

1. Wechseln Sie in die Schema-Übersicht  (Globale Elemente anzeigen).
2. Klicken Sie auf das Symbol "Diagramm anzeigen"  in der Zeile Company.
3. Klicken Sie mit der rechten Maustaste auf das Element Person und wählen Sie Als global definieren | Element. Ein kleines "link"-Symbol wird in das Element Person eingefügt, das anzeigt, dass das Element das als global deklarierte Element Person referenziert. Das Kontrollkästchen `isRef` in der Details Eingabehilfe ist nun auf aktiv gesetzt.

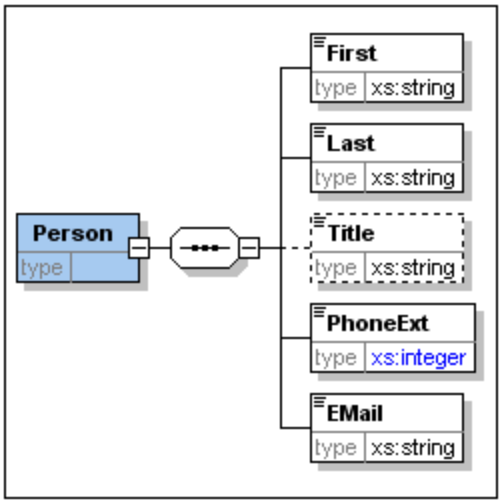


4. Klicken Sie auf das Symbol "Globale Elemente anzeigen"  um zur Schema-Übersicht zurückzukehren. Das Element Person wird nun als globales Element aufgelistet und erscheint auch in der Komponenten-Eingabehilfe.



5. Doppelklicken Sie in der Komponenten-Eingabehilfe auf das Element Person, um das Content Model des globalen Elements Person anzuzeigen.

 © 2016-2022 Altova GmbH



Bitte beachten Sie, dass das Kästchen für das globale Element kein Link-Symbol hat, da es sich hierbei um das referenzierte Element und nicht das referenzierende Element handelt. Nur das referenzierende Element ist mit einem Link-Symbol gekennzeichnet.

Anmerkung:

- Ein Element, das ein globales Element referenziert, muss denselben Namen haben, wie das globale Element, das es referenziert.
- Globale Deklarationen geben keine Auskunft über die Positionierung der Elemente innerhalb des XML-Dokumentes. Sie bestimmen nur den Inhalt des jeweiligen Elementes. Nur wenn eine globale Deklaration von einer anderen Komponente aus referenziert wird, wird ihre Position im XML-Dokument definiert.

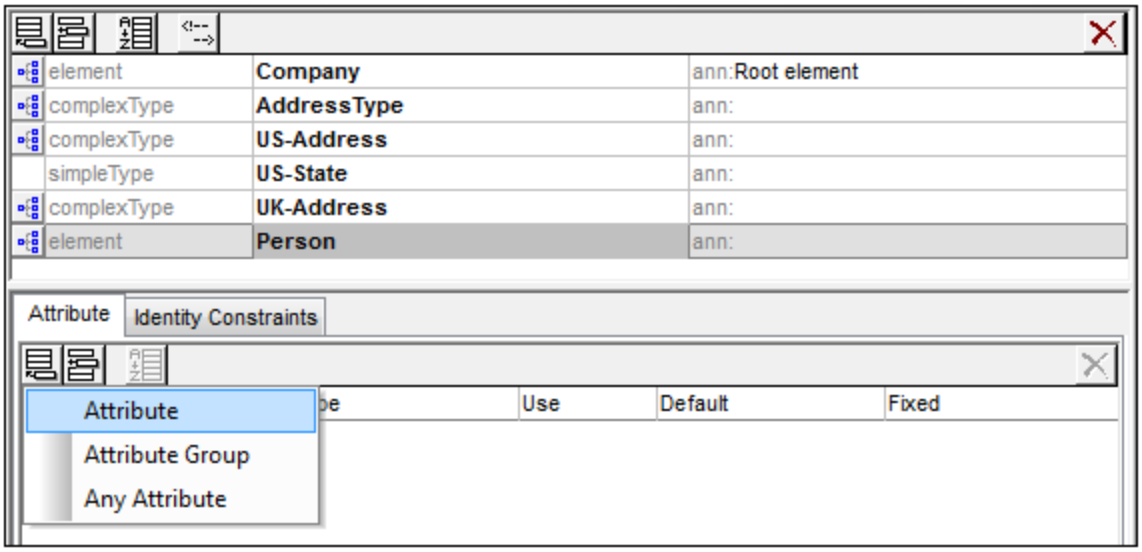
Ein global deklariertes Element kann an mehreren Stellen wiederverwendet werden. Es unterscheidet sich insofern von einem global deklarierten ComplexType, als sein Content Model nicht geändert werden kann, ohne dass auch das globale Element selbst geändert wird. Wenn Sie das Content Model eines Elements ändern, das ein globales Element referenziert, so wird auch das Content Model des globalen Elements geändert und damit auch das Content Model aller anderen Elemente, die das globale Element referenzieren.

2.3.3 Attribute und Attribut-Enumerationen

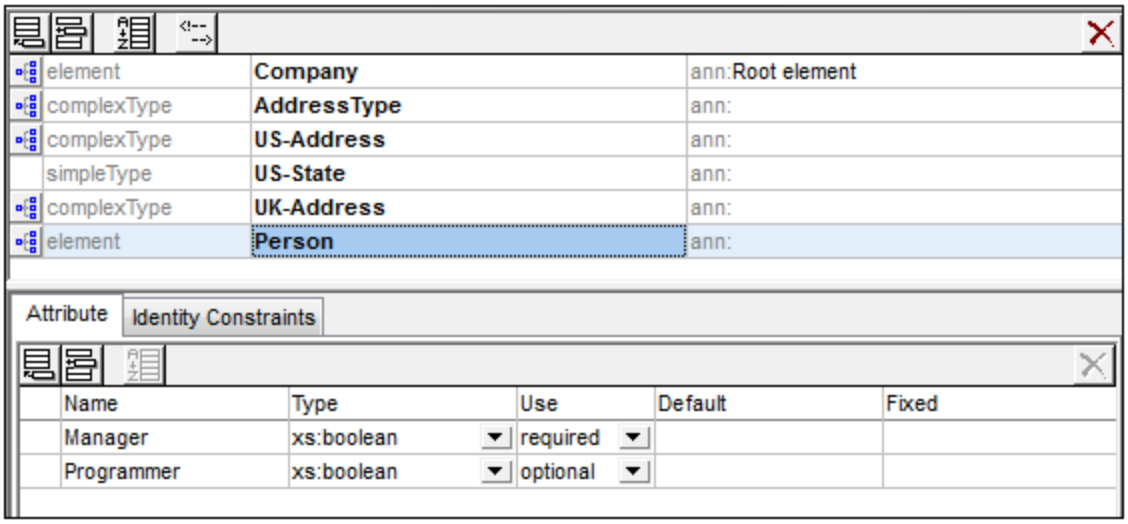
In diesem Abschnitt lernen Sie, wie man Attribute und Enumerationen für Attribute erstellt.

Definieren von Element-Attributen

1. Klicken Sie in der Schema-Übersicht auf das Element `Person` um es aktiv zu setzen.
2. Klicken Sie auf das Symbol "Anhängen" , (oben links im Fenster "Attribute/Identity Constraints", das den unteren Bereich des Schema-Übersichtsfensters einnimmt) und wählen Sie den Eintrag "Attribute".



3. Geben Sie Manager in das Namensfeld ein.
4. Wählen Sie in der Dropdown-Liste Type den Eintrag `xs:boolean`.
5. Wählen Sie in der Dropdown-Liste Use den Eintrag required.



6. Erstellen Sie nun auf dieselbe Art ein Attribut Programmer mit einem Type=`xs:boolean` und Use=`optional`.

Definieren von Enumerationen für Attribute

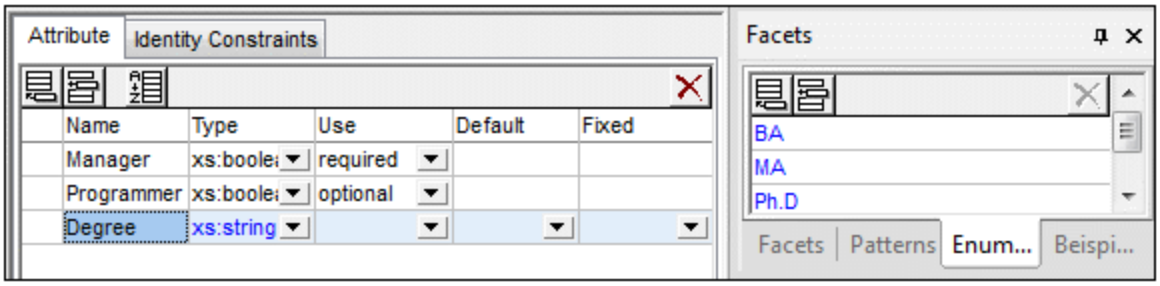
Enumerationen sind Werte, die bei einem bestimmten Attribut zulässig sind. Wenn der Wert des Attributs im XML-Instanzdokument nicht in einer im XML-Schema definierten Enumeration enthalten ist, ist das Dokument ungültig. Wir werden Enumerationen für das Attribut Degree des Elements `Person` erstellen.

Gehen Sie folgendermaßen vor:

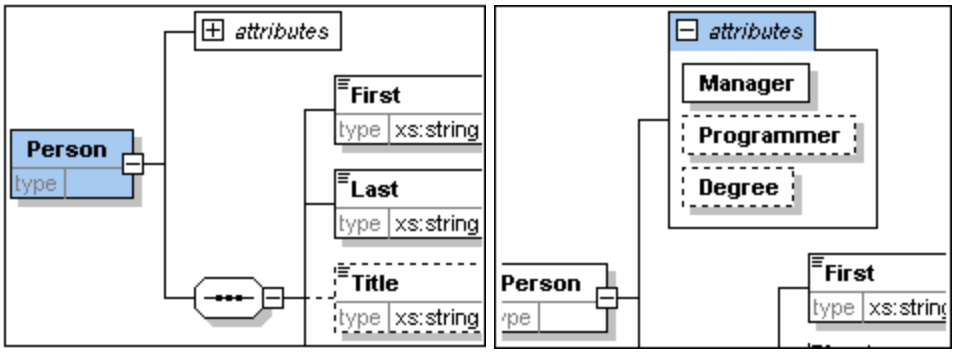
1. Klicken Sie in der Schemaübersicht auf das Element `Person`, um es zum aktiven zu machen.

 © 2016-2022 Altova GmbH

2. Klicken Sie auf das Anhängen-Symbol oben links im Attribut-Fenster und wählen Sie den Eintrag "Attribute".
3. Geben Sie Degree als Attributsnamen ein und wählen Sie `xs:string` als Attribut-Type.
4. Klicken Sie in der Facets-Eingabehilfe auf das Register Enumerations (siehe Abbildung unten).



5. Klicken Sie auf dem Register "Enumerations" auf das Anhängen-Symbol .
6. Geben Sie `BA` (Bachelor of Arts) ein und bestätigen Sie mit der Eingabetaste.
7. Mit derselben Methode geben Sie zwei weitere Werte in die Enumerations-Liste ein: `MA` und `PhD`.
8. Klicken Sie auf das Symbol "Content Model Ansicht" des Elements `Person`.



Die zuvor definierten Attribute werden in der Content Model-Ansicht angezeigt. Wenn Sie auf das Erweiterungssymbol klicken, werden alle für dieses Element definierten Attribute angezeigt. Sie den Anzeigemodus und das Register "Attribute" ein- und ausblenden, indem Sie die Menüoption **Schema-Design | Ansicht konfigurieren** wählen und das Kontrollkästchen für Attribute aktivieren bzw. deaktivieren.

9. Klicken Sie auf das Symbol "Globale Elemente anzeigen", um zur Schema-Übersicht zurückzukehren.

Speichern des fertigen Schemas

Anmerkung: Bevor Sie Ihre Schema-Datei speichern, sollten Sie zuerst die vorhandene Datei `AddressLast.xsd`, die mit XMLSpy mitgeliefert wird, umbenennen (z.B. in `AddressLast_original.xsd`), um sie nicht zu überschreiben.

Speichern Sie das fertige Schema unter einem Namen Ihrer Wahl (**Datei | Speichern unter**). Wir empfehlen Ihnen, es unter dem Namen `AddressLast.xsd` zu speichern, da die XML-Datei, die Sie im nächsten Teil des Tutorials erstellen werden, auf dem Schema `AddressLast.xsd` basieren wird.

2.4 XML-Schemas: XMLSpy-Funktionen

Nach Fertigstellung Ihres XML-Schemas sollten Sie sich nun mit einigen Navigationskürzeln[76] vertraut machen und etwas über die Schemadokumentation[78], die Sie in XMLSpy generieren können, erfahren. Diese Themen werden in den folgenden Unterabschnitten erörtert.

In diesem Abschnitt verwendete Befehle

Wir verwenden in diesem Abschnitt des Tutorials ausschließlich die Schema-Ansicht. Dabei werden die folgenden Befehle verwendet:

	Diagramm anzeigen (oder Content Model-Ansicht anzeigen). Mit diesem Symbol, das in der Schema-Übersicht links von allen globalen Komponenten angezeigt wird, sehen Sie das Content Model der dazugehörigen globalen Komponente.

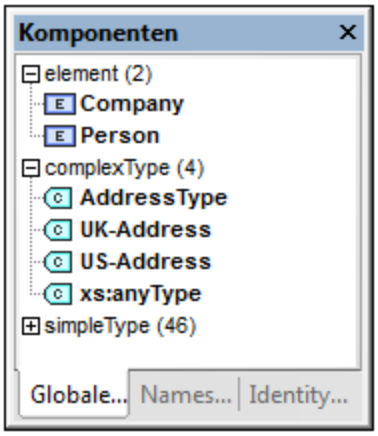
2.4.1 Schemanavigation

In diesem Abschnitt wird beschrieben, wie Sie effizient innerhalb der Schema-Ansicht navigieren können. Probieren Sie diese Navigationsverfahren am besten aus, um sich damit vertraut zu machen.

Anzeigen des Content Model einer globalen Komponente

Globale Komponenten, die ein Content Model haben können, sind ComplexTypes, Elemente und Elementgruppen. Die Content Model-Ansicht dieser Komponenten lässt sich auf folgende Arten aufrufen:

- Klicken Sie in der Schema-Übersicht auf das Symbol "Diagramm anzeigen"  links vom Namen der Komponente.
- Doppelklicken Sie entweder in der Schema-Übersicht oder in der Content Model-Ansicht in der Komponenten-Eingabehilfe auf das Element, den ComplexType oder die Elementgruppe (*siehe Abbildung unten*). Daraufhin wird der Inhalt des Content Model dieser Komponente angezeigt.



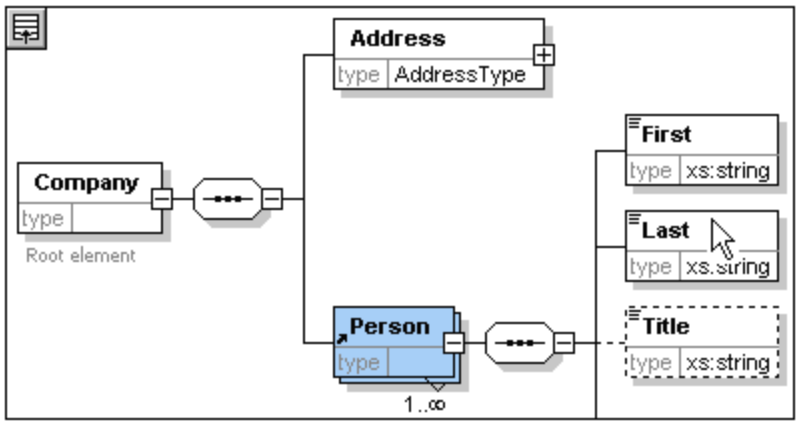
Wenn Sie in der Komponenten-Eingabehilfe auf einer der anderen globalen Komponenten (SimpleType, Attribut, Attributgruppe) doppelklicken, erscheint diese Komponente in der Schema-Übersicht (da eine solche Komponente kein Content Model hätte) markiert.

© 2016-2022 Altova GmbH

Das Anzeigen der Komponenten durch Doppelklick funktioniert in der Komponenten-Eingabehilfe sowohl auf dem Register "Globale Elemente" als auch auf dem Register "Namespaces".

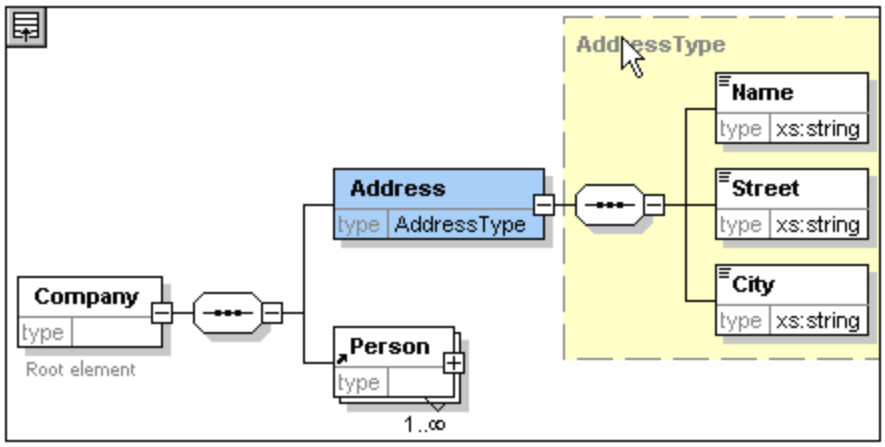
So gelangen Sie von einem referenzierenden Element aus zur Definition eines globalen Elements:
Wenn ein Content Model ein Element enthält, das ein globales Element referenziert, können Sie das Content Model dieses globalen Elements oder das seiner darin enthaltenen Komponenten direkt aufrufen, indem Sie die Strg-Taste gedrückt halten, während Sie auf das gewünschte Element doppelklicken.

Wenn Sie z.B. während der Ansicht des Content Model von `Company` die **Strg**-Taste gedrückt halten, während Sie auf `Last` doppelklicken, wird das Content Model `Person` geöffnet und das Element `Last` darin wird markiert.

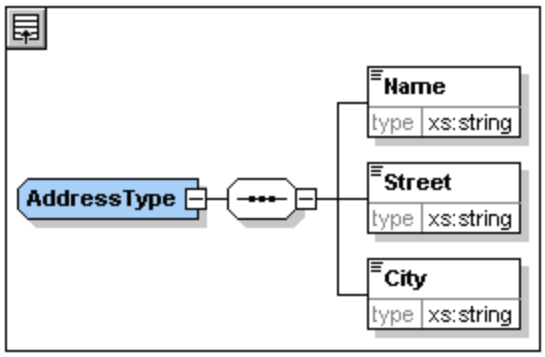


Wenn das Element `Last` markiert ist, werden alle seine Eigenschaften sofort in den entsprechenden Eingabehilfen und Informationsfenstern angezeigt.

So gelangen Sie zur Definition eines ComplexType
ComplexTypes werden oft als Elementtyp in einem Content Model verwendet. Um die Definition eines ComplexType direkt von einem Content Model aus aufzurufen, doppelklicken Sie im gelben Kästchen auf den Namen des ComplexType (siehe Mauszeiger in der Abbildung unten).



Auf diese Art gelangen Sie zur Content Model-Ansicht des ComplexType.



Anmerkung: Wie auch bei referenzierten globalen Elementen können Sie direkt zu einem Element innerhalb der ComplexType-Definition gehen, indem Sie die **Strg**-Taste gedrückt halten und auf das gewünschte Element im Content Model, das den ComplexType enthält, doppelklicken.

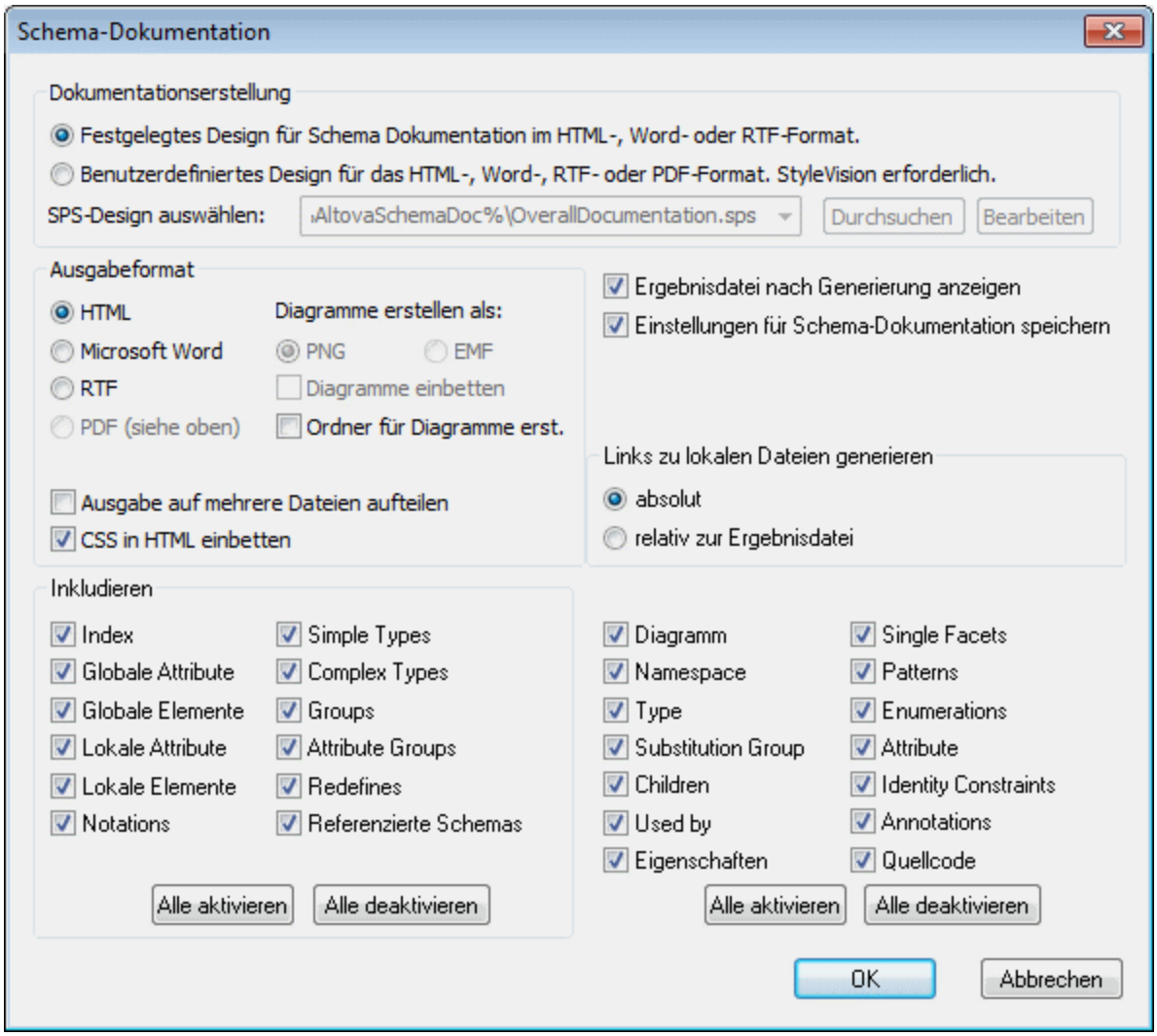
2.4.2 Schema-Dokumentation

XMLSpy verfügt über Funktionen zur Erstellung einer detaillierten Dokumentation von XML-Schemas im HTML- und MS Word-Format. Sie können die Komponenten und die Genauigkeit der Dokumentation auswählen. Miteinander in Zusammenhang stehende Komponenten werden im HTML- und im Word-Dokument durch Hyperlinks verbunden. Um Dokumentation im MS Word-Format zu generieren, muss MS Word auf Ihrem Computer (oder Netzwerk) installiert sein.

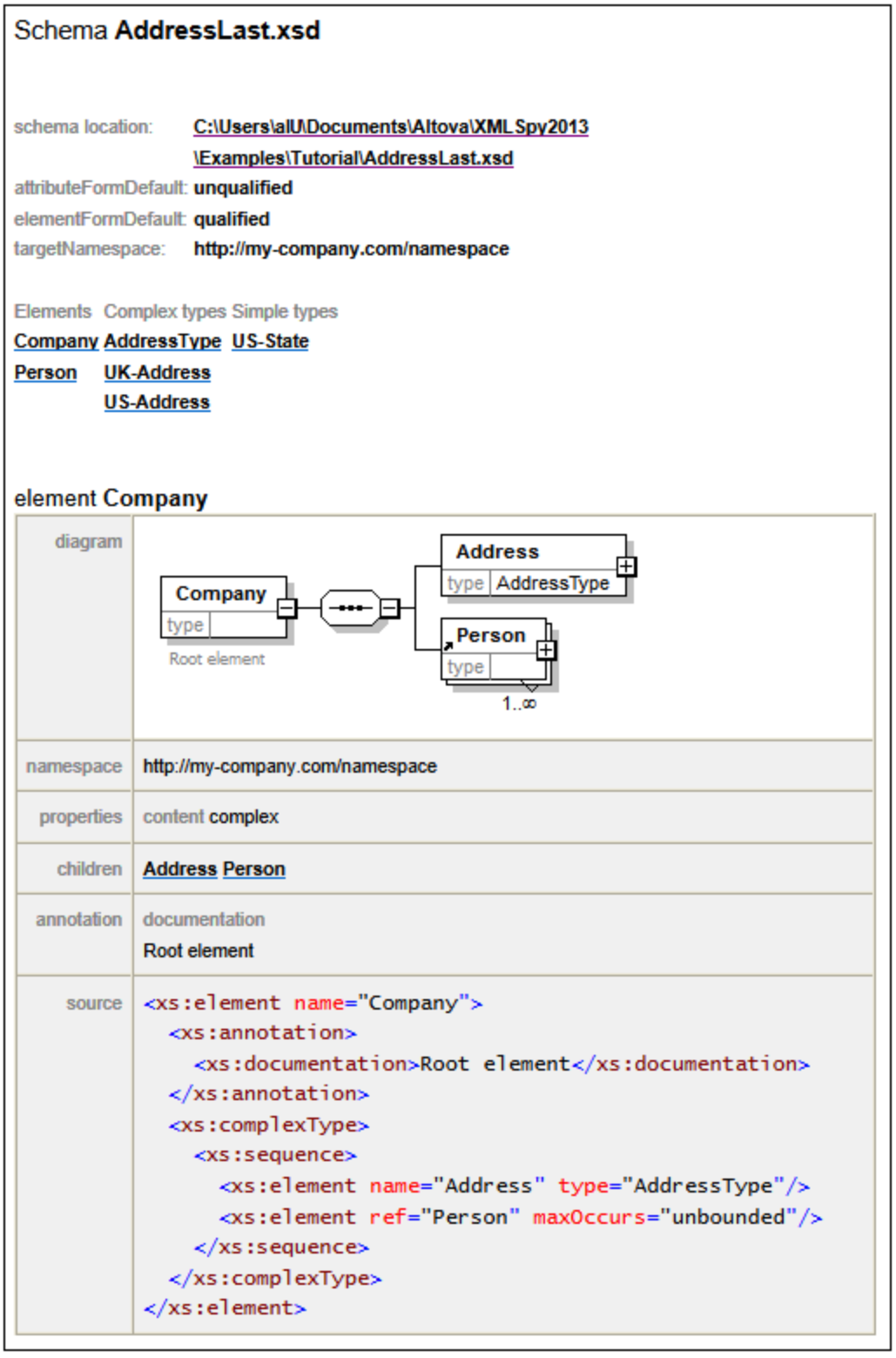
Sie werden in diesem Abschnitt Dokumentation für das XML-Schema `AddressLast.xsd` generieren. Gehen Sie dazu folgendermaßen vor:

1. Wählen Sie die Menüoption **Schema-Design | Dokumentation generieren,** um das Dialogfeld "Schema-Dokumentation" aufzurufen.

© 2016-2022 Altova GmbH



2. Wählen Sie das Ausgabeformat HTML und bestätigen Sie mit OK.
3. Wählen Sie im Dialogfeld "Speichern unter" das Verzeichnis und geben Sie einen Namen für die HTML-Datei ein (z.B. `AddressLast.html`). Bestätigen Sie mit **Speichern**. Das HTML-Dokument wird in der Browser-Ansicht von XMLSpy angezeigt. Wenn Sie auf einen Link klicken, gelangen Sie zur damit verknüpften Komponente.



Schema **AddressLast.xsd**

schema location: C:\Users\alU\Documents\Altova\XMLSpy2013\Examples\Tutorial\AddressLast.xsd
attributeFormDefault: **unqualified**
elementFormDefault: **qualified**
targetNamespace: **http://my-company.com/namespace**

Elements Complex types Simple types
Company **AddressType** **US-State**
Person **UK-Address**
US-Address

element **Company**

diagram

namespace	http://my-company.com/namespace
properties	content complex
children	**Address** **Person**
annotation	documentation Root element

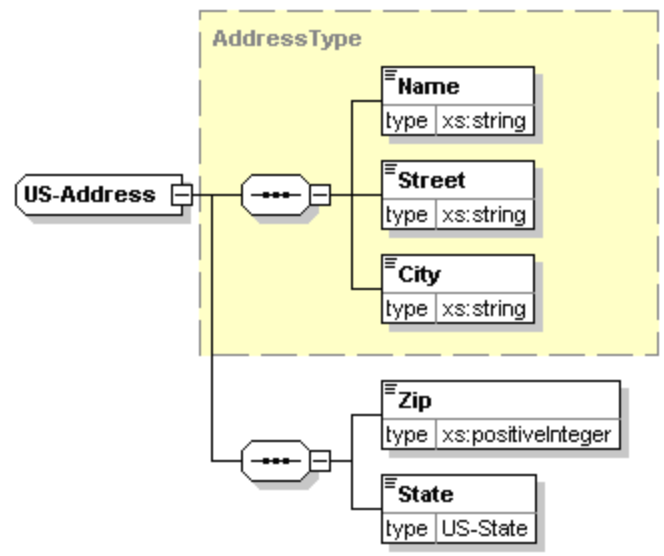
source

```
<xs:element name="Company">
  <xs:annotation>
    <xs:documentation>Root element</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Address" type="AddressType"/>
      <xs:element ref="Person" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

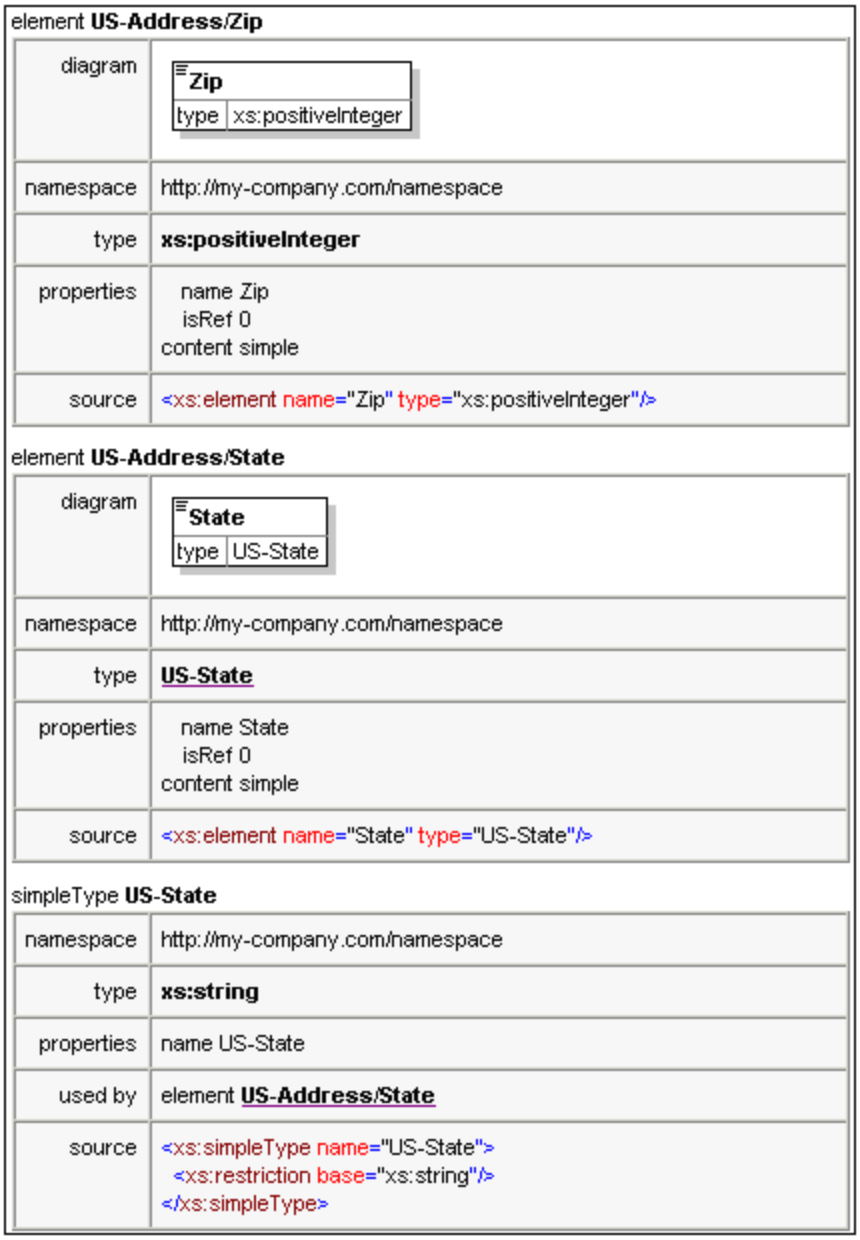
Das oben abgebildete Diagramm stellt die **erste Seite** der Schema-Dokumentation im HTML-Format dar. Falls Komponenten anderer Schemas miteinbezogen wurden, werden diese ebenfalls dokumentiert.

© 2016-2022 Altova GmbH

complexType **US-Address**

diagram	
namespace	http://my-company.com/namespace
type	extension of **AddressType**
properties	name US-Address base AddressType
children	**Name Street City Zip State**
source	<xs:complexType name="US-Address"> <xs:complexContent> <xs:extension base="AddressType"> <xs:sequence> <xs:element name="Zip" type="xs:positiveInteger"/> <xs:element name="State" type="US-State"/> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType>

Das oben abgebildete Diagramm stellt dar, wie ComplexTypes dokumentiert werden.



element **US-Address/Zip**

diagram	Zip type xs:positiveInteger
namespace	http://my-company.com/namespace
type	**xs:positiveInteger**
properties	name Zip isRef 0 content simple
source	<xs:element name="Zip" type="xs:positiveInteger"/>

element **US-Address/State**

diagram	State type US-State
namespace	http://my-company.com/namespace
type	**US-State**
properties	name State isRef 0 content simple
source	<xs:element name="State" type="US-State"/>

simpleType **US-State**

namespace	http://my-company.com/namespace
type	**xs:string**
properties	name US-State
used by	element **US-Address/State**
source	<xs:simpleType name="US-State"> <xs:restriction base="xs:string"/> </xs:simpleType>

Das oben abgebildete Diagramm zeigt an, wie Elemente und SimpleTypes dokumentiert werden.

Sie können nun die MS Word-Ausgabeoption ausprobieren. Das Word-Dokument wird in MS Word geöffnet. Um Hyperlinks im MS Word-Dokument zu verwenden, halten Sie die **Strg**-Taste gedrückt, während Sie auf den Link klicken.

© 2016-2022 Altova GmbH

2.5 XML Dokumente

In diesem Abschnitt lernen Sie:
Erstellen eines neuen XML Dokumentes in XMLSpy, Verwenden von verschiedenen Ansichten und Eingabehilfen für erleichtertes Bearbeiten und Validieren von Daten in XMLSpy.

Ziel

In diesem Abschnitt werden folgende Punkte behandelt:

- Erstellen eines neuen XML-Dokumentes basierend auf dem Schema `AddressLast.xsd`.
- Definition des Elementtyps, um für das Element bei der Validierung ein erweitertes Content Model für dieses Element bereitzustellen.
- Einfügen von Elementen und Attributen und Eingabe von Inhalt für diese in der Grid- und der Text-Ansicht mit Hilfe der intelligenten Eingabehilfen.
- Kopieren von XML-Daten von XMLSpy in ein MS Excel-Blatt, Einfügen von Daten in MS Excel und Zurückkopieren der Daten nach XMLSpy. Diese Funktion steht in der Tabellenansicht der Grid-Ansicht zur Verfügung.
- Sortieren von XML-Elementen mit Hilfe der Sortierfunktion der Tabellenansicht.
- Validieren des XML-Dokumentes.
- Ändern des Schemas, so dass dreistellige Telefondurchwahlnummern eingegeben werden können.

In diesem Abschnitt verwendete Befehle

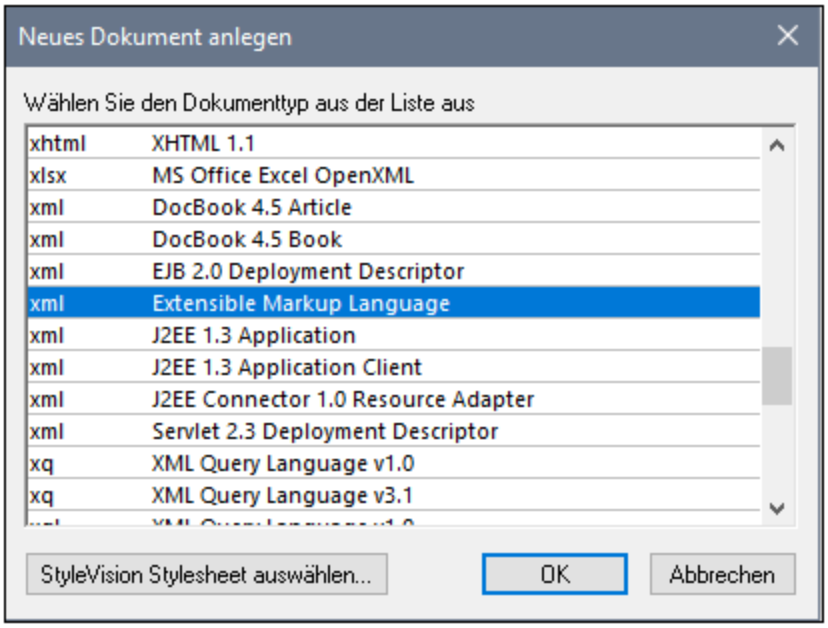
	**Datei | Neu** erstellt eine neue XML-Datei.
	**Ansicht | Text** wechselt in die Text-Ansicht.
	**Ansicht | Grid-Ansicht** wechselt in die Grid-Ansicht.
	**XML | Tabelle | Als Tabelle anzeigen** zeigt Mehrfachinstanzen eines einzelnen Elementtyps auf einer einzelnen hierarchischen Ebene als Tabelle an. Diese Ansicht des Elements nennt sich Tabellenansicht (oder einfach Tabellenansicht). Das Symbol dient zum Wechseln zwischen der Tabellen,- und der normalen Grid-Ansicht.
	**F7** Überprüft, ob das Dokument wohlgeformt ist.
	**F8** Validiert das XML-Dokument anhand des damit verknüpften Schemas bzw. der DTD.
	Öffnet die dazugehörige DTD oder XML-Schema-Datei.

2.5.1 Erstellen eines neuen XML-Dokuments

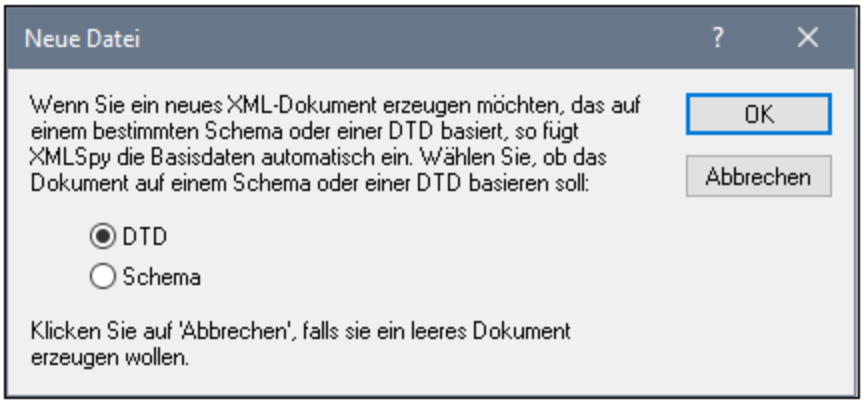
Beim Erstellen einer neuen XML-Datei in XMLSpy haben Sie die Option, dieser ein Schema (DTD oder XML-Schema) zugrunde zu legen oder auch nicht. In diesem Abschnitt werden wir eine neue Datei erstellen, die auf dem zuvor im Rahmen dieses Tutorials erzeugten Schema `AddressLast.xsd` basiert.

So erstellen Sie eine neue XML-Datei:

1. Wählen Sie die Menüoption **Datei | Neu**, um Dialogfeld "Neues Dokument anlegen" aufzurufen.



2. Wählen Sie im Dialogfeld den Eintrag `Extensible Markup Language` (oder einen generischen XML-Dokumenteintrag) und bestätigen Sie mit **OK**. Im nächsten Fenster werden Sie gefragt, ob das XML-Dokument auf einer DTD oder einem Schema basieren soll.

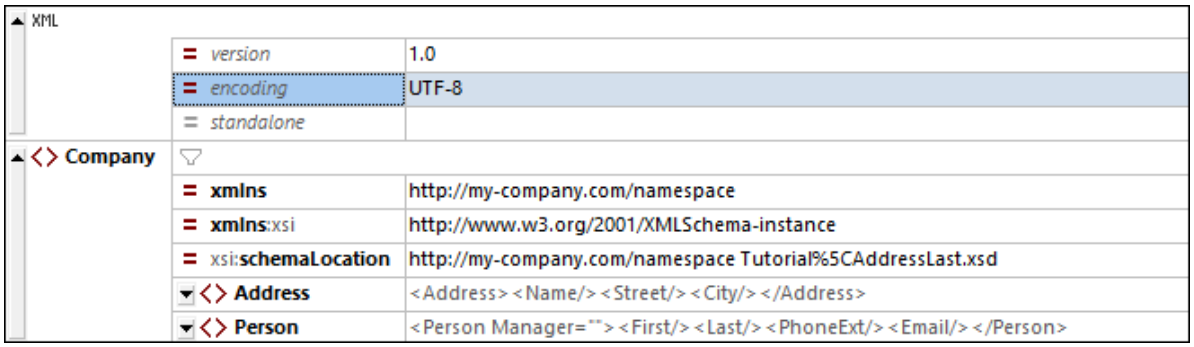


3. Wählen Sie das Optionsfeld "Schema" und bestätigen Sie mit OK. Sie werden nun aufgefordert, die Schema-Datei auszuwählen, auf der Ihr XML-Dokument basieren soll.
4. Wählen Sie "Durchsuchen" oder "Fenster", um die Schema-Datei zu suchen. Bei Klicken auf die Schaltfläche "Fenster" werden alle Dateien, die in XMLSpy und Projekten geöffnet sind, aufgelistet.

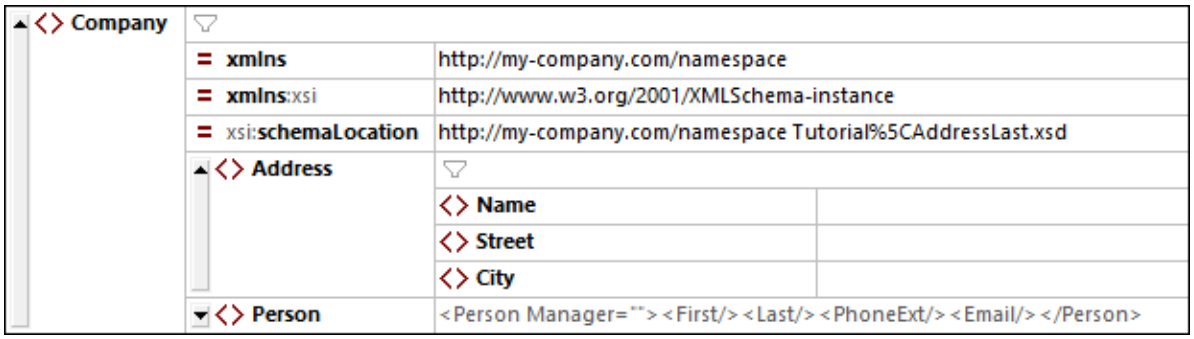
© 2016-2022 Altova GmbH

Wählen Sie `AddressLast.xsd` (Pfad siehe Tutorial Einführung[36]) und bestätigen Sie Ihre Auswahl mit OK. Im Hauptfenster wird ein neues XML-Dokument geöffnet. Das Dokument enthält alle Hauptelemente, die im Schema definiert wurden.

5. Klicken Sie auf das Register **Grid**, um die Grid-Ansicht auszuwählen.
6. Beachten Sie in der Grid-Ansicht die Struktur des Dokuments. Wenn Sie auf ein Element klicken, wird die Auswahl auf dieses Element reduziert. Ihr Dokument sollte etwa folgendermaßen aussehen:



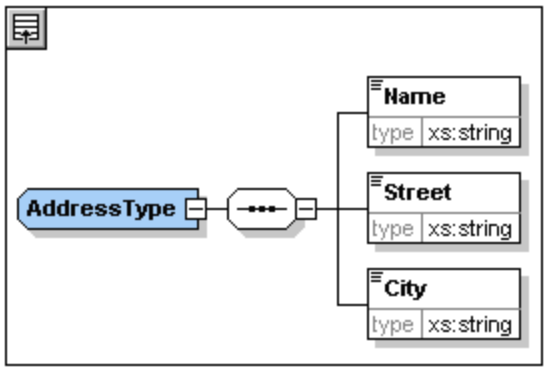
7. Klicken Sie auf das Symbol neben Address, um die Child-Elemente von Address anzuzeigen. Ihr Dokument sollte aussehen wie dieses:



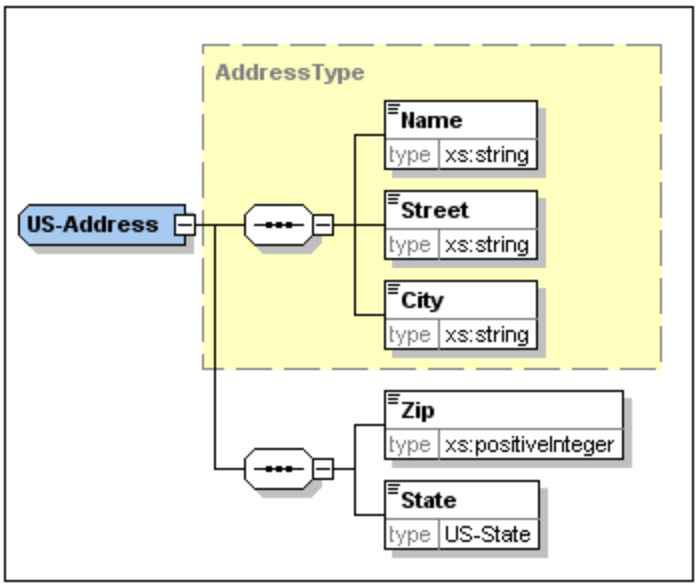
8. Wählen Sie die Menüoption **Datei | Speichern** und speichern Sie sie im Ordner **`Tutorial`**. Geben Sie Ihrem XML-Dokument einen passenden Namen (z.B. **`CompanyFirst.xml`**). Beachten Sie, dass der Ordner **`Tutorial`** die fertige Tutorial-Datei **`CompanyFirst.xml`** enthält, daher sollten Sie diese Datei eventuell umbenennen, bevor Sie der soeben erstellten Datei diesen Namen geben.

2.5.2 Festlegen des Typs eines Elements

Die in der Grid-Ansicht angezeigten Child-Elemente von **`Address`** sind die, welche für den globalen ComplexType **`AddressType`** (*das Content Model dazu ist im XML-Schema* **`AddressLast.xsd`** *definiert, das Sie in der Abbildung unten sehen*) definiert wurden.



Wir möchten jedoch anstelle des generischen Adresstyps einen spezifischen US- oder UK-Adresstyp verwenden. Sie werden sich erinnern, dass wir im Schema **AddressLast.xsd** globale ComplexTypes für **US-Address** und **UK-Address** erstellt haben, indem wir den ComplexType **AddressType** erweitert haben. Unten sehen Sie das Content Model von **US-Address**.

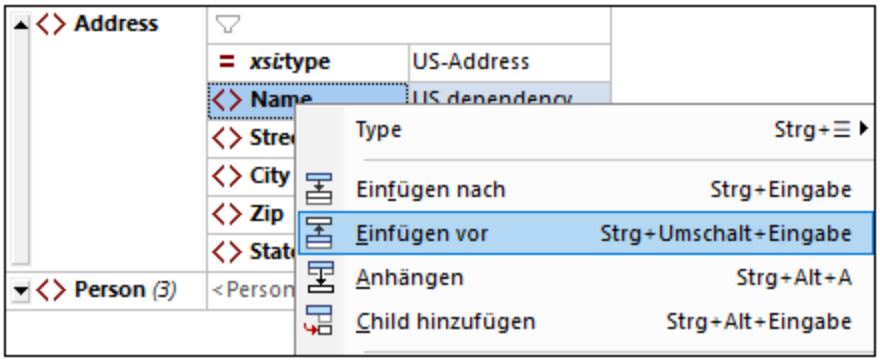


Um festzulegen, dass das Element **Address** im XML-Dokument einem der erweiterten **Address**-Typen entsprechen muss (**US-Address** oder **UK-Address**) und nicht dem generischen **AddressType**, müssen wir den erforderlichen erweiterten ComplexType als ein Attribut des Elements **Address** definieren.

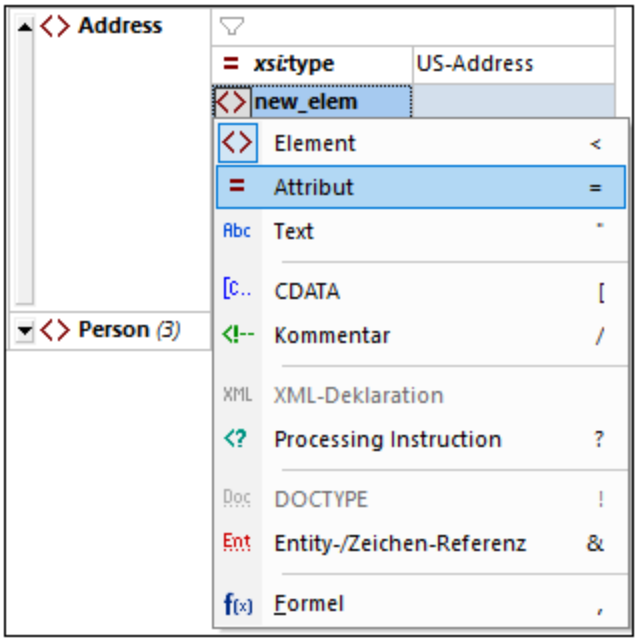
Dieses Attribut des Elements **Address** wird folgendermaßen hinzugefügt:

1. Klicken Sie im XML-Dokument mit der rechten Maustaste auf das Element `Name` und wählen Sie im Kontextmenü **Einfügen vor** (*siehe Abbildung unten*).

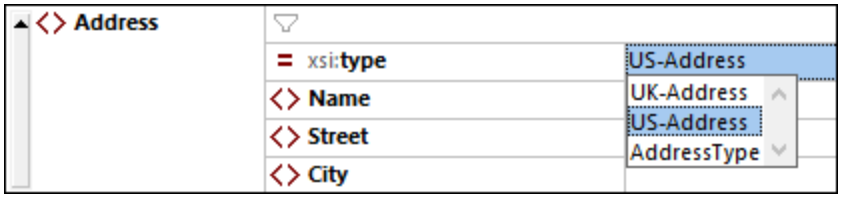
© 2016-2022 Altova GmbH



2. Daraufhin wird oberhalb des Elements **`Name`** ein neuer Element-Node namens **`new_elem`** hinzugefügt (*siehe Abbildung unten*). Klicken Sie links vom Namen des Node auf den Elementtyp und wählen Sie im daraufhin angezeigten Menü (*Abbildung unten*) den Node-Typ *Attribut* aus. Daraufhin ändert sich der Node-Typ in den Node-Typ *Attribut*; der Name bleibt jedoch weiterhin **`new_elem`**.



3. Doppelklicken Sie auf den Node-Namen und wählen Sie in der daraufhin angezeigten Eingabehilfe **`xsi:type`**.
4. Drücken Sie die **Tabulatortaste**, um zum Wertefeld des Attributs zu gelangen. Daraufhin wird ein Popup-Menü mit den verfügbaren **`xsi:type`**-Werten angezeigt (*siehe Abbildung unten*). Diese Werte sind die für das Element **`Address`** im Schema definierten complexTypes.



5. Wählen Sie als Wert des Attributs **`xsi:type US-Address`** aus.

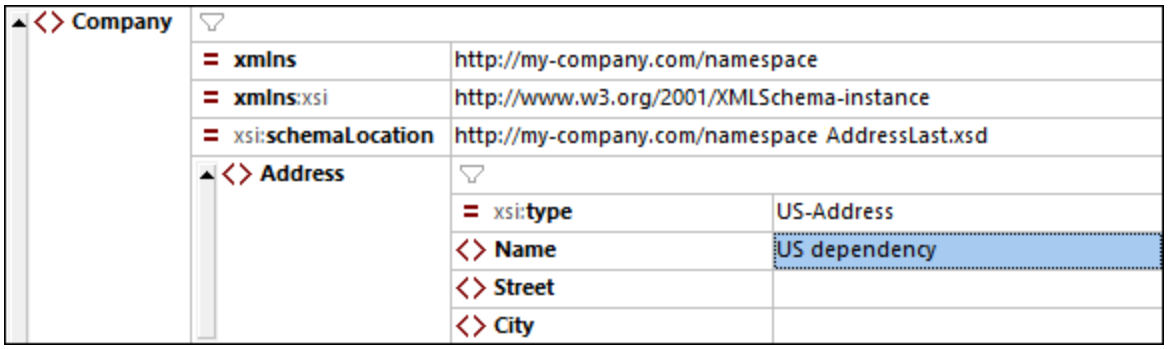
Anmerkung: Das vorgesetzte **xsi:** ermöglicht die Verwendung von speziellen XML-Schema-spezifischen Befehlen in Ihrer XML-Dokumentinstanz. Beachten Sie, dass der Namespace für das Präfix **xsi.** automatisch zum Dokumentelement hinzugefügt wurde, als Sie Ihrer XML-Datei ein Schema zugewiesen haben. Sie haben soeben einen Typ für das Element **Address** festgelegt. Nähere Informationen dazu finden Sie in der XML Schema-Spezifikation.

2.5.3 Eingabe von Daten in der Grid-Ansicht

Sie können jetzt Daten in Ihr XML-Dokument eingeben.

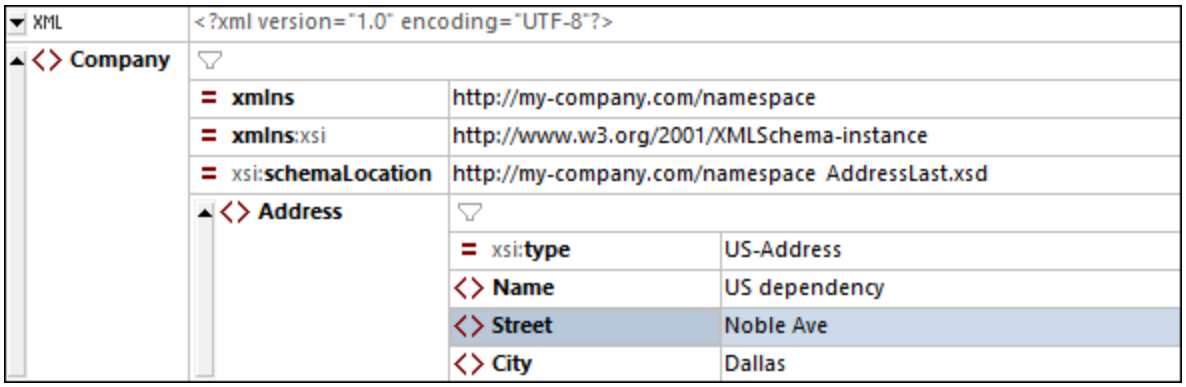
Gehen Sie folgendermaßen vor:

1. Doppelklicken Sie in das Wertefeld `Name` (oder verwenden Sie die Pfeiltasten) und geben Sie `US dependency` ein. Bestätigen Sie mit der **Eingabetaste**.



2. Geben Sie auf dieselbe Art einen `Street` und `City` Namen ein (z.B. `Noble Ave.` und `Dallas`).

3. Klicken Sie auf das Element `Person` und drücken Sie die **Entf**-Taste, um es zu löschen. (Wir werden es im nächsten Abschnitt des Tutorials wieder hinzufügen). Daraufhin erscheint nun das gesamte Element `Address` markiert.
4. Klicken Sie auf ein beliebiges Child-Element des Elements `Address`, um die Auswahl für alle Child-Elemente von `Address` mit Ausnahme des gewählten Elements aufzuheben. Ihr XML-Dokument sollte folgendermaßen aussehen:



 © 2016-2022 Altova GmbH

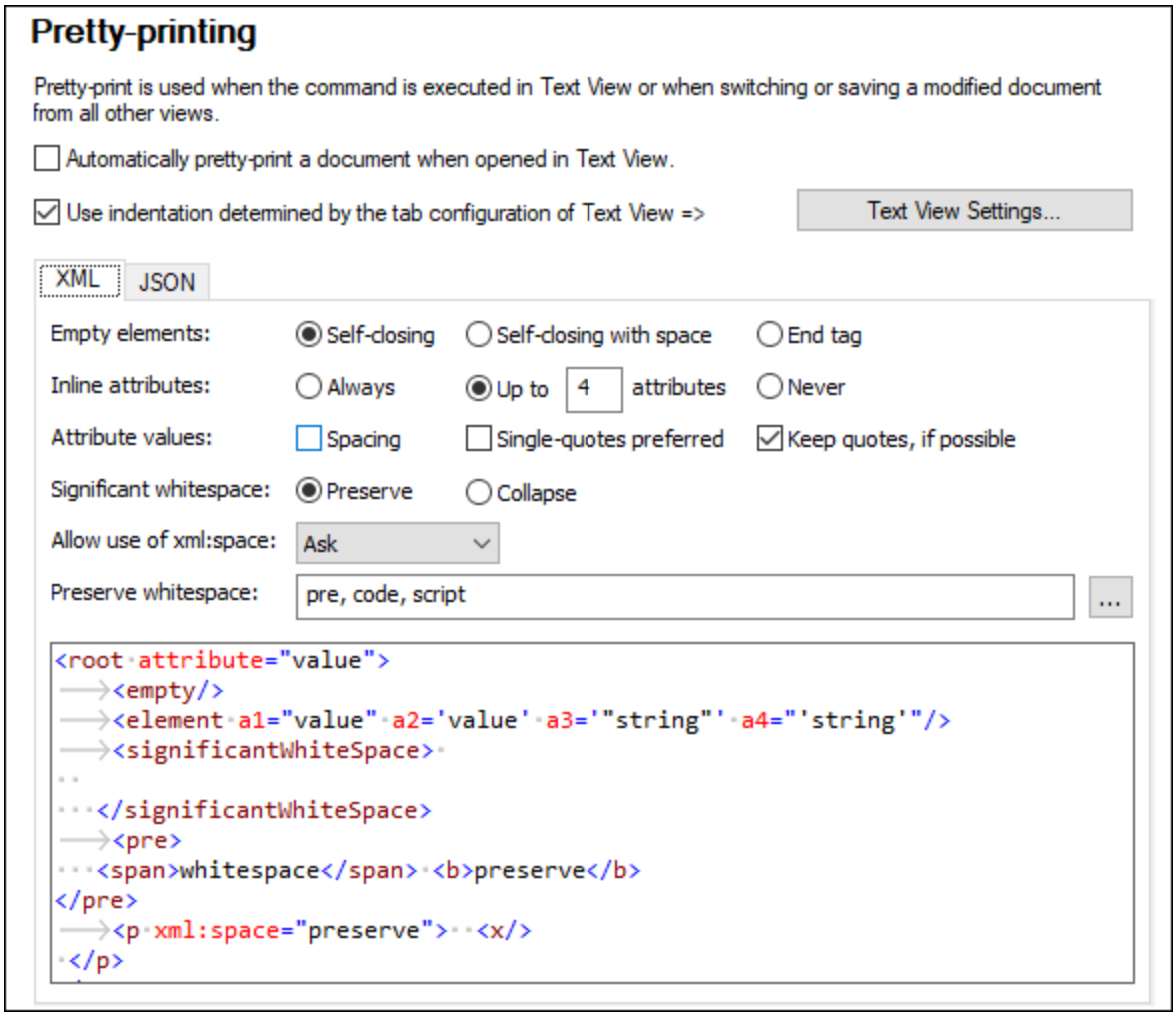
2.5.4 Eingabe von Daten in der Textansicht

In der Textansicht werden die Daten und der Markup-Code der XML-Dateien in einem übersichtlich strukturierten Layout angezeigt, und gemäß dem jeweiligen Schema werden intelligente Bearbeitungsfunktionen bereitgestellt.

Dokument-Layout

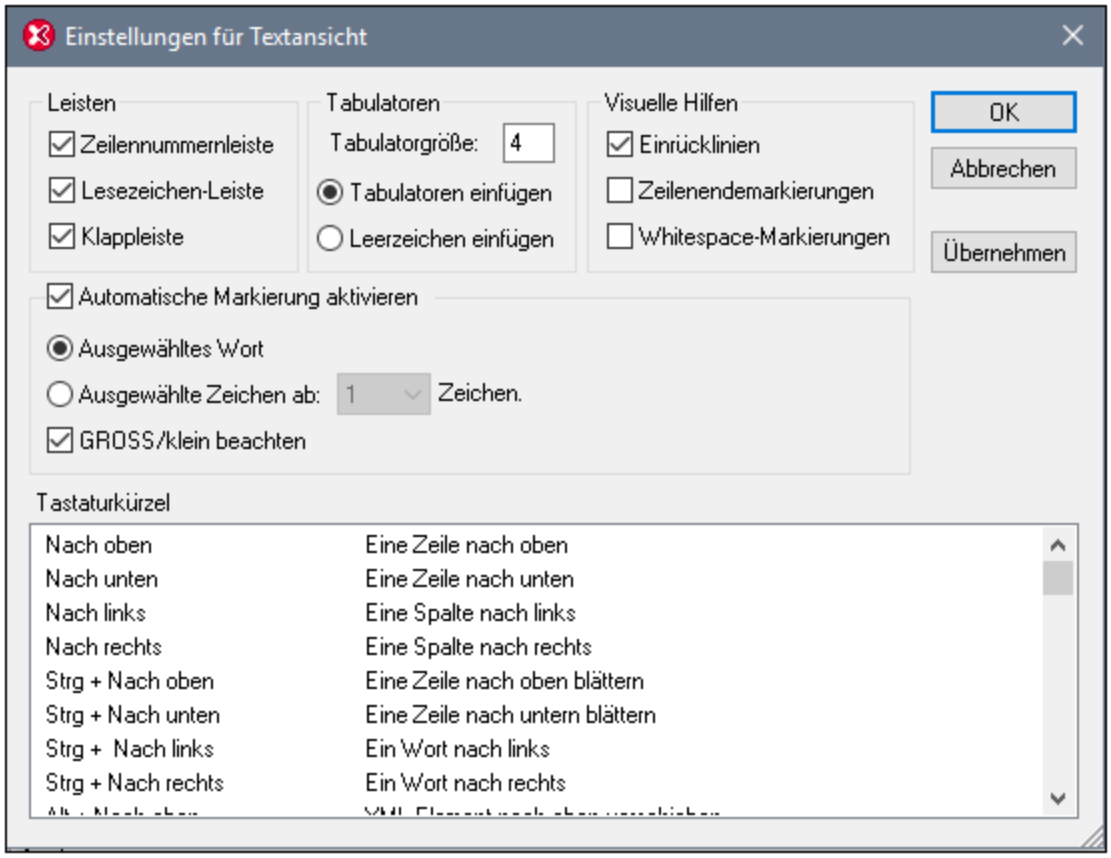
Das Dokument-Layout der Textansicht wird an zwei Stellen definiert:

Pretty-Print-Optionen
Diese Einstellungen befinden sich im Abschnitt Pretty-Print des Dialogfelds "Optionen" (*Abbildung unten*). Wenn Sie eine Option definieren, sehen Sie die Auswirkung davon sofort unten im Vorschaubereich. Konfigurieren Sie die Pretty-Print-Optionen nach Bedarf. Während Sie das Dokument in der Textansicht bearbeiten, kann es vorkommen, dass das Layout des Dokuments unstrukturiert erscheint, vor allem, wenn Sie Textblöcke kopiert haben. Um ein sauberes und hierarchisches Layout zu erhalten, klicken Sie einfach auf **Bearbeiten | Pretty-Print**.

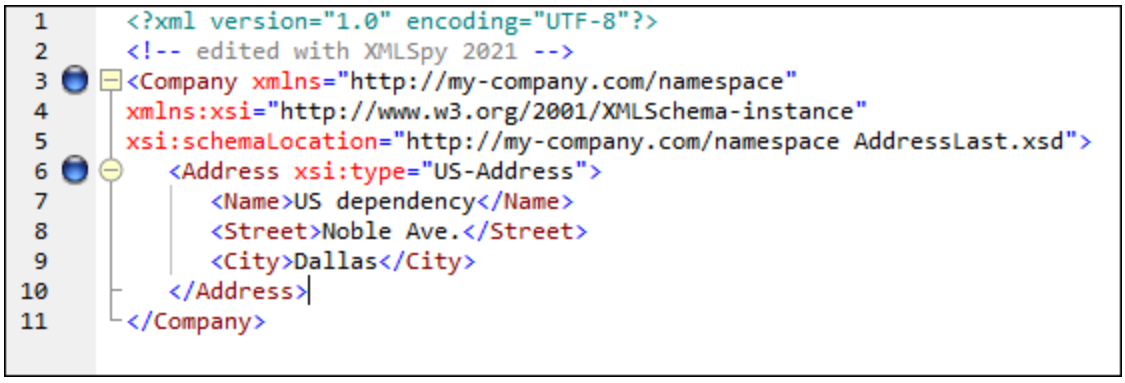


Einstellungen für Textansicht

Das Dialogfeld "Einstellungen für Textansicht (*Abbildung unten*) bietet nicht nur zusätzliche Layout-Optionen, sondern dient auch zum Ein-/Ausschalten nützlicher Funktionalitäten der Textansicht, wie z.B. Zeilennummerierung und Klappleisten. Das Dialogfeld "Einstellungen für Textansicht" wird mit dem Befehl **Ansicht | Einstellungen für Textansicht** aufgerufen.



In der Abbildung unten sehen Sie die aktuelle XML-Datei in der Textansicht. Die einzelnen Elemente der Textansicht sind gemäß den Einstellungen im Dialogfeld oben eingeblendet.



Auf der linke Seite befinden sich drei Leisten: (i) die Zeilennummernleiste, (ii) die Lesezeichenleiste (die zwei blaue Lesezeichen enthält) und (iii) die Klappleiste (über die Sie XML-Elemente ein- und ausklappen können).

© 2016-2022 Altova GmbH

Einrücklinien sind die hellgrauen senkrechten Linien, die die Einrückung von Tags auf derselben hierarchischen Ebene anzeigen. Zusätzliche visuelle Hilfen sind Zeilenendemarkierungen und Whitespace-Markierungen, die im Bereich *Visuelle Hilfen* (*siehe Abbildung oben*) aktiviert oder deaktiviert werden können.

Anmerkung: Die Textansicht-spezifischen Funktionen "Pretty-Print" und "Lesezeichen" wurden bereits weiter oben im Abschnitt Einstellungen für Textansicht 44 dieses Tutorials behandelt.

Bearbeitung in der Textansicht

In diesem Abschnitt werden wir Daten in der Textansicht eingeben und bearbeiten, um die Funktionen der Textansicht zu erläutern.

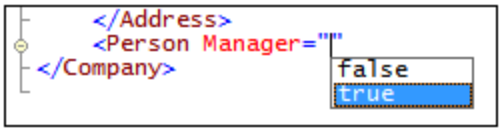
Anmerkung: Da die Funktion *Bei Bearbeitung validieren* standardmäßig aktiv ist, werden alle bei der Bearbeitung entstandenen Validierungsfehler sofort gekennzeichnet. Die Fehlermeldung(en) dazu werden im Fenster "Meldungen" angezeigt. Ignorieren Sie diese Fehler und Meldungen für den Moment. Wenn Sie die Validierung um Hintergrund stört, können Sie die Funktion *Bei Bearbeitung validieren* in den Validierungseinstellungen des Dialogfelds "Optionen" 1603 abschalten. Beachten Sie, dass Sie Ihr Dokument jederzeit validieren können (Beschreibung im nächsten Abschnitt 93 dieses Tutorials).

Gehen Sie folgendermaßen vor:

1. Wählen Sie die Menüoption **Ansicht | Text** oder klicken Sie auf das Register **Text**. Sie sehen nun das XML-Dokument in seiner Textform mit der Syntaxfärbung.
2. Setzen Sie den Cursor hinter den End-Tag des Elements **`Address`** und drücken Sie die **Eingabetaste**, um eine neue Zeile einzufügen.
3. Geben Sie ein "Kleiner als..." Zeichen (**<**) an dieser Stelle ein. Eine Auswahlliste aller an dieser Stelle (laut Schema) zulässigen Elemente erscheint. Da an dieser Stelle nur das Element **`Person`** zulässig ist, wird nur dieses eine Element in der Liste angezeigt.

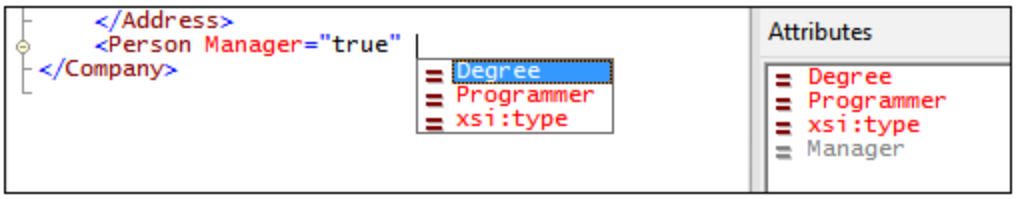


4. Wählen Sie den Eintrag **`Person`** aus. Das Element **`Person`** wird mit seinem Attribut **`Manager`** eingefügt, wobei sich der Cursor innerhalb des Wertefelds des Attributs **`Manager`** befindet.
5. Wählen Sie in der Dropdown-Liste des Attributs **`Manager`** den Eintrag **`true`** aus.

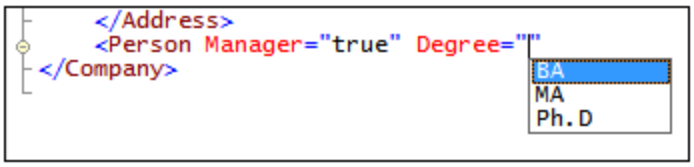


6. Setzen Sie den Cursor an das Ende der Zeile (oder betätigen Sie die "**Ende**"-Taste Ihrer Tastatur). Drücken Sie die Leerzeichentaste einmal. Daraufhin wird eine Dropdown-Liste geöffnet, die eine Liste

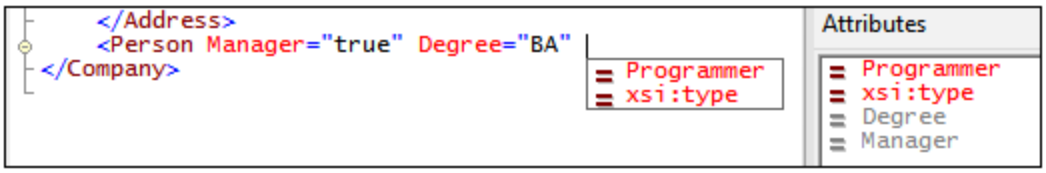
von an dieser Stelle zulässigen Attributen enthält. In der Attribut-Eingabehilfe sind verfügbare Attribute rot markiert. Das Attribut **Manager** ist ausgegraut, da es bereits verwendet wurde.



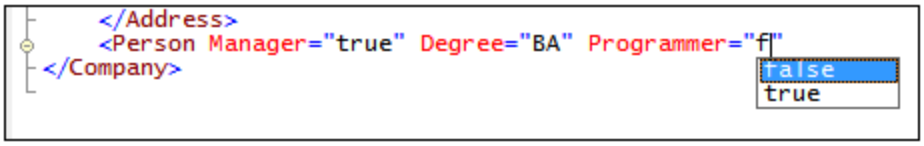
7. Wählen Sie **Degree** durch Betätigen des 'Nach unten'-Pfeiles und drücken Sie die **Eingabetaste**. Eine weitere Liste wird geöffnet. Hier können Sie eine der vordefinierten Enumerationen auswählen (**BA**, **MA** oder **PhD**). (Enumerationen sind Werte, die gemäß einem XML-Schema zulässig sind).



8. Wählen Sie **BA** durch Betätigen des 'Nach unten'-Pfeiles und bestätigen Sie die Auswahl mit der **Eingabetaste**. Setzen Sie den Cursor anschließend an das Ende der Zeile (über die **Ende**-Taste Ihrer Tastatur). Drücken Sie die Leerzeichentaste einmal. Nun sind **Manager** und **Degree** in der Attribut-Eingabehilfe ausgegraut.



9. Wählen Sie **Programmer** durch Betätigen des 'Nach unten'-Pfeiles und drücken Sie die **Eingabetaste**.



10. Geben Sie den Buchstaben **"f"** ein und drücken Sie die **Eingabetaste**.
11. Setzen Sie den Cursor (mit der **Ende**-Taste) ans Ende der Zeile, und geben Sie das "größer als..." Zeichen (>) ein. XMLSpy fügt automatisch alle erforderlichen Child-Elemente von **Person** ein. (Beachten Sie, dass das optionale Element **Title** nicht eingefügt wird). Jedes Element hat einen Start- und einen End-Tag aber keinen Inhalt.

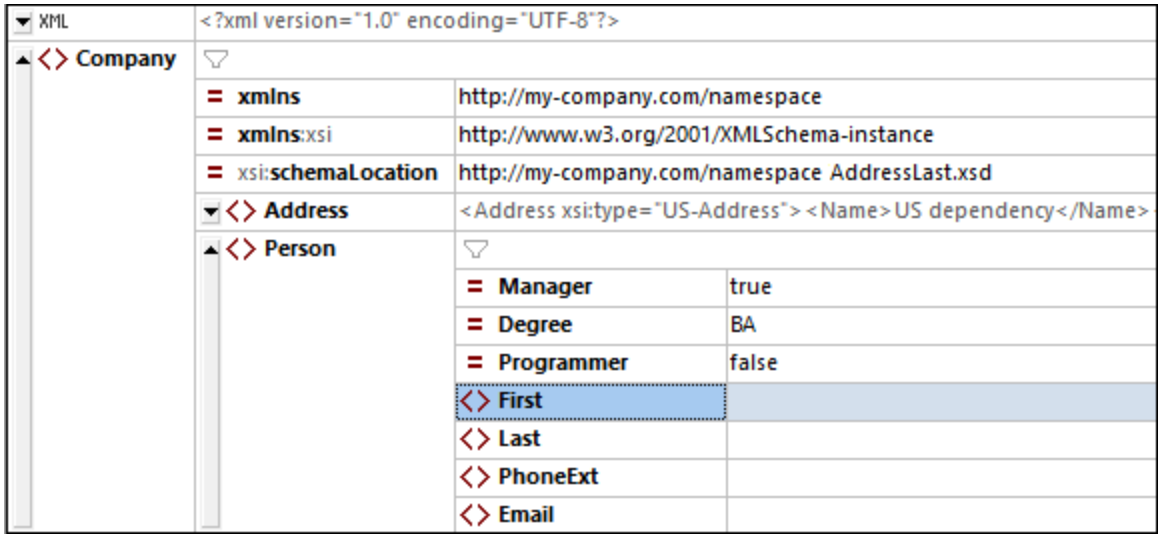
© 2016-2022 Altova GmbH

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy 2021 -->
<Company xmlns="http://my-company.com/namespace" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://my-company.com/namespace AddressLast.xsd">
   <Address xsi:type="US-Address">
      <Name>US dependency</Name>
      <Street>Noble Ave.</Street>
      <City>Dallas</City>
   </Address>
   <Person Manager="true" Degree="BA" Programmer="false">
      <First></First>
      <Last></Last>
      <PhoneExt></PhoneExt>
      <Email></Email>
   </Person>
</Company>
```

Sie könnten nun die **Personendaten** hier eingeben, doch wollen wir nun in die Grid-Ansicht wechseln, damit Sie sehen, wie einfach sich Ansichten beim Bearbeiten eines Dokuments wechseln lassen.

Wechseln zur Grid-Ansicht

Um zur Grid-Ansicht zu wechseln, wählen Sie die Menüoption **Ansicht | Grid-Ansicht** oder klicken Sie auf das Register **Grid**. Sehen Sie selbst, wie die neu hinzugefügten Child-Nodes von `Person` angezeigt werden.



Wir wollen das Dokument nun validieren und etwaige Fehler korrigieren.

2.5.5 Validierung des Dokuments

XMLSpy bietet zwei wichtige Funktionen zur Überprüfung des XML-Dokuments:

- eine Wohlgeformtheitsprüfung
- eine Validierung

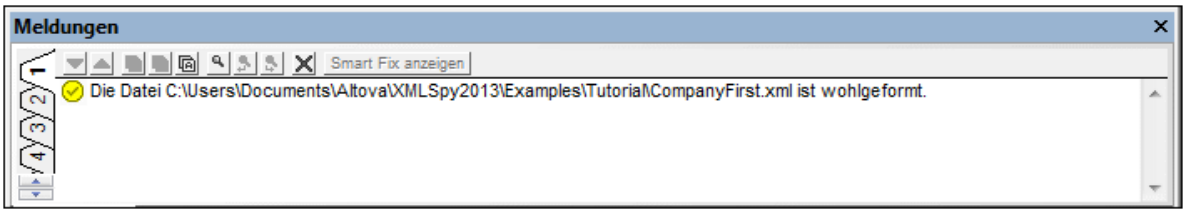
	Wohlgeformtheit prüfen
	XML validieren

Da die Funktion *Bei Bearbeitung validieren* standardmäßig aktiv ist, werden alle bei der Bearbeitung entstandenen Validierungsfehler sofort gekennzeichnet. Die Fehlermeldung(en) dazu werden im Fenster "Meldungen" angezeigt. Wenn Sie die Validierung um Hintergrund stört, können Sie die Funktion *Bei Bearbeitung validieren* in den Validierungseinstellungen des Dialogfelds "Optionen" 1603 abschalten. Falls Sie dies tun, beachten Sie, dass Sie Wohlgeformtheitsprüfungen und Validierungen jederzeit durch Aufruf des entsprechenden Befehls über das Menü "XML" durchführen können. In diesem Abschnitt des Tutorials wird beschrieben, wie Sie diese Überprüfungen durchführen.

Überprüfung der Wohlgeformtheit

Ein XML-Dokument ist wohlgeformt, wenn es zu jedem Start-Tag einen End-Tag gibt, Elemente korrekt verschachtelt sind und keine Zeichen fehlen oder an der falschen Stelle stehen (z.B. eine Entity ohne das Semikolon). Sie können in jeder Bearbeitungsansicht eine Wohlgeformtheitsprüfung durchführen. Überprüfen Sie Ihr Dokument folgendermaßen:

1. Wählen Sie die Textansicht.
2. Wählen Sie die Menüoption **XML | Wohlgeformtheit prüfen** oder drücken Sie die Taste **F7**. (Alternativ dazu können Sie in der Symbolleiste auf die entsprechende Schaltfläche dafür klicken). Am unteren Rand des Hauptfensters erhalten Sie im Fenster "Meldungen" eine Meldung, welche anzeigt, ob das Dokument wohlgeformt ist.



Beachten Sie, dass die Ausgabe des Fensters Meldungen" über neun Register verfügt. Das Ergebnis der Aktion wird immer auf dem aktiven Register angezeigt. So könnten Sie etwa die Wohlgeformtheit auf dem Register 1 prüfen und zum Register 2 wechseln, um eine Valdierung durchzuführen. Wenn Sie das Register nicht wechseln, wird das vorherige Ergebnis auf dem aktiven Register durch das neue Ergebnis überschrieben.

Anmerkung: Bei der Wohlgeformtheitsprüfung wird nicht geprüft, ob das XML-Dokument schemakonform ist. Dies wird bei der Validierung der Datei überprüft.

Überprüfung der Gültigkeit (Validierung)

Ein XML-Dokument ist gemäß einem Schema gültig, wenn mit der im Schema definierten Struktur und dem Inhalt übereinstimmt. Sie können in jeder Bearbeitungsansicht eine Validierung durchführen. Validieren Sie Ihr Dokument folgendermaßen:

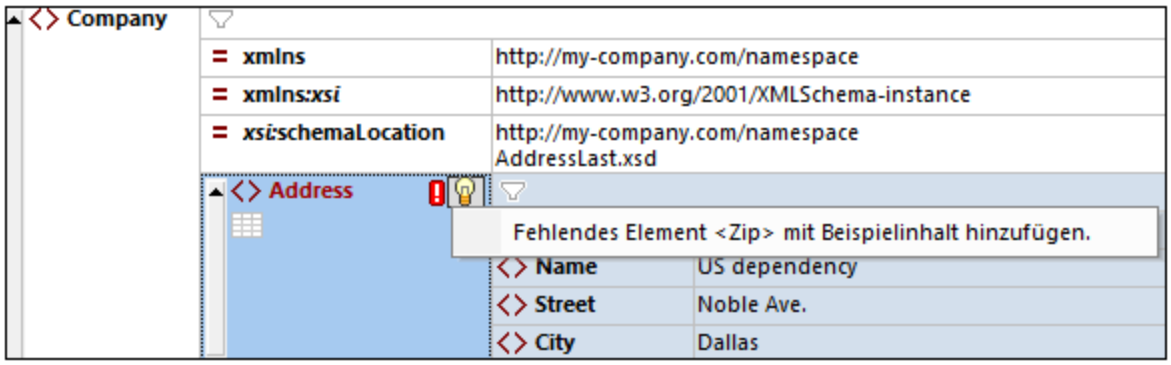
1. Wählen Sie die Grid-Ansicht.
2. Wählen Sie die Menüoption **XML | Validieren** oder drücken Sie die Taste **F8**. (Alternativ dazu können Sie in der Symbolleiste auf die entsprechende Schaltfläche dafür klicken). Im Fenster "Meldungen" wird eine Fehlermeldung angezeigt, dass die Datei nicht gültig ist. Nach dem Element `City` in `Address` werden obligatorische Elemente erwartet. Wenn Sie Ihr Schema überprüfen, werden Sie feststellen,

 © 2016-2022 Altova GmbH

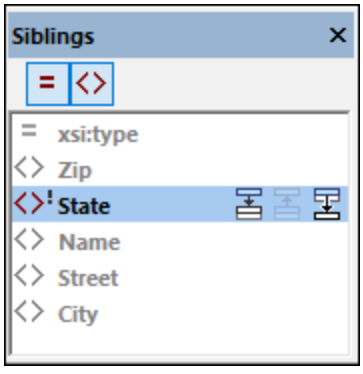
dass der ComplexType **US-Address** (dem aufgrund des **xsi:type**-Attributs dieses **Address**-Element entsprechen muss) ein Content Model hat, in dem auf das Element **City** ein Element **Zip** und ein Element **State** folgen muss.

Beheben von Validierungsfehlern

Die Stelle, an der das Dokument ungültig wird, erscheint rot markiert. Gleichzeitig werden hier ein Fehlersymobl und ein Smart Fix[339]-Symbol angezeigt. Das ungültige Element ist in diesem Fall **Address**. Wenn Sie auf das Smart Fix-Symbol klicken, sehen Sie eine Popup-Meldung mit ungefähr folgendem Inhalt: *Fehlendes Element <Zip> mit Beispielinhalt hinzufügen.* Wenn Sie das Schema überprüfen, wird sich herausstellen, dass auf das Element **Address/City** das obligatorische Element **Zip** folgen muss. Um dies zu überprüfen, wählen Sie das Element **City** aus und sehen Sie sich die Eingabehilfe "Gleichrangige" an. Sie werden feststellen, dass vor dem Element **Zip** ein Ausrufezeichen steht, welches darauf hinweist, dass das Element im aktuellen Kontext obligatorisch ist.

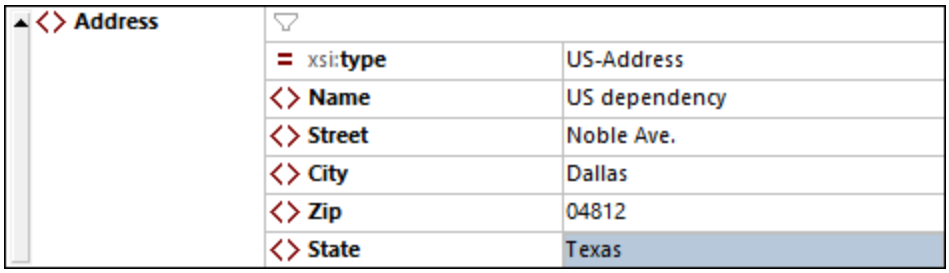


Klicken Sie nun auf Smart Fix (*siehe Abbildung oben*). Daraufhin wird das Element **Zip** mit Beispielinhalt, durch den das Element gültig wird, angezeigt. Geben Sie die korrekte Postleitzahl (**Zip**) ein (z.B. **04812** für Dallas). Sehen Sie sich nun wieder die Eingabehilfe "Gleichrangige" an. Sie sehen darin nun, dass das Element **State** obligatorisch ist (es steht ein Ausrufezeichen davor). Wenn Sie das Element **State** auswählen, werden die Eingabehilfeoptionen, die dafür zur Verfügung stehen, aktiv (*Siehe Abbildung unten*). Dies sind die Aktionen, um das Element **State** nach dem aktuell im Hauptfenster ausgewählten Element (in diesem Fall **City**) einzufügen oder **State** nach allen gleichrangigen Elementen von **City** anzuhängen.



Da beide Aktionen in diesem Fall denselben Effekt haben, wählen Sie eine der beiden Aktionen aus. Daraufhin wird ein **State**-Element nach **City** hinzugefügt. Doppelklicken Sie in das Inhaltsfeld von **State** und geben Sie als Namen des Bundesstaats **Texas** ein (*Abbildung unten*). Beachten Sie, dass die Eingabehilfe

"Gleichrangige" nun nur ausgegraute Elemente enthält. Dies weist darauf hin, dass keine weiteren obligatorischen Elemente vorhanden sind, die hinzugefügt werden müssen.

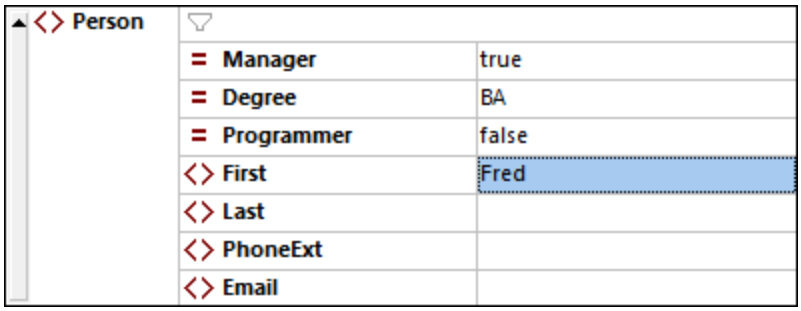


Fertigstellen des Dokuments und erneutes Validieren

Stellen Sie nun das Dokument fertig (geben Sie die Daten für das Element `Person` ein), bevor Sie es erneut validieren.

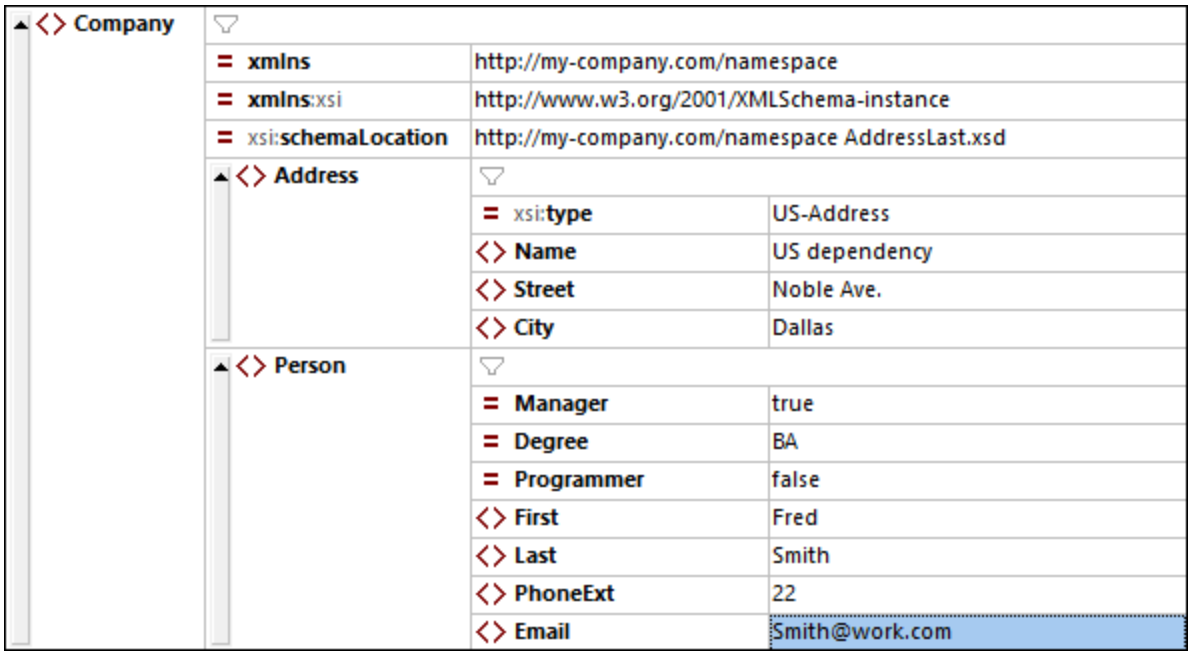
Gehen Sie folgendermaßen vor:

1. Klicken Sie auf das Wertefeld des Elements **`First`** und geben Sie den Vornamen der Person ein - z.B. `Fred`, bestätigen Sie mit der **Eingabetaste**.

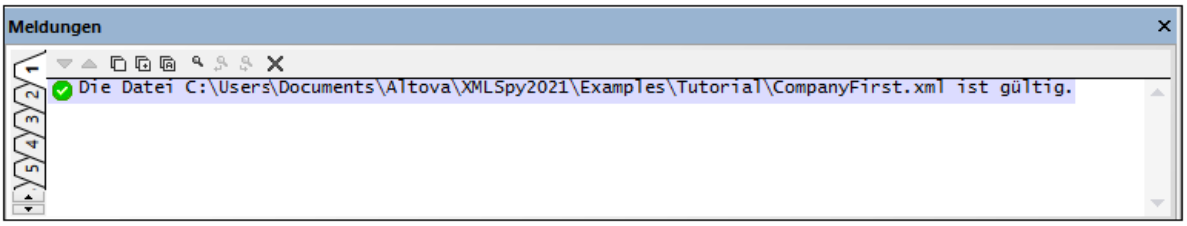


2. Geben Sie nun auf dieselbe Art Daten für alle Child-Elemente von **`Person`**, also **`Last`**, **`PhoneExt`** und **`Email`** ein. Sie können sich mit der Tabulatortaste durch die Zellen bewegen. Beachten Sie, dass der Wert von **`PhoneExt`** eine Ganzzahl mit einem Maximalwert von **`99`** sein muss (da dies der gültige Wertebereich von **`PhoneExt`** ist, den Sie im Schema definiert haben). Ihr XML-Dokument sollte in der Grid-Ansicht nun so aussehen:

© 2016-2022 Altova GmbH



3. Klicken Sie auf die Schaltfläche , um das Dokument neuerlich zu prüfen. Diesmal erhalten Sie im Fenster "Meldungen" die Nachricht ", dass die Datei gültig ist. Das XML-Dokument ist nun gemäß dem Schema gültig.



4. Wählen Sie die Menüoption **Datei | Speichern**.

Anmerkung: Ein XML-Dokument kann auch gespeichert werden wenn es nicht gültig ist. Beim Speichern werden Sie gefragt, ob Sie das nicht korrekte Dokument speichern wollen. Mit **Trotzdem speichern** wird das Dokument in seinem aktuellen, ungültigen Status gespeichert.

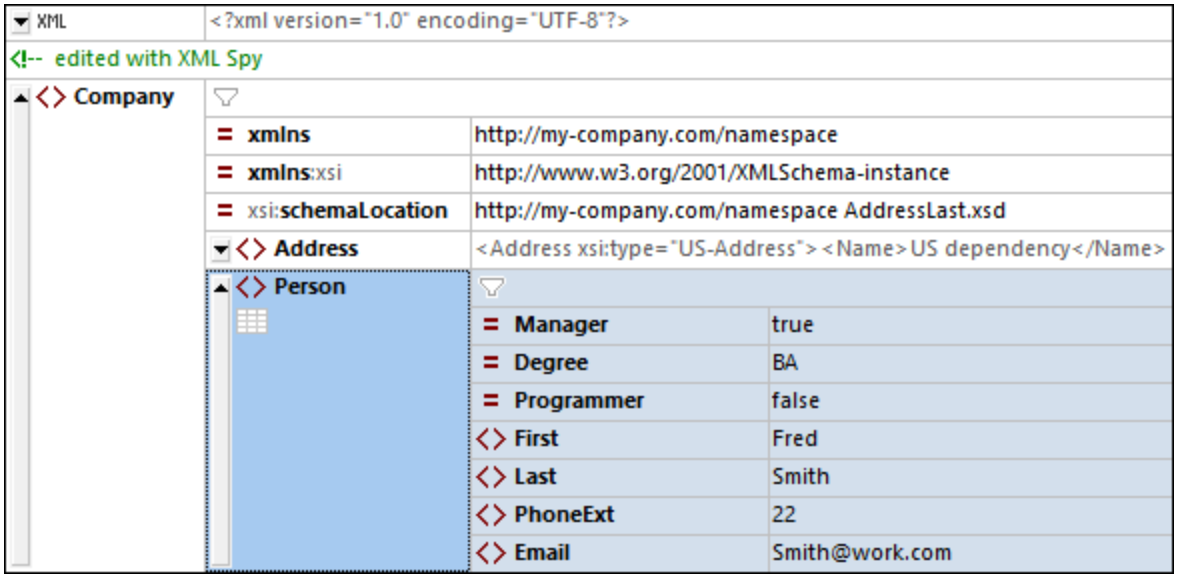
2.5.6 Hinzufügen von Elementen und Attributen

Momentan gibt es im Dokument nur ein `Person`-Element.

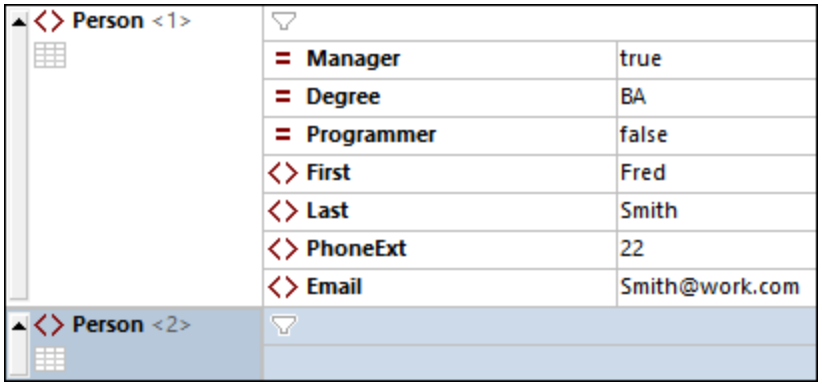
Um ein neues `Person`-Element hinzuzufügen, gehen Sie folgendermaßen vor:

1. Klicken Sie auf die graue Bildlaufleiste links neben dem Element `Address`, um die `Address`-Elemente auszublenden und mehr Platz in der Ansicht zu schaffen.

2. Wählen Sie das gesamte Element **Person** aus, indem Sie in der Grid-Ansicht auf oder unter den Namen des Elements **Person** klicken. Beachten Sie, dass das Element **Person** nun in der Eingabehilfe "Gleichrangige" verfügbar ist.

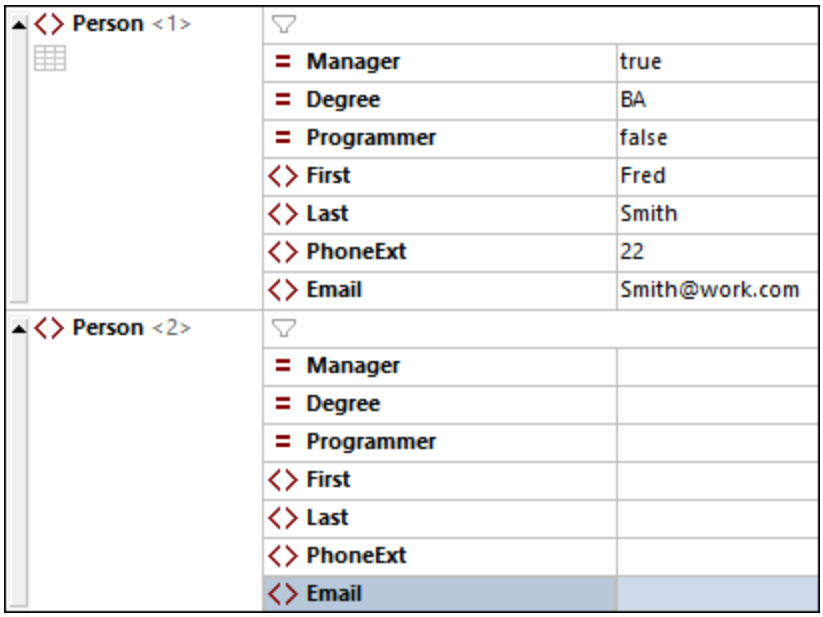


3. Wählen Sie das **Person**-Element in der Eingabehilfe "Gleichrangige" aus und klicken Sie entweder auf **Einfügen nach** oder **Anhängen**. Daraufhin wird ein neues **Person**-Element angehängt (*siehe Abbildung unten*).



4. Wenn das Element **Person** ausgewählt ist, werden die Child-Attribute und -Elemente dieses Elements in der Eingabehilfe "Children" angezeigt. Doppelklicken Sie auf Attribute und Elemente, um dieselben Child-Nodes wie für das erste **Person**-Element hinzuzufügen. Wenn der Fokus im Hauptfenster vom Element **Person** zu einem hinzugefügten Child-Element geändert wird, können Sie auf eine von zwei Arten Zusätzliche Diagrammarten Children des Elements **Person** hinzufügen: (i) Bringen Sie das Element **Person** in den Fokus (indem Sie es auswählen) und ein neues Child aus der Eingabehilfe "Children" hinzufügen; (ii) Fügen Sie, während sich der Fokus auf dem hinzugefügten Child-Element befindet, aus der Eingabehilfe "Gleichrangige" ein gleichrangiges Child-Element hinzu. Child-Nodes von **Person**, die bereits hinzugefügt wurden, erscheinen in beiden Eingabehilfen ausgegraut.

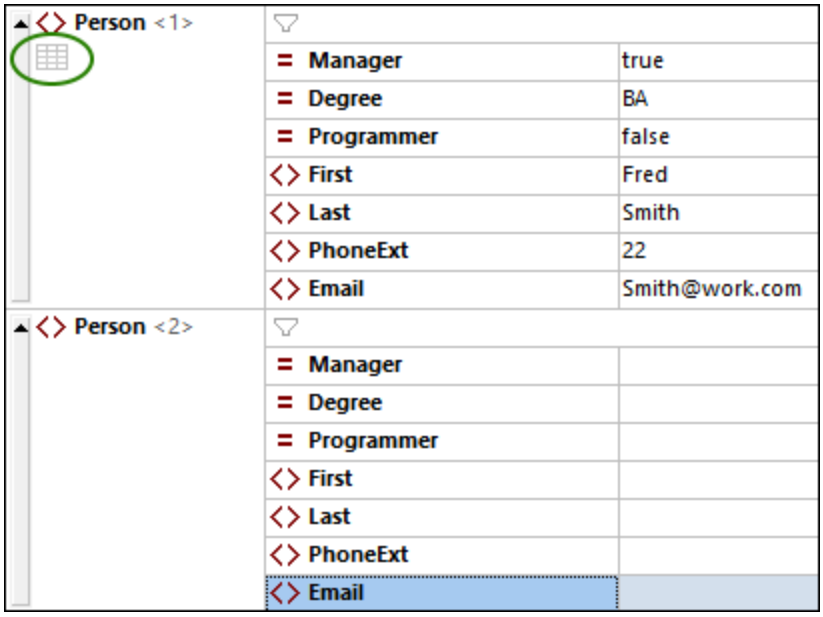
© 2016-2022 Altova GmbH



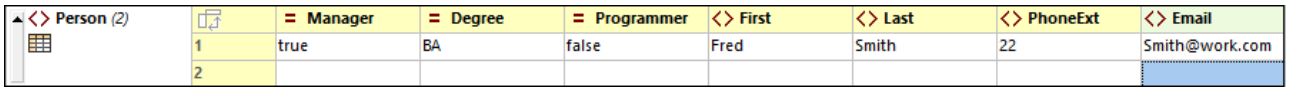
Sie könnten in der normalen Grid-Ansicht Inhalt für die Child-Nodes des Elements **Person** hinzufügen, wechseln wir jedoch nun in die Tabellenansicht der Grid-Ansicht, da sich diese besser zum Editieren einer Struktur mit mehreren Instanzen wie z.B. bei **Person** eignet.

2.5.7 Bearbeiten in der Tabellenansicht

Die Grid-Ansicht enthält eine spezielle Ansicht namens Tabellenansicht, die sich zum Bearbeiten von Elementen mit mehreren Instanzen eignet. So hat etwa das Element **Person** mehrere Instanzen (*siehe Abbildung unten*) und kann daher als Tabelle angezeigt werden. Um ein solches Element als Tabelle anzuzeigen, klicken Sie auf das **Tabellenansicht**-Symbol der ersten Instanz des Elements. In der Abbildung unten ist das **Tabellenansicht**-Symbol des **Person**-Elements grün umrandet. (Wählen Sie alternativ dazu den Menübefehl **XML | Als Tabelle anzeigen** oder klicken Sie auf die Symbolleisten-Schaltfläche in der Grid-Ansichts-Symbolleiste 296).



Wenn Sie auf das Tabellenansicht-Symbol klicken, wird das Element **Person** als Tabelle angezeigt. Die Child-Nodes des Elements (seine Attribute und Elemente) werden dabei als Spalten angezeigt, während die einzelnen **Person**-Elemente als Zeilen angezeigt werden (*siehe Abbildung unten*).



Vorteile der Tabellenansicht
Die Tabellenansicht bietet die folgenden Vorteile:

- Sie können eine Spaltenüberschrift mit der Maus verschieben, um ganze Spalten relativ zueinander zu verschieben. Im eigentlichen XML-Dokument verschiebt sich dabei die relative Position von Child-Nodes aller Element-Instanzen (die den Zeilen der Tabelle entsprechen).
- Tabellen - und die Element-Instanzen, die diesen entsprechen - können nach dem Inhalt jeder beliebigen Spalte in aufsteigender und absteigender Reihenfolge sortiert werden. Verwenden Sie dazu den Menübefehl **XML | Aufsteigend sortieren** oder **Absteigend sortieren**.
- Mit Hilfe der Befehle im Menü **XML** können rasch zusätzliche Zeilen (d.h. Element-Instanzen) angehängt oder eingefügt werden. Der Vorteil davon ist, dass nicht nur ein neues Element, sondern alle seine durch die Spalten der Tabelle repräsentierten Children hinzugefügt werden.
- Sie können *strukturierte Daten* von und in Produkte von Drittanbietern wie z.B. Excel kopieren.
- Diese intelligenten Bearbeitungsfunktionen von XMLSpy stehen auch in der Tabellenansicht zur Verfügung.

Anzeigen eines Elements mit mehreren Instanzen als Tabelle

Um den Elementtyp **Person** in Tabellenform anzuzeigen, gehen Sie folgendermaßen vor:

1. Klicken Sie, wie oben beschrieben, auf das **Tabellensicht**-Symbol der ersten Instanz des **Person**-Elements.

© 2016-2022 Altova GmbH

2. Wählen Sie die Menüoption **Ansicht | Optimale Breite** oder das Symbol **Optimale Breite** der Grid-Ansichts-Symbolleiste[296].

Anmerkung: Die Tabellensicht kann für alle Elemente mit mehreren Instanzen aktiviert oder deaktiviert werden. Child-Elemente, die als Tabellen angezeigt wurden, werden jedoch weiterhin als Tabellen angezeigt.

Eingabe von Inhalt in der Tabellenansicht

Um Inhalt für das zweite **Person**-Element einzugeben, doppelklicken Sie in jede der Tabellenzellen in der zweiten Zeile und geben Sie Daten ein. Die intelligenten Bearbeitungsfunktionen sind auch innerhalb der Zellen einer Tabelle aktiv, sodass Sie, wo verfügbar, Optionen aus Dropdown-Listen auswählen können (z.B. Booleschen Inhalt und die Enumerationen des Attributs **Degree**).

	= Manager	= Degree	= Programmer	<> First	<> Last	<> PhoneExt	<> Email
1	true	BA	false	Fred	Smith	22	Smith@work.com
2	false	MA	true	Alfred	Aldrich	33	Aldrich@work.com

Dynamische Valdierung

Beachten Sie jedoch, dass **PhoneExt**, wie im Schema definiert, eine Ganzzahl zwischen **0** und **99** sein muss, damit die Datei gültig ist. Sie können die XMLSpy-Funktion zur Validierung während der Bearbeitung aktivieren. Wenn diese Funktion aktiv ist, wird die Datei, jedes Mal, wenn der Fokus auf einen neuen Node verschoben wird, validiert. Testen Sie die dynamische Valdierung folgendermaßen:

1. Aktivieren Sie den Menübefehl **XML | Bei Bearbeitung validieren**.
2. Geben Sie, wie in der Abbildung unten gezeigt, einen ungültigen **PhoneExt**-Wert ein (einen beliebigen Wert, der größer als **99** ist).
3. Drücken Sie die **Tabulatortaste**. Daraufhin erscheinen in der Zelle **PhoneExt** ein Fehlersymbol und ein Smart Fix-Symbol (*siehe Abbildung unten*).
4. Wenn Sie die Maus über das Fehlersymbol platzieren, sehen Sie die Validierungsfehlermeldung (*Abbildung unten*).

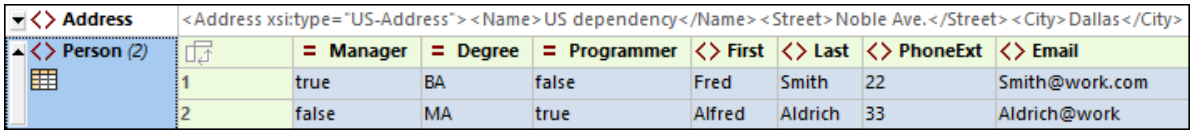


5. Klicken Sie auf das Smart Fix-Symbol und anschließend auf die angezeigte Smart Fix-Option. Daraufhin wird der ungültige Werte durch einen gültigen Wert ersetzt und das Fehlersymbol verschwindet.

Kopieren von XML Daten aus und in Tabellenkalkulationsprogramme

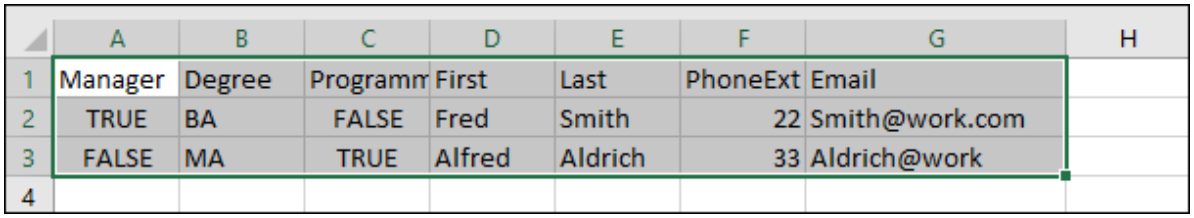
In der Tabellenansicht können Sie Daten als tabulatorgetrennten Text kopieren, sodass er mit Tabellenkalkulationsprogrammen wie z.B. MS Excel ausgetauscht werden kann. Um aus Ihrer XML-Datei zu kopieren, gehen Sie folgendermaßen vor:

1. Klicken Sie auf das Element **Person** (*siehe Abbildung unten*). Dadurch werden die Spaltenüberschriften sowie beide Zeilen der Tabelle ausgewählt.



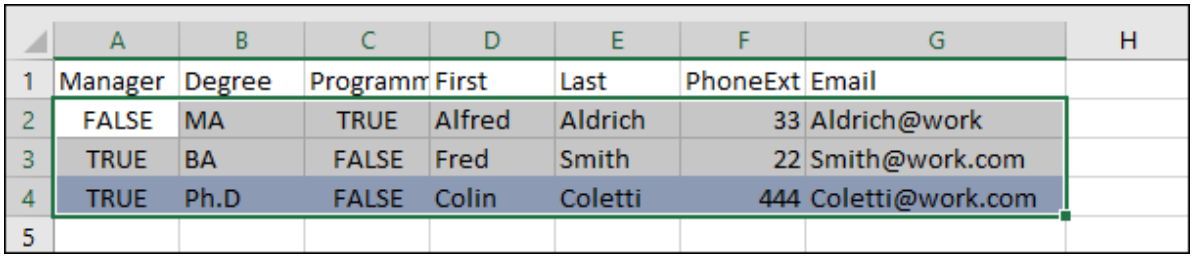
Address	<Address xsi:type="US-Address"><Name>US dependency</Name><Street>Noble Ave.</Street><City>Dallas</City>						
Person (2)	Manager	Degree	Programmer	First	Last	PhoneExt	Email
1	true	BA	false	Fred	Smith	22	Smith@work.com
2	false	MA	true	Alfred	Aldrich	33	Aldrich@work

2. Klicken Sie mit der rechten Maustaste auf die Auswahl und wählen Sie im daraufhin angezeigten Kontextmenü den Befehl **Kopieren | Als tabulatorgetrennten Text kopieren**. Drücken Sie alternativ dazu **Strg+ C**.
3. Wechseln Sie zu einem Excel-Arbeitsblatt, wählen Sie die Zelle A1 aus und fügen Sie die XML-Daten ein (**Strg+V**). Die Daten werden gemäß der Tabellenstruktur in der Tabellenansicht in Form von Zeilen eingegeben (*siehe Abbildung unten*).



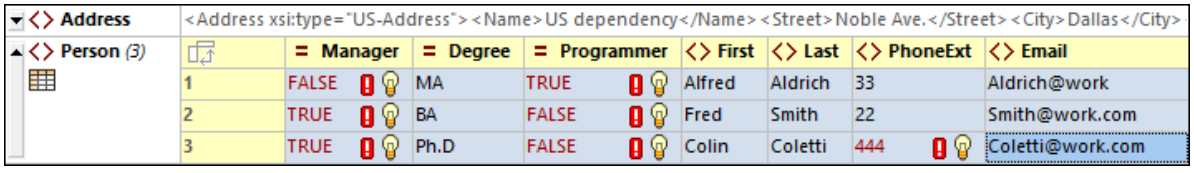
	A	B	C	D	E	F	G	H
1	Manager	Degree	Programm	First	Last	PhoneExt	Email	
2	TRUE	BA	FALSE	Fred	Smith	22	Smith@work.com	
3	FALSE	MA	TRUE	Alfred	Aldrich	33	Aldrich@work	
4								

4. Geben Sie, wie in der Abbildung unten gezeigt, eine neue Datenzeile in Excel ein. Stellen Sie sicher, dass Sie für das Element **PhoneExt** eine dreistellige Zahl (z.B. **444**) eingeben.
5. Markieren Sie die Tabellendaten in Excel (ohne die Spaltenüberschriften) (*grüne Umrandung in der Abbildung unten*) und kopieren Sie sie mit **Strg+C**.



	A	B	C	D	E	F	G	H
1	Manager	Degree	Programm	First	Last	PhoneExt	Email	
2	FALSE	MA	TRUE	Alfred	Aldrich	33	Aldrich@work	
3	TRUE	BA	FALSE	Fred	Smith	22	Smith@work.com	
4	TRUE	Ph.D	FALSE	Colin	Coletti	444	Coletti@work.com	
5								

6. Stellen Sie in XMLSpy sicher, dass der Befehl **XML | bei Bearbeitung validieren** aktiviert ist.
7. Wählen Sie in XMLSpy in der Tabellenansicht Ihres XML-Dokuments die Zelle **Manager** der ersten Zeile aus und fügen Sie den Inhalt der Zwischenablage mit **Strg+V** ein. Ihre neue Tabelle sieht in etwa wie in der Abbildung unten aus.



Address	<Address xsi:type="US-Address"><Name>US dependency</Name><Street>Noble Ave.</Street><City>Dallas</City>							
Person (3)	Manager	Degree	Programmer	First	Last	PhoneExt	Email	
1	FALSE	MA	TRUE	Alfred	Aldrich	33	Aldrich@work	
2	TRUE	BA	FALSE	Fred	Smith	22	Smith@work.com	
3	TRUE	Ph.D	FALSE	Colin	Coletti	444	Coletti@work.com	

8. Der Grund für die Validierungsfehler für die Booleschen Werte ist der Unterschied in der Groß- und Kleinschreibung zwischen XML und Excel. Um diese Fehler zu beheben, wenden Sie die Smart Fix-Korrekturen auf die entsprechenden Tabellenzellen an.

Sortieren der Tabellen nach Spalteninhalt

© 2016-2022 Altova GmbH

Eine Tabelle kann in der Tabellenansicht nach jeder ihrer Spalten in aufsteigender oder absteigender Reihenfolge sortiert werden. Wir möchten die Tabelle **Person** nach Nachnamen sortieren. Gehen Sie dazu folgendermaßen vor:

1. Klicken Sie auf die Kopfzeile der Spalte **Last**.

Person (3)		Manager	Degree	Programmer	First	Last	PhoneExt	Email
	1	true	BA	false	Fred	Smith	22	Smith@work.com
	2	false	MA	true	Alfred	Aldrich	33	Aldrich@work
	3	true	Ph.D	false	Colin	Coletti	444	Coletti@work.com

2. Wählen Sie die Menüoption **XML | Aufsteigend sortieren** oder klicken Sie das Symbol **Aufsteigend sortieren** der Grid-Ansichts-Symbolleiste[296]. Die Spalte wird unter Einbeziehung der gesamten Tabelle alphabetisch sortiert. Die Spalte bleibt hierbei markiert.

Person (3)		Manager	Degree	Programmer	First	Last	PhoneExt	Email
	1	false	MA	true	Alfred	Aldrich	33	Aldrich@work
	2	true	Ph.D	false	Colin	Coletti	444	Coletti@work.com
	3	true	BA	false	Fred	Smith	22	Smith@work.com

Da die Telefondurchwahl **444** korrekt, aber ungültig ist, müssen wir das XML-Schema ändern, damit diese Nummer gültig ist. Dies wird im nächsten Abschnitt beschrieben.

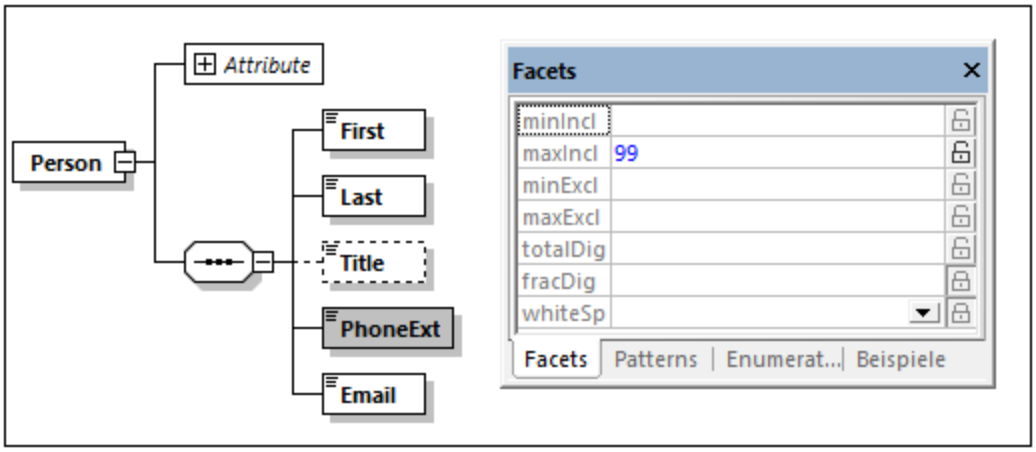
2.5.8 Ändern des Schemas

Da wir eine Durchwahl haben, die sich außerhalb des im XML-Schema definierten Bereichs (**0** bis **99**) befindet, wollen wir den Bereich bis **999** erweitern. Gehen Sie dazu vor, wie unten beschrieben.

1. Wählen Sie in der Grid-Ansicht eine beliebige der **PhoneExt**-Zellen aus (*siehe Abbildung unten*).

Person (3)		Manager	Degree	Programmer	First	Last	PhoneExt	Email
	1	false	MA	true	Alfred	Aldrich	33	Aldrich@work
	2	true	Ph.D	false	Colin	Coletti	444	Coletti@work.com
	3	true	BA	false	Fred	Smith	22	Smith@work.com

2. Wählen Sie die Menüoption **DTD/Schema | Gehe zu Definition** oder klicken Sie das Symbol **Gehe zu Definition** der Grid-Ansichts-Symbolleiste[296]. Dadurch wird das damit verknüpfte Schema, in diesem Fall **AddressLast.xsd** geöffnet und die Definition **PhoneExt** wird markiert (*siehe Abbildung unten*).



3. Das Facet **maxIncl** des Elements ist **99** (*siehe Abbildung*). Ändern Sie diesen Wert in **999** und speichern Sie das Schema anschließend.
4. Gehen Sie zurück zum XML-Dokument und validieren Sie es. Es ist nun gültig.
5. Speichern Sie Ihre Datei unter **companyLast.xml**.

Anmerkung: Der Ordner "Tutorial" von XMLSpy enthält eine Datei namens **companyLast.xml**, die dieselben Daten enthält wie die Datei, die Sie nach Beendigung dieses Tutorials gespeichert haben.

© 2016-2022 Altova GmbH

2.6 XSLT-Transformationen

Ihr Ziel in diesem Abschnitt
Erstellen einer HTML-Datei von einer XML-Datei mittels eines XSLT Stylesheet zum Transformieren der XML-Datei. Beachten Sie, dass die XML-Datei bei einer "Transformation" nicht geändert wird. Es wird nur eine neue Ausgabedatei erzeugt. Das Wort "Transformation" ist eine Konvention.

Methode:

Gehen Sie beim Transformieren folgendermaßen vor:

- Weisen Sie dem XML-Dokument eine vordefinierte XSL-Datei, nämlich `company.xsl` zu.
- Führen Sie die Transformation mit Hilfe eines der beiden in XMLSpy integrierten Altova XSLT-Prozessoren durch (*siehe Hinweis unten*).

In diesem Abschnitt verwendete Befehle

Die folgenden XMLSpy-Befehle werden in diesem Abschnitt verwendet:



XSL/XQuery | XSL zuweisen weist dem aktiven XML-Dokument eine XSL-Datei zu.



XSL/XQuery | Gehe zu XSL öffnet die vom aktiven XML-Dokument referenzierte XSL-Datei.



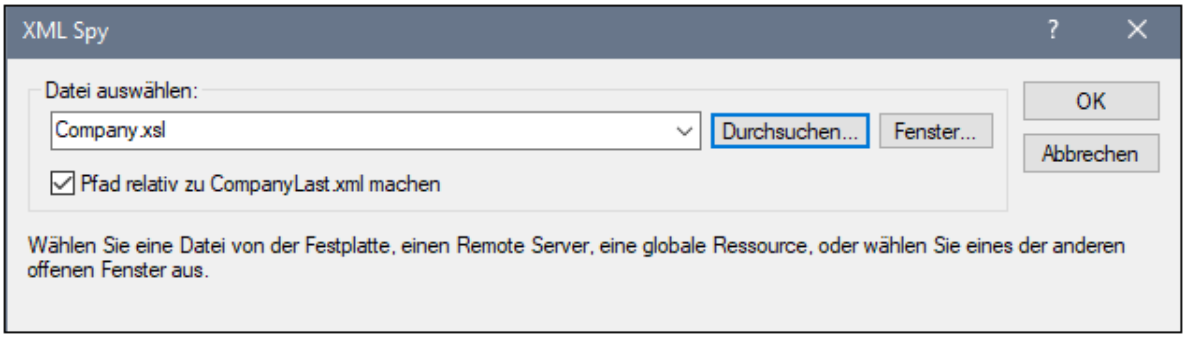
XSL/XQuery | XSL-Transformation (F10) oder die Symbolleistenschaltfläche transformiert das aktive XML-Dokument mit Hilfe des diesem zugewiesenen XSL Stylesheet. Wenn keine XSL-Datei zugewiesen wurde, werden Sie beim Aufrufen dieses Befehls aufgefordert, eine zuzuweisen.

Anmerkung: XMLSpy verfügt über integrierte XSLT-Prozessoren für XSLT 1.0, 2.0 und 3.0. Der richtige Prozessor wird von XMLSpy automatisch auf Basis des Versionsattributs im Element `xsl:stylesheet` oder `xsl:transform` ausgewählt. In diesem Tutorial verwenden wir für die Transformation ein XSLT 3.0 Stylesheet. Bei Aufruf des Befehls **XSL-Transformation** wird automatisch der XSLT 3.0-Prozessor verwendet.

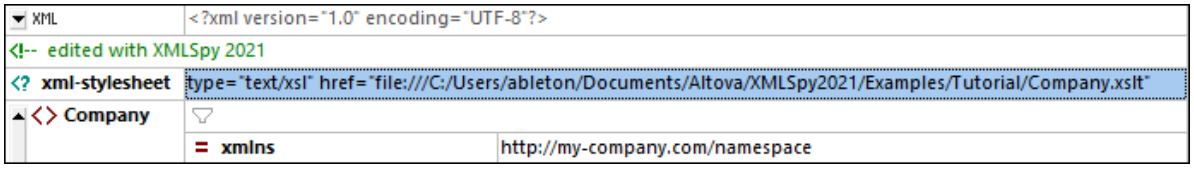
2.6.1 Zuweisen einer XSLT-Datei

So weisen Sie der Datei **`CompanyLast.xml`** eine XSLT-Datei zu:

1. Klicken Sie im Hauptfenster auf das Register `CompanyLast.xml`, um es aktiv zu setzen und wechseln Sie in die Textansicht.
2. Wählen Sie die Menüoption **XSL/XQuery | XSL zuweisen**.
3. Klicken Sie auf die **Durchsuchen**-Schaltfläche und wählen Sie im Ordner Tutorial die Datei `Company.xsl` aus. Sie können im Dialogfeld die Option "*Pfad relativ zu CompanyLast.xml machen*", aktivieren, um den Pfad zur XSL-Datei im XML-Dokument relativ zu machen.



4. Klicken Sie auf **OK** um dem XML-Dokument die XSL-Datei zuzuweisen.
5. Wechseln Sie in die Grid-Ansicht, um die Zuweisung zu sehen (*Abbildung unten).* Im XML-Dokument wird eine XML-Stylesheet-Verarbeitungsanweisung eingefügt, die die XSL-Datei referenziert. Wenn Sie das Kontrollkästchen "*Pfad relativ zu CompanyLast.mxl machen*" aktiviert haben, ist der Pfad relativ, andernfalls ist er absolut (wie in der Abbildung).

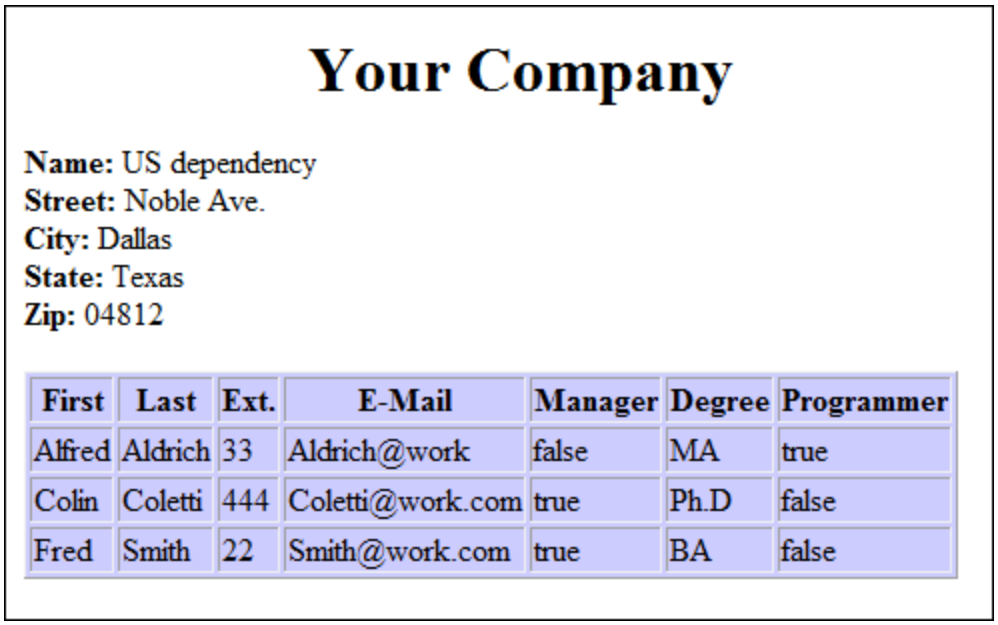


2.6.2 Transformieren der XML-Datei

Um das XML-Dokument mit Hilfe der zugewiesenen XSL-Datei zu transformieren, gehen Sie folgendermaßen vor:

1. Stellen Sie sicher, dass die XML-Datei das aktive Dokument ist.
2. Wählen Sie die Menüoption **XSL/XQuery | XSL-Transformation (F10)** oder klicken Sie in der Symbolleiste auf die entsprechende Befehlsschaltfläche. Daraufhin wird die Transformation anhand des im XML-Dokument referenzierten XSLT-Stylesheet gestartet. Das Ausgabedokument wird in der Browser-Ansicht angezeigt; es hat den Namen **`XSL Output.html`**. Die **`Company/Address`**-Daten werden in einem Block links oben angezeigt, während die **`Company/Person`**-Daten unterhalb davon in Tabellenform dargestellt werden.

 © 2016-2022 Altova GmbH



Your Company

Name: US dependency
Street: Noble Ave.
City: Dallas
State: Texas
Zip: 04812

First	Last	Ext.	E-Mail	Manager	Degree	Programmer
Alfred	Aldrich	33	Aldrich@work	false	MA	true
Colin	Coletti	444	Coletti@work.com	true	Ph.D	false
Fred	Smith	22	Smith@work.com	true	BA	false

Anmerkung: Da die Datei `Company.xslt` ein XSLT 3.0-Dokument ist, wird für die Transformation automatisch der integrierte Altova XSLT 3.0-Prozessor ausgewählt. (Wenn die HTML-Ausgabedatei nicht generiert wird, stellen Sie sicher, dass die Standarddateierweiterung im Dialogfeld "Optionen" (**Extras | Optionen**) im Abschnitt "*XSL*" auf `.html` gesetzt wurde.) Damit wird sichergestellt, dass der Browser das Ausgabedokument korrekt als HTML-Datei liest.

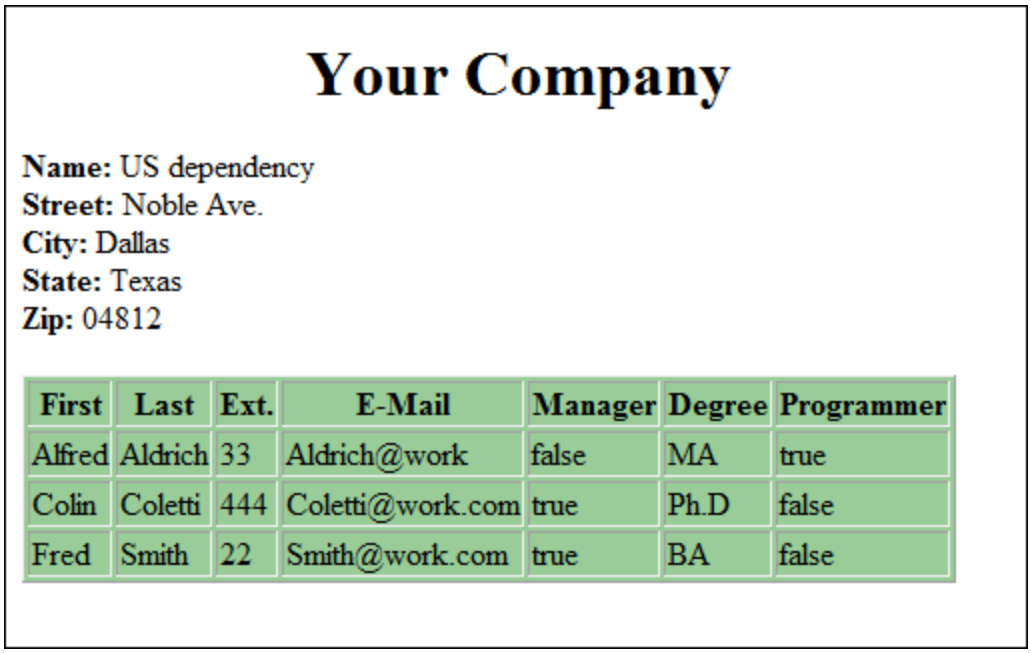
2.6.3 Ändern der XSL-Datei

Sie können die Ausgabe der Datei ändern, indem Sie die zugrunde liegende XSL-Datei bearbeiten. In diesem Fall soll die Hintergrundfarbe der Tabelle von `#ccccff` in `#99cc99` geändert werden. Gehen Sie dazu folgendermaßen vor:

1. Klicken Sie auf das Register `CompanyLast.xml` um das Dokument zu aktiven zu machen.
2. Wählen Sie die Menüoption **XSL/XQuery | Gehe zu XSL**. Daraufhin wird die im XML-Dokument referenzierte Datei `Company.xslt` geöffnet.
3. Suchen Sie nach dem Start-Tag des Elements `table` und anschließend dem Attribut `bgcolor` (*in der Abbildung unten markiert*). Ändern Sie den Wert des Attributs von `#ccccff` in `#99cc99`.

```
<xsl:template match="/">
   <html>
      <head><title>Your company</title></head>
         <body>
            <h1><center>Your Company</center></h1>
            <xsl:apply-templates select="//my:Address"/>
               <table border="1" bgcolor="#ccccff">
                  <thead align="center">
                     <td><strong>First</strong></td>
                     <td><strong>Last</strong></td>
                     <td><strong>Ext.</strong></td>
                     <td><strong>E-Mail</strong></td>
                     <td><strong>Manager</strong></td>
                     <td><strong>Degree</strong></td>
                     <td><strong>Programmer</strong></td>
                  </thead>
                  <xsl:apply-templates select="//my:Person"/>
               </table>
         </body>
   </html>
</xsl:template>
```

4. Wählen Sie die Menüoption **Datei | Speichern**, um die Änderung zu speichern.
5. Klicken Sie auf das Register `CompanyLast.xml`, um die XML-Datei zur aktiven zu machen.
6. Wählen Sie den Befehl **XSL/XQuery | XSL-Transformation** oder drücken Sie alternativ dazu **F10**. In der Browser-Ansicht wird daraufhin eine neue `XSLT Output.html`-Datei mit einer Tabelle mit einer neuen Hintergrundfarbe angezeigt (*siehe Abbildung unten*).



Your Company

Name: US dependency
Street: Noble Ave.
City: Dallas
State: Texas
Zip: 04812

First	Last	Ext.	E-Mail	Manager	Degree	Programmer
Alfred	Aldrich	33	Aldrich@work	false	MA	true
Colin	Coletti	444	Coletti@work.com	true	Ph.D	false
Fred	Smith	22	Smith@work.com	true	BA	false

7. Wählen Sie die Menüoption **Datei | Speichern** und speichern Sie das Ausgabedokument unter dem Namen `Company.html`.

© 2016-2022 Altova GmbH

2.7 Projektverwaltung

In diesem Abschnitt werden Sie mit den Projektverwaltungsfunktionen von XMLSpy vertraut gemacht. Nachdem Sie zuerst die Vorteile der Gliederung von XML-Dateien in Projekten kennengelernt haben, wird in diesem Abschnitt nun beschrieben, wie Sie die soeben erstellten Dateien in einem einfachen Projekt strukturieren.

2.7.1 Vorteile von Projekten

Im Folgenden sind die Vorteile der Strukturierung von XML-Dateien in Form von Projekten aufgelistet.

- Dateien können nach einem gemeinsamen Kriterium in Form von Ordnern gegliedert werden. So könnten Sie z.B. XML- und XSD-Dateien in separate Ordner einteilen. Sie können ganz nach Bedarf jede beliebige Hierarchie erstellen.
- Jeder Ordner hat bestimmte Eigenschaften, die Sie definieren können. So können Sie etwa einem Ordner von XML-Dateien ein Schema für die Validierung zuweisen. Daraufhin können alle Dateien in diesem Projektordner auf einmal anhand der Schema-Datei des Ordners validiert werden. Wenn Sie die Schemazuweisung des Projektordners ändern, können Sie schnell eine neue Batch-Validierung durchführen. Sie können eine Reihe weiterer nützlicher Ordnereigenschaften definieren, wie z.B. eine XSLT-Datei für Batch-Transformationen anhand einer einzigen XSLT-Datei.
- Die Batch-Verarbeitung kann auf bestimmte Ordner oder das ganze Projekt angewendet werden.
- DTD / XML-Schemas können bestimmten Ordnern zugewiesen werden, um eine sofortige Validierung der darin befindlichen Dateien zu ermöglichen.
- XSL-Dateien können bestimmten Ordnern zugewiesen werden, um die XML-Dateien in diesem Ordner anhand der zugewiesenen XSLT-Datei zu transformieren.
- Es können Zielordner für die XSL-Transformationsdateien definiert werden.

Alle oben aufgelisteten Einstellungen können über die Menüoption **Projekt | Eigenschaften**definiert werden. Projektbefehle stehen außerdem in den Kontextmenüs des Projekts und einzelner Projektordner zur Verfügung. Im nächsten Abschnitt werden Sie über das Menü **Projekt** ein Projekt erstellen.

Des Weiteren stehen die folgenden Projektfunktionen zur Verfügung:

- XML-Dateien können unter Versionskontrolle gestellt werden (**Projekt | Versionskontrolle | Zu Versionskontrolle hinzufügen**). (Nähere Informationen dazu finden Sie im Abschnitt Versionskontrolle[1320]).
- Sie können zu Projekten externe Ordner in Ihrem Netzwerk[1337] sowie Web-Ordner[1340] hinzufügen. Dadurch können alle Funktionalitäten von Projektordnern wie z.B. Validierung und Transformationen auf Ordner in Ihrem Netzwerk oder im Internet angewendet werden.

2.7.2 Erstellen eines Projekts

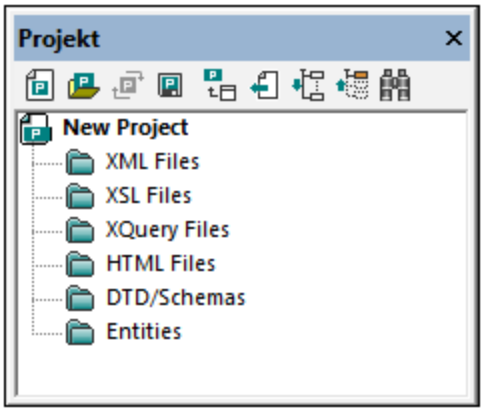
Im Hauptfenster ist nun eine Reihe von Tutorial-Dateien geöffnet. Sie können diese Dateien in einem Tutorial-Projekt gliedern. Zuerst müssen Sie ein neues Projekt erstellen und anschließend die Tutorial-Dateien zu den jeweiligen Unterordnern des Projekts hinzufügen.

© 2016-2022 Altova GmbH

Erstellen eines neuen Projekts

Gehen Sie dazu folgendermaßen vor:

1. Wählen Sie die Menüoption **Projekt | Neues Projekt**. Im Projektfenster wird ein Projektordner mit dem Namen `New Project` erstellt (*Abbildung unten*). Das neue Projekt enthält leere Ordner für typische XML-Dateikategorien.



2. Klicken Sie auf das Register `CompanyLast.xml` um diese Datei im Hauptfenster aktiv zu setzen.
3. Wählen Sie die Menüoption **Projekt | Aktive und verwandte Dateien hinzufügen**. Es werden zwei Dateien zum Projekt hinzugefügt: `CompanyLast.xml` und `AddressLast.xsd`. Die XML-Datei wird zum Unterordner *XML* hinzugefügt, da sie die aktive Datei ist. Die Schema-Datei wird zum Ordner *DTD/Schemas* hinzugefügt, da die XML-Datei eine Referenz auf diese Datei enthält und sie zu einer verwandten Datei macht. Beachten Sie, dass durch Processing Instructions referenzierte Dateien wie z.B. XSLT-Dateien nicht als verwandte Dateien gelten.
4. Wählen Sie die Menüoption **Projekt | Projekt speichern** und speichern Sie das Projekt unter dem Namen `Tutorial`.

Anmerkung: Ordner (nicht aber das Projekt) haben jeweils eine Eigenschaft namens *Dateierweiterungen*. Es handelt sich dabei um eine Liste von durch Kommas getrennten Dateierweiterungen (z.B. `xml;svg;wml`). Anhand dieser Liste wird ermittelt, welche Dateien zu welchen Ordnern hinzugefügt werden, wenn Dateien zu einem Projekt hinzugefügt werden. Wenn z.B. wie oben aktive und verwandte Dateien zu einem Projekt hinzugefügt werden, wird anhand der *Dateierweiterungseigenschaften* des Ordners ermittelt, zu welchem Ordner die Dateien hinzugefügt werden sollen.

Projekt- und Ordnereigenschaften

Eigenschaften (wie z.B. das Schema für die Validierung und die XSLT-Datei für die Transformation) können nicht nur für das gesamte Projekt, sondern auch für einzelne Ordner definiert werden. Anschließend können Sie am gesamten Projekt oder einzelnen Ordnern Aktionen wie Validierung und Transformation durchführen. Um eine Aktion durchzuführen, klicken Sie mit der rechten Maustaste auf das Projekt oder den Ordner und wählen Sie im Kontextmenü die gewünschte Aktion aus.

Beachten Sie die folgenden Punkte:

- Eine für einen Ordner definierte Eigenschaft setzt dieselbe Eigenschaft des Projekts außer Kraft.
- Wenn eine Eigenschaft für ein Projekt definiert wurde, wird sie auf alle Ordner, für die diese Eigenschaft noch nicht definiert wurde, angewendet.

- Wenn eine Aktion an einem Projekt durchgeführt wird, wird sie auf alle anwendbaren Dateitypen in allen Ordnern des Projekts angewendet. Wenn z.B eine Validierung an einem Projekt durchgeführt wird, wird sie an allen XML-Dateien in allen Ordnern des Projekts durchgeführt. In diesem Fall wird das für das Projekt definierte Schema für alle Validierungen mit Ausnahme der Dateien, die sich in Ordnern befinden, für deren Schemavalidierungseigenschaft ein anderes Schema definiert wurde, verwendet.

Hinzufügen von Dateien zu einem Projekt

Sie können auch andere Dateien zum Projekt hinzufügen. Gehen Sie dazu folgendermaßen vor:

1. Klicken Sie auf eine beliebige offene XML-Datei (Erweiterung **.xml**) mit Ausnahme von **CompanyLast.xml**, um diese XML-Datei zur aktiven zu machen. (Wenn keine andere XML-Datei geöffnet ist, öffnen Sie eine oder erstellen Sie eine neue XML-Datei).
2. Wählen Sie die Menüoption **Projekt | Aktive Datei zu Projekt hinzufügen**. Die XML-Datei wird anhand ihres Dateityps (**.xml**) zum Ordner *XML Files* hinzugefügt.
3. Fügen Sie auf dieselbe Art eine HTML-Datei und eine XSDL-Datei (z.B. **Company.html** und **AddressFirst.xsd**) zum Projekt hinzu. Diese Dateien werden zum Ordner *HTML Files* bzw. *DTD/Schemas* hinzugefügt.
4. Speichern Sie das Projekt, entweder durch Auswahl der Menüoption **Projekt | Projekt speichern** oder durch Auswahl einer beliebigen Datei oder eines Ordners im Projektfenster und Klicken auf das Symbol **Speichern** in der Symbolleiste (oder **Datei | Speichern**).

Anmerkung: Alternativ dazu können Sie mit der rechten Maustaste auf einen Projektordner klicken und den Befehl **Aktive Datei hinzufügen** auswählen, um die aktive Datei zu diesem bestimmten Ordner hinzuzufügen.

Andere nützliche Befehle

Hier sind einige weitere gebräuchliche Projektbefehle aufgelistet:

- Um einen neuen Ordner zu einem Projekt hinzuzufügen, wählen Sie die Menüoption **Projekt | Projektordner zu Projekt hinzufügen** und fügen Sie den Namen des Projektordners ein.
- Um einen Ordner aus einem Projekt zu löschen, rechtsklicken Sie auf den Ordner und wählen Sie im Kontextmenü den Befehl **Löschen**.
- Um eine Datei aus einem Projekt zu löschen, wählen Sie die Datei aus und drücken Sie die **Entf**-Taste.

© 2016-2022 Altova GmbH

2.8 Fertig!

Wenn Sie bis hierher gekommen sind, herzlichen Glückwunsch und vielen Dank!

Wir hoffen, dass Ihnen diese Anleitung geholfen hat, um die Grundzüge von XMLSpy kennen zu lernen und dass Sie nun in der Lage sind, Ihre XML-Dateien mit XMLSpy als Editor zu bearbeiten. Wenn Sie nähere Informationen zu bestimmten Funktionen benötigen, konsultieren Sie bitte den Index oder die Suchfunktion dieses Handbuchs. Sie können die PDF-Version dieses Tutorials auch ausdrucken. Sie steht unter dem Namen **`Tutorial.pdf`** in Ihrem XMLSpy Applikationsordner[34] zur Verfügung.

3 Benutzeroberfläche und Umgebung

Dieser Abschnitt enthält eine Beschreibung :

- der Benutzeroberfläche der Applikation[115] und
- der Applikationsumgebung[135].

Der Abschnitt "Benutzeroberfläche"[115] beginnt mit einer Übersicht über die Benutzeroberfläche. Anschließend werden die einzelnen Fenster der Benutzeroberfläche näher beschrieben. Außerdem erfahren Sie, wie Sie die Größe der Fenster anpassen und die Fenster verschieben können und mit den Fenstern und der Benutzeroberfläche arbeiten.

Im Abschnitt "Applikationsumgebung"[135] werden die verschiedenen Einstellungen erläutert, mit denen festgelegt wird, wie die Dateien angezeigt und bearbeitet werden können. Außerdem wird darin erklärt, wie und wo Sie Ihre Applikation anpassen können. In diesem Abschnitt erfahren Sie, wo wichtige Beispiel- und Tutorial-Dateien auf Ihrem Rechner installiert wurden. Weiter hinten in diesem Abschnitt finden Sie einen Link zur Altova Website, wo Sie eine Feature Matrix Ihrer Applikation finden, Ihr Benutzerhandbuch in verschiedenen Formaten abrufen können und Informationen über die verschiedenen Support-Optionen für Sie finden. Des Weiteren finden Sie dort Informationen über andere Produkte von Altova.

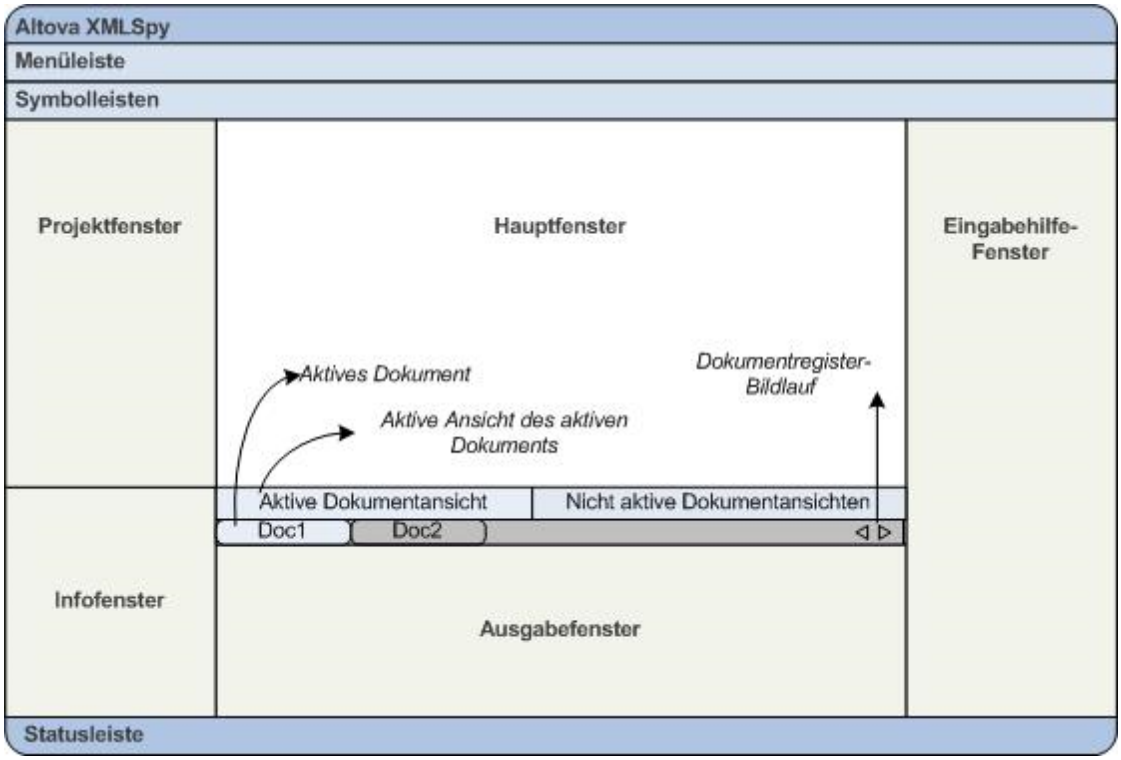
© 2016-2022 Altova GmbH

3.1 Die grafische Benutzeroberfläche

Die grafische Benutzeroberfläche besteht aus einem Hauptfenster und mehreren Seitenleisten (siehe *Abbildung unten*). Standardmäßig sind die Seitenleisten rund um das Hauptfenster angeordnet und werden in die folgenden Gruppen eingeteilt:

- Projekt-Fenster
- Info-Fenster
- Eingabehilfen: Elemente, Attribute, Entities, usw. (je nachdem, welche Dokumentart gerade aktiv ist)
- Ausgabefenster: Meldungen, XPath/XQuery, XSL-Übersicht, HTTP, In Dateien suchen, In Schemas suchen, In XBRL suchen, Diagramme

Das Hauptfenster und die Seitenleisten werden in den Unterabschnitten dieses Abschnitts beschrieben.



Außerdem enthält die Benutzeroberfläche eine Menüleiste, eine Statusleiste sowie Symbolleisten, die alle im Kapitel Menüleiste, Symbolleisten und Statusleiste 133 beschrieben sind.

Ein- und Ausblenden der Seitenleisten

Die Seitenleistengruppen (Projektfenster, Info-Fenster, Eingabehilfen, Ausgabefenster) können über die entsprechenden Befehle im Menü **Fenster** ein- und ausgeblendet werden. Eine Seitenleiste, die gerade angezeigt wird, (oder eine mit Registern versehene Seitenleistengruppe) kann auch durch Rechtsklick auf die Titelleiste der Seitenleiste (oder die Seitenleistengruppe) und Auswahl des Befehls **Ausblenden** ausgeblendet werden.

Wenn Sie eines der Ausgabefenster schließen, können Sie dieses mit dem Menübefehl **Fenster | Ausgabefenster** wieder öffnen.

Seitenleisten freischwebend anzeigen und andocken

Eine einzelne Seitenleiste kann entweder frei schwebend oder angedockt angezeigt werden. Wenn ein frei schwebendes Fenster angedockt wird, wird es an der Stelle angedockt, an der es zuletzt angedockt war. Ein Fenster kann auch als Register in einem anderen Fenster angedockt werden.

Ein Fenster kann auf die folgenden Arten an- oder abgedockt werden:

- Klicken Sie mit der rechten Maustaste auf die Titelleiste eines Fensters und wählen Sie den entsprechenden Befehl aus (**Abgedockt** oder **Angedockt**).
- Doppelklicken Sie auf die Titelleiste des Fensters. Wenn das Fenster angedockt war, wird es nun frei schwebend angezeigt. Wenn es frei schwebend angezeigt wurde, wird es nun dort angedockt, wo es zuletzt angedockt war.
- Ziehen Sie das Fenster (an der Titelleiste als Ziehpunkt) aus seiner angedockten Position bis es frei schwebend angezeigt wird. Ziehen Sie ein frei schwebendes Fenster (an seiner Titelleiste) an die Stelle, an der es angedockt werden soll. Es erscheinen zwei Gruppen blauer Pfeile. Mit den äußeren vier Pfeilen können Sie das Fenster relativ zum Applikationsfenster andocken (am oberen, rechten, unteren oder linken Rand der Benutzeroberfläche). Mit den inneren Pfeilen können Sie das Fenster relativ zu dem Fenster andocken, über dem sich der Cursor gerade befindet. Wenn Sie ein Fenster über den Mittelpunkt der inneren Pfeile ziehen (oder auf die Titelleiste eines Fensters), wird es als Registerkarte in dem Fenster angezeigt, in das es gezogen wurde.

Um ein Fenster, das als Registerkarte angezeigt wird, frei schwebend anzuzeigen, doppelklicken Sie auf sein Register. Um ein als Register angezeigtes Fenster aus einer Gruppe derartiger Fenster herauszuziehen, ziehen Sie es am Register heraus.

Automatisches Ausblenden der Seitenleisten

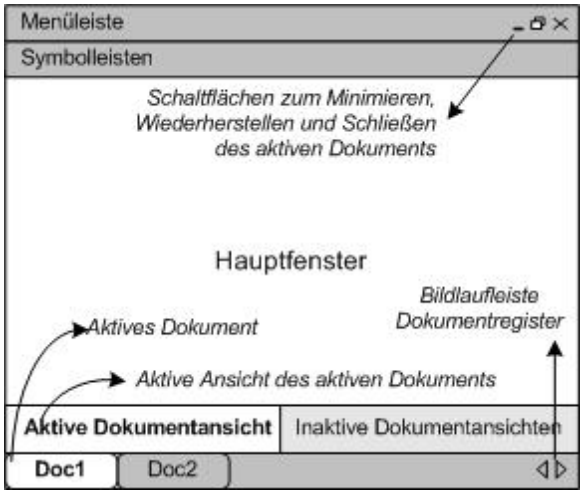
Mit der Funktion "Automatisch ausblenden" können Sie angedockte Fenster zu Schaltflächen am Rand des Applikationsfensters minimieren, um mehr Bildschirmplatz für das Hauptfenster und andere Fenster zu haben. Wenn Sie die Maus über eine minimierte Seitenleiste platzieren, wird sie ausgerollt.

Um Seitenleisten automatisch auszublenden und wiederherzustellen, klicken Sie in der Titelleiste des Fensters auf das Pin-Symbol (oder klicken Sie mit der rechten Maustaste auf die Titelleiste und wählen Sie **Automatisch ausblenden**).

3.1.1 Hauptfenster

Das Hauptfenster (*Abbildung unten*) ist der Bereich, in dem alle Dokumente angezeigt und bearbeitet werden.

© 2016-2022 Altova GmbH



Dateien im Hauptfenster

- Sie können eine beliebige Anzahl an Dokumenten gleichzeitig geöffnet haben und bearbeiten.
- Jedes offene Dokument hat sein eigenes Fenster und ein Register mit seinem Namen am unteren Rand des Hauptfensters. Um ein Dokument zum aktiven zu machen, klicken Sie auf sein Register.
- Wenn mehrere Dateien geöffnet sind, werden manchmal auf Grund von Platzmangel in der Dokumentregisterleiste nicht alle Register angezeigt. Um Dokumentregister zu sehen, (i) verwenden Sie entweder die Scroll-Schaltflächen rechts unten in der Dokumentregisterleiste oder (ii) wählen Sie das gewünschte Dokument aus der Liste im unteren Bereich des Menüs **Fenster**[1650] aus.
- Wenn das aktive Dokument maximiert ist, befinden sich die Schaltflächen zum **Minimieren, Wiederherstellen** und **Schließen** auf der rechten Seite der Menüleiste. Wenn ein Dokument überlappend, nebeneinander oder minimiert angezeigt wird, befinden sich die Schaltflächen zum **Maximieren, Wiederherstellen** und **Schließen** in der Titelleiste des Dokumentfensters.
- Wenn Sie eine Datei maximieren, werden auch alle anderen offenen Dateien maximiert.
- Um offene Dateien überlappend oder nebeneinander anzuzeigen, verwenden Sie die Befehle im Menü Fenster[1650].
- Sie können offene Dateien auch durch Verwendung von **Strg+Tab** oder **Strg+F6** in der Reihenfolge aktivieren, in der sie geöffnet wurden.
- Durch Rechtsklick auf ein Dokumentregister wird ein Kontextmenü mit einer Auswahl an **Datei**-Befehlen wie z.B. **Drucken** und **Schließen** angezeigt.

Ansichten im Hauptfenster

Das aktive Dokument kann auf verschiedene Arten angezeigt und bearbeitet werden. Die verfügbaren Ansichten werden in einer Leiste oberhalb der Dokumentregister angezeigt (*siehe Abbildung oben*), wobei die aktive Ansicht markiert erscheint. Um eine Ansicht auszuwählen, klicken Sie auf die entsprechende Ansichtsschaltfläche oder rufen Sie im Menü **Ansicht**[1501] den entsprechenden Befehl auf.

Verfügbar sind entweder Editier- oder Browser-Ansichten:

- **Textansicht**[144]**:** Eine Editieransicht mit Syntaxfärbung für die Arbeit am Dokumentcode selbst.
- **Grid-Ansicht**[169]**:** Zur strukturierten Bearbeitung. Das Dokument wird in Tabellenform dargestellt und kann grafisch bearbeitet werden. Diese Ansicht enthält auch eine Tabellendarstellung, in der sich wiederholende Elemente in Tabellenform dargestellt werden.
- **Schema-Ansicht**[140]**:** Zum Anzeigen und Bearbeiten von XML-Schemas.

- **WSDL-Ansicht**[140]**:** Zum Anzeigen und Bearbeiten von WSDL-Dokumenten.
- **Authentic-Ansicht**[622]**:** Zum Bearbeiten von XML-Dokumenten in einer grafischen Ansicht auf Basis von StyleVision Power Stylesheets
- **Browser-Ansicht**[282]**:** Eine integrierte Browser-Ansicht, die sowohl CSS- als auch XSL-Stylesheets unterstützt.

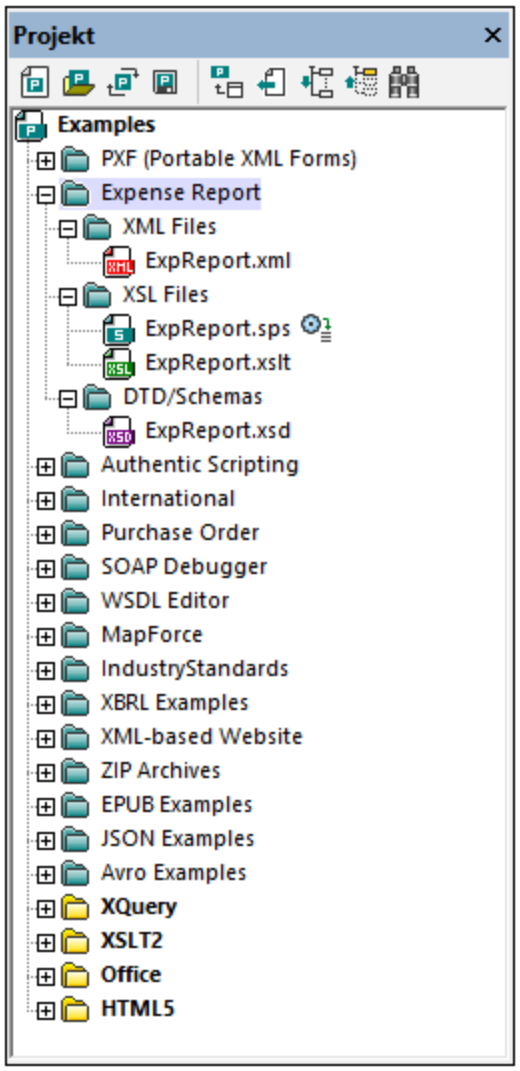
Hinweis: Sie können die Standardansicht für bestimmte Dateierweiterungen im Dialogfeld **Extras | Optionen**[1602] anpassen: Gehen Sie dazu auf dem Register "Dateiarten" zum Fenster "Standardansicht".

3.1.2 Projektfenster

Ein Projekt ist eine Sammlung von Dateien, die miteinander auf eine von Ihnen festgelegte Art in Zusammenhang stehen. In der Abbildung unten enthält z.B. ein Projekt namens `Examples` die Dateien für verschiedene Beispiele in Beispielordnern, von denen jeder weitere Unterordner enthalten kann. Im Projekt `Examples` in der Abbildung ist der Beispielordner `Expense Report` in Unterordner für XML-, XSL- und Schema-Dateien gegliedert.

Anmerkung: Im Projektfenster von XMLSpy wird standardmäßig das Projekt "Examples" geöffnet. Um das Standardprojekt "Examples" zu öffnen, gehen Sie *im Ordner (Eigene) Dokumente*[34] zum Ordner **`Examples`** der Applikation und doppelklicken Sie auf die Datei **`Examples.spp`**.

© 2016-2022 Altova GmbH



Projekte dienen also als Sammelbehälter für gemeinsam verwendete Dateien, damit diese schneller aufgerufen werden können. Des Weiteren können Sie Schemas und XSLT-Dateien für einzelne Ordner definieren, um die Batch-Verarbeitung für Dateien in einem Ordner zu ermöglichen.

Projektoperationen

Die Befehle für Ordneroperationen stehen im Menü **Projekt** zur Verfügung. Einige Befehle stehen in den Kontextmenüs (Aufruf mittels Rechtsklick) des Projekts und seinen Ordnern zur Verfügung. Ein Teil der Befehle aus dem Menü **Projekt** steht aufgrund ihrer häufigen Verwendung auch in der Symbolleiste des Projektfensters zur Verfügung (*siehe Abbildung unten*).



Die Symbolleisten-Schaltflächen sind von links nach rechts: *Neues Projekt, Projekt öffnen, Projekt neu laden, Projekt speichern, Aktive Datei zu Projekt hinzufügen, Aktive Datei auswählen, Alle erweitern, Alle reduzieren, Suchen*. Die Namen dieser Befehle sind selbsterklärend und werden im Menü "Projekt" 1316 erläutert.

In der Folge finden Sie eine Liste der wichtigsten Operationen im Zusammenhang mit dem Projektfenster.

- Im Projektfenster ist immer nur ein Projekt geöffnet. Wenn ein neues Projekt erstellt oder ein bestehendes geöffnet wird, wird das Projekt, das gerade im Projektfenster geöffnet ist, durch das neue ersetzt.
- Nachdem an einem Projekt Änderungen vorgenommen wurden, muss das Projekt (durch Auswahl des Befehls **Projekt | Projekt speichern)** gespeichert werden. Wenn ein Projekt nicht gespeicherte Änderungen enthält, so wird dies durch ein Sternchen neben dem Projektnamen angezeigt (*siehe Abbildung oben*).
- Das Projekt hat eine Baumstruktur bestehend aus Ordnern, Dateien und anderen Ressourcen. Diese können in jeder Ebene und bis zu einer beliebigen Tiefe hinzugefügt werden.
- Projektordner sind *semantische* Ordner, die eine logische Dateigruppierung darstellen und **nicht** der hierarchischen Organisation von Dateien auf Ihrer Festplatte entsprechen müssen.
- Ordner können physischen Verzeichnissen in Ihrem Dateisystem entsprechen und eine direkte Beziehung zu diesen haben. Solche Ordner werden *externe Ordner* genannt und im Projektfenster durch ein gelbes Ordnersymbol gekennzeichnet (im Gegensatz zu normalen Projektordnern, die grün angezeigt werden). Externe Projektordner müssen explizit mit dem Befehl **Aktualisieren** synchronisiert werden.
- Ein Ordner kann eine beliebige Mischung von Dateiarten enthalten. Alternativ dazu können Sie (im Dialogfeld "Eigenschaften" für diesen Ordner) für jeden Ordner Dateierweiterungen definieren, damit bestimmte Dateiarten jederzeit über einen Ordner aufgerufen werden können. Wenn eine Datei zum übergeordneten Ordner hinzugefügt wird, wird sie automatisch zu dem Unterordner hinzugefügt, für den diese Dateierweiterung definiert wurde.
- Wenn Sie die Maus über eine Bilddatei im Projektordner platzieren, wird eine Vorschau des Bilds (Formate `.png`, `.jpg`, `.gif`, `.bmp`, `.tiff` und `.ico`). angezeigt. Doppelklicken Sie auf das Bild, um es im Standard Bildansichts-/-bearbeitungsprogramm des Systems zu öffnen.
- Im Projektordner kann ein Ordner in einen anderen Ordner oder an eine andere Stelle im selben Ordner gezogen werden; eine Datei kann in einen anderen Ordner gezogen werden, nicht aber innerhalb desselben Ordners (in dem die Dateien alphabetisch geordnet sind) verschoben werden. Dateien und Ordner können auch aus dem Windows Datei-Explorer in die Projekt-Seitenleiste gezogen werden.
- Jeder Ordner hat bestimmte Eigenschaften, die im Dialogfeld "Eigenschaften" dieses Ordners definiert werden. Dazu gehören die Dateierweiterungen für diesen Ordner, das Schema, mit dem die XML-Dateien validiert werden, die XSLT-Datei, mit der die XML-Dateien transformiert werden sollen usw.
- Die Batch-Verarbeitung von Dateien in einem Ordner erfolgt durch Rechtsklick auf den Ordner und Auswahl des entsprechenden Befehls aus dem Kontextmenü (z.B. **XML validieren** oder **Wohlgeformtheit prüfen**).

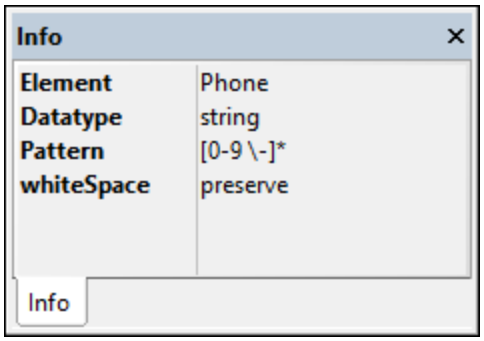
Eine nähere Beschreibung zu Projekten finden Sie im Abschnitt Projekte 1080.

Anmerkung: Das Projektfenster kann über das Menü **Fenster** ein- und ausgeblendet werden.

3.1.3 Info-Fenster

Das Fenster "Info" (*Abbildung unten*) enthält detaillierte Informationen über das Element oder Attribut, in dem sich der Cursor gerade befindet. Die Informationen stehen im Info-Fenster, in der Text- und der Grid-Ansicht, der XBRL-Ansicht und der Authentic-Ansicht zur Verfügung.

© 2016-2022 Altova GmbH



Sie können das Info-Fenster über das Menü **Fenster** ein- oder ausblenden.

Beachten Sie die folgenden Punkte:

- Wenn ein XSLT-Dokument aktiv ist, stehen auf den Register "XSLT" des Info-Fensters zusätzliche XSLT-spezifische Informationen und Befehle zur Verfügung. Eine Erklärung zu den Informationen und Befehlen auf dem Register "XSLT" finden Sie im Abschnitt XSLT und XQuery | XSLT | XSL-Übersicht[504] .
- Wenn ein JSON-Dokument (Instanz oder Schema) aktiv ist, stehen auf dem Register "JSON" des Info-Fensters Optionen zur Validierung und intelligenten Bearbeitung des Dokuments zur Verfügung. Nähere Informationen dazu finden Sie unter Validieren von JSON-Dokumenten[765].
- Wenn ein XML-Schema-Dokument (`.xsd`-Datei) aktiv ist, stehen im Fenster "Info" auf dem Register "Schema" Optionen zum Aktivieren der erweiterten Validierung zur Verfügung. Informationen zum Einrichten der erweiterten Validierung finden Sie im Abschnitt Schemaregeln[456].
- Wenn ein XULE-Dokument aktiv ist, steht auf dem Register "XULE" des Info-Fensters eine Option zur Auswahl einer XBRL-Taxonomie für Informationen über die Struktur einer XBRL-Instanz zur Verfügung. Anhand der Taxonomieinformationen werden Vorschläge für die Autokomplettierungsfunktion des XULE-Editors[939] gemacht.

3.1.4 Eingabehilfen

Eingabehilfen sind intelligente Editierfunktionen, mit Hilfe derer Sie schnell gültige XML-Dokumente erstellen können. Wenn Sie ein Dokument bearbeiten, sehen Sie in den Eingabehilfen je nach Cursorposition Bearbeitungsoptionen zur Struktur. Die Eingabehilfen beziehen die erforderlichen Informationen aus der zugrunde liegenden DTD, dem XML-Schema und/oder dem StyleVision Power Stylesheet usw. Wenn Sie z.B. ein XML-Datendokument bearbeiten, werden in den entsprechenden Eingabehilfenfenstern die Elemente, Attribute und Entities angezeigt (sowie dazugehörige Informationen), die an der aktuellen Cursorposition eingefügt werden können.

Welche Eingabehilfen angezeigt werden, ist abhängig von den folgenden Faktoren:

1. *der Art des bearbeiteten Dokuments.* So haben XML-Dokumente z.B. andere Eingabehilfen als XQuery-Dokumente: Im ersteren stehen Elemente, Attribute und Entities-Eingabehilfen zur Verfügung, in XQuery Eingabehilfen für XQuery-Schlüsselwörter, Variablen und Funktionen. Die für die einzelnen Dokumenttypen verfügbaren Eingabehilfen werden in dieser Dokumentation unter der Beschreibung zum jeweiligen Dokumenttyp näher behandelt.
2. *der aktuellen Ansicht*. Da die Bearbeitungsweise in den verschiedenen Ansichten unterschiedlich ist, sind die Eingabehilfen so konzipiert, dass sie mit der Bearbeitungsweise in der aktuellen Ansicht

kompatibel sind. So kann z.B. in der Textansicht ein Element nur an der Cursorposition eingefügt werden. Daher ist die Eingabehilfe so konzipiert, dass ein Element eingefügt wird, wenn Sie darauf doppelklicken. In der Grid-Ansicht dagegen kann ein Element vor dem ausgewählten Node eingefügt, dahinter angehängt oder als Child Node hinzugefügt werden, daher hat die Element-Eingabehilfe in der Grid-Ansicht drei Register für das *Einfügen, Anhängen* und *Hinzufügen als Child*, wobei jedes Register die Elemente enthält, die für diese Operation zur Verfügung stehen.

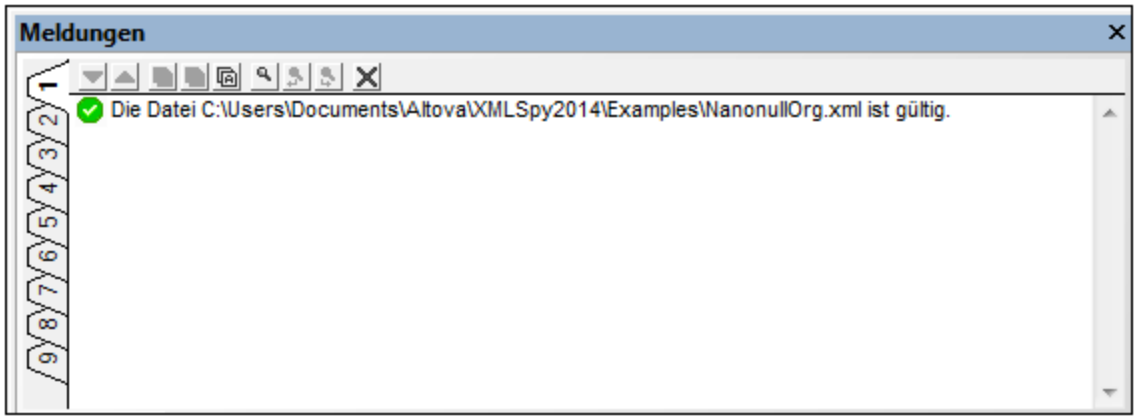
Eine Beschreibung der Eingabehilfen der einzelnen Ansichten finden Sie unter Bearbeitungsansichten[140]. Weitere Unterschiede im Zusammenhang mit Dokumentarten in einer Ansicht finden Sie in den Abschnitten über die einzelnen Dokumentarten, z.B. XML-Eingabehilfen[337] und XQuery-Eingabehilfen[518].

Hinweis:

- Über die Menüoption **Fenster | Eingabehilfen** können Sie die Eingabehilfen ein- oder ausblenden.
- In Visual Studio .NET haben die Eingabehilfefenster ein Präfix, das der Applikationsname ist.

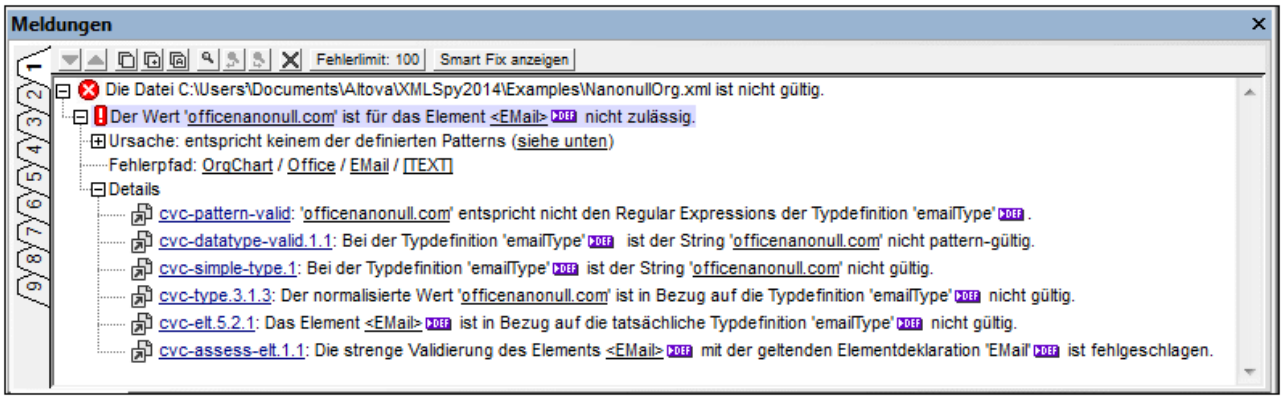
3.1.5 Ausgabefenster "Meldungen"

Im Fenster "Meldungen" werden Meldungen über in XMLSpy ausgeführte Aktionen sowie Fehlermeldungen und andere Ausgabemeldungen angezeigt. Wenn z.B. ein XML-, XML-Schema, DTD- oder XQuery-Dokument validiert wird und gültig ist, wird im Validierungsfenster eine entsprechende Meldung (*Abbildung unten*) angezeigt.



Andernfalls wird eine Meldung, die den Fehler beschreibt, angezeigt (*Abbildung unten*). Beachten Sie in der Abbildung unten, dass es Links (schwarzen Text) zu Nodes und Node-Inhalt im XML-Dokument gibt sowie Links zum entsprechenden Abschnitt in der Spezifikation im Internet, in dem die betreffende Regel beschrieben wird (blauer Text). Wenn Sie auf eines der violetten Dev-Symbole klicken, wird die entsprechende Schemadefinition in der Schema-Ansicht geöffnet.

© 2016-2022 Altova GmbH

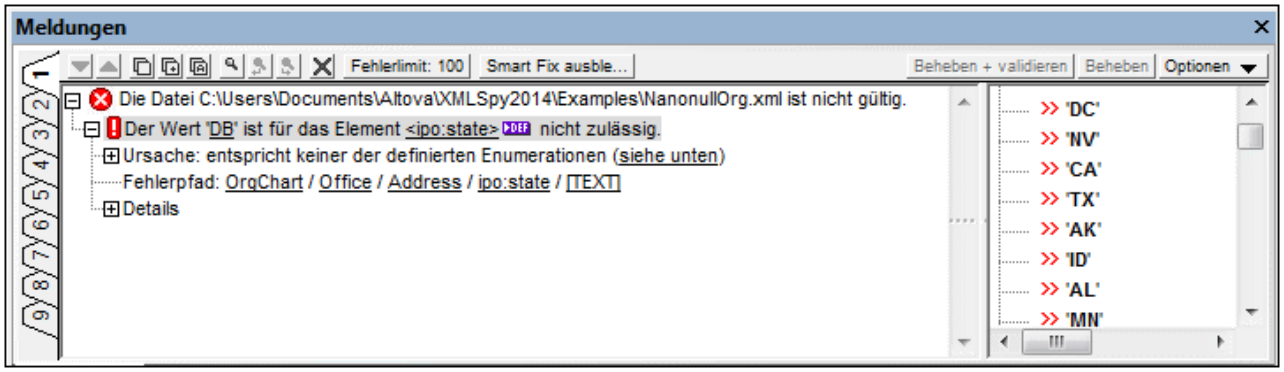


Das Fenster "Meldungen" ist in allen Ansichten aktiv. Wenn Sie jedoch auf einen Link zum XML-Dokumentinhalt klicken, wird der entsprechende Node in der Text-Ansicht markiert. Beachten Sie jedoch: Wenn ein XML-Schema in der Schema-Ansicht validiert wird, ändert sich die Ansicht bei Klicken auf das `Def`-Symbol nicht.

Schnelllbehebung von XML-Validierungsfehlern mit Smart Fix

Auf Basis der im Schema enthaltenen Informationen werden bei Ausführung der Validierung in der Text- oder Grid-Ansicht Smart Fix-Optionen vorgeschlagen. Um eine Liste von Smart Fix-Optionen anzuzeigen, klicken Sie auf die Schaltfläche **Smart Fix anzeigen** (*siehe Abbildung oben*). Daraufhin wird im Fenster "Meldungen" ein Fenster mit den vorgeschlagenen Smart Fix-Optionen angezeigt (*Abbildung unten*).

Beachten Sie, dass immer nur ein Fehler im Fenster "Meldungen" angezeigt wird, d.h. dass die Fehler der Reihe nach angezeigt werden. Außerdem werden Wohlgeformtheitsfehler (wie z.B. fehlende Start- und End-Tags), falls vorhanden, vor den Validierungsfehlern angezeigt. Daher wird die Schaltfläche **Smart Fix anzeigen** erst dann aktiviert, wenn die Validierungsfehler erreicht werden (nachdem alle Wohlgeformtheitsfehler behoben wurden).



Wählen Sie im Fenster "Smart Fix" einen der vorgeschlagenen Smart Fixes aus und klicken Sie entweder auf die Schaltfläche **Beheben + validieren** oder auf die Schaltfläche **Beheben** (*siehe Abbildung oben*). Der ungültige Abschnitt im XML-Dokument wird daraufhin durch den ausgewählten Smart Fix ersetzt. Alternativ dazu können Sie auch auf den gewünschten Smart Fix-Vorschlag doppelklicken. Je nachdem, welche Option in der Dropdown-Liste *Optionen* ausgewählt ist, wird der Fehler damit behoben oder behoben und validiert. Der Befehl **Beheben + validieren** ist nützlich, da nach Beheben des Fehlers und Validierung des Dokuments weitere Validierungsfehler angezeigt werden, falls vorhanden.

Um das Smart Fix-Fenster auszublenden, klicken Sie auf die Schaltfläche **Smart Fix ausblenden** (*siehe Abbildung oben*). Nähere Informationen dazu finden Sie im Abschnitt Bearbeitungsansichten | Schema-Ansicht | Validierung und Smart Fixes 239.

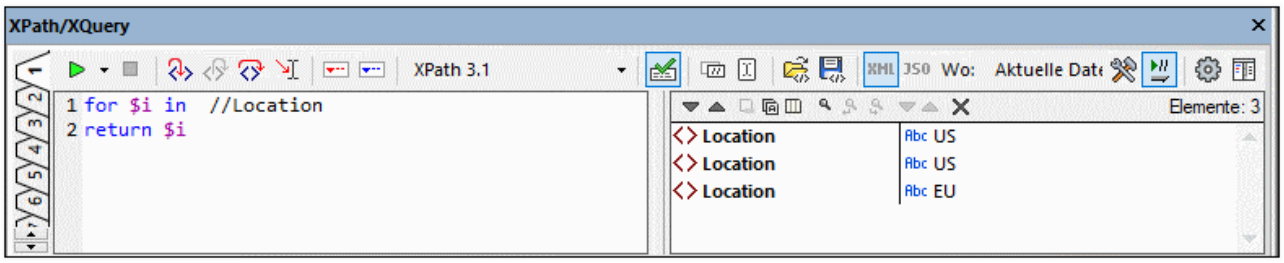
Validieren von Ordnern und Dateien im Projektfenster

Der Befehl **Validieren** (im Menü XML) wird normalerweise auf das aktive Dokument angewendet. Sie können den Befehl jedoch auch auf eine Datei oder eine Gruppe von Dateien im aktiven Projekt anwenden. Wählen Sie die gewünschte Datei bzw. den gewünschten Ordner im Projekt-Fenster (durch Anklicken) aus und klicken Sie auf **XML | Validieren** 1353 oder **F8**. Ungültige Dateien in einem Projekt werden geöffnet und im Hauptfenster aktiviert und die Fehlermeldung "*Die Datei ist nicht gültig*" wird angezeigt.

Anmerkung: Sie können die Wohlgeformtheitsprüfung auch im Projektfenster durchführen (**Wohlgeformtheit prüfen** 1351 oder **F7**).

3.1.6 Ausgabefenster: "XPath/XQuery"

Im Fenster "XPath/XQuery" (*Abbildung unten*) können Sie XPath- und XQuery-Ausdrücke anhand von XML- oder JSON-Dokumenten erstellen, auswerten und debuggen. (In XPath/XQuery 3.1 wurden Funktionen, die auch JSON-Abfragen ermöglichen, eingeführt. *Siehe JSON-Transformationen mit XSLT/XQuery* 769*)*



Dieser Abschnitt enthält eine kurze Übersicht über die wichtigsten Funktionen des Fensters "XPath/XQuery". Eine ausführliche Anleitung zum Arbeiten mit dem Fenster "XPath/XQuery" finden Sie im Abschnitt XPath/XQuery-Ausdrücke 582.

Wichtigste Funktionen

Über das Fenster "XPath/XQuery" stehen Ihnen die folgenden wichtigen Funktionen zur Verfügung:

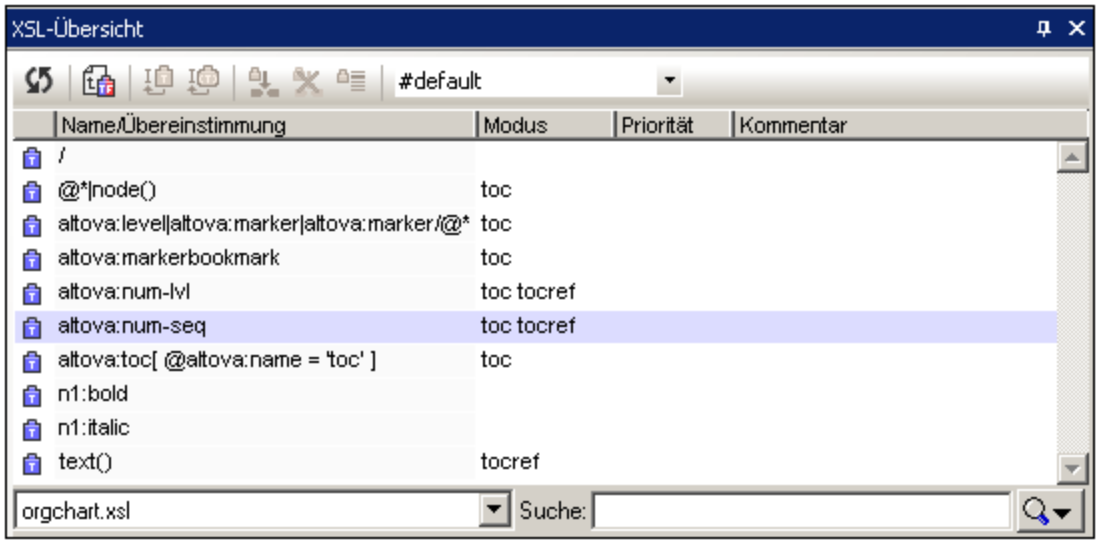
- *Auswertungsmodus:* Im Auswertungsmodus wird ein XPath- oder XQuery-Ausdruck anhand eines oder mehrerer XML/JSON-Dokumente ausgewertet. Der Ausdruck wird in das Ausdrucksfenster eingegeben und das Ergebnis wird im daneben liegenden Fenster "Ergebnisse" angezeigt. Wenn Sie auf Nodes im Ergebnis klicken, gelangen Sie zu diesem Node im XML- oder JSON-Dokument. Nähere Informationen dazu finden Sie im Abschnitt XPath/XQuery-Ausdrücke 582.
- *Debug-Modus:* Im Debug-Modus können Sie einen XPath/XQuery-Ausdruck während der Anwendung am gerade aktiven XML-Dokument debuggen. Sie können Breakpoints und Tracepoints setzen und sich Schritt für Schritt durch die Auswertung bewegen. Bei jedem Schritt sehen Sie den Inhalt der Variablen. Außerdem können Sie benutzerdefinierte Überwachungsausdrücke festlegen, um zusätzliche Aspekte der Auswertung zu überprüfen. Nähere Informationen dazu finden Sie im Abschnitt Debuggen des Ausdrucks 591.

- *Ausdruckerstellung:* Wenn Sie einen Ausdruck erstellen, stehen Ihnen Eingabehilfen und Popup-Fenster mit Informationen zur Verfügung, die Ihnen bei der Erstellung syntaktisch korrekter Ausdrücke helfen. Nähere Informationen dazu finden Sie im Abschnitt Ausdruckserstellung[598].
- *Unterstützung für mehrere Sprachen:* Sie können zwischen verschiedenen Versionen von XPath 1.0 bis XPath 3.1 und XQuery 3.1 wechseln. Der eingegebene Ausdruck wird gemäß den Regeln der ausgewählten Sprache ausgewertet.
- *Öffnen von Ausdrücken aus einer Datei und Speichern von Ausdrücken in einer Datei:* Sie können einen XPath/XQuery-Ausdruck zusammen mit den aktuellen Einstellungen des Fensters in einer XQuery-Datei speichern und Ausdrücke aus einer XQuery-Datei laden.
- *Automatische Erkennung des Dateityps (XML/JSON):* Der Dateityp des aktuellen Dokuments (XML oder JSON) wird automatisch erkannt und der korrekte Zieldokument-Modus wird automatisch ausgewählt. Wenn der Geltungsbereich eine Gruppe von Dokumenten umfasst, können Sie den Zieldokumenttyp manuell auswählen.
- *Flexibler Geltungsbereich für Zieldokumente:* Im Feld *Wo* können Sie auswählen, ob der Ausdruck an der gerade aktiven Datei, an allen offenen Dateien, am aktuellen Projekt oder einem Ordner getestet werden soll.
- *Optionen für das Ergebnis- und das Überwachungsfenster:* Sie können festlegen, wie Einträge im Ergebnisfenster und im Überwachungsausdruckfenster[591] angezeigt werden sollen.
- *Ergebnisse enthalten direkte Links zu Dokumenten und Dokument-Nodes:* Die Zeilen im Ergebnisfenster enthalten Links zu den jeweiligen Dokumenten oder Dokument-Nodes. Auf diese Art gelangen Sie direkt zu bestimmten Nodes und können die Daten dort überprüfen.
- *Debugger-Analysefunktionen:* Im Debug-Modus wird eine breite Palette von Analyseinformationen angezeigt. Zusätzlich dazu können Sie benutzerdefinierte Ausdrücke definieren, um weitere Aspekte der Auswertung zu überprüfen.

Eine ausführliche Beschreibung der Funktionen des Fensters "XPath/XQuery" finden Sie im Abschnitt XPath/XQuery-Ausdrücke[582], in dem alle in der Symbolleiste verfügbaren Modi, Symbole und Funktionalitäten beschrieben sind.

3.1.7 Ausgabefenster "XSL-Übersicht"

Im Fenster "XSL-Übersicht" (*Abbildung unten*) werden alle Vorlagen und Funktionen in einem XSLT Stylesheet und optional dazu auch in allen inkludierten und importierten XSLT Stylesheets aufgelistet. Das Fenster "XSL-Übersicht" ist standardmäßig an das Ausgabefenster um unteren Rand des XMLSpy-Fensters angedockt. Es kann abgedockt oder an einer andere Kante des XMLSpy-Fensters angedockt werden.



Das Fenster "XSL-Übersicht" enthält Informationen über Vorlagen und Funktionen im Stylesheet. Die Informationen können sortiert und durchsucht werden und die Symbolleiste des Fensters enthält Befehle, mit denen Sie jederzeit Aufrufe von Named Templates einfügen und Named Templates als Ausgangspunkt von Transformationen definieren können. Eine Beschreibung der Verwendung des Fensters "XSL-Übersicht" finden Sie im Abschnitt XSLT und XQ | XSLT | XSL-Übersicht 504.

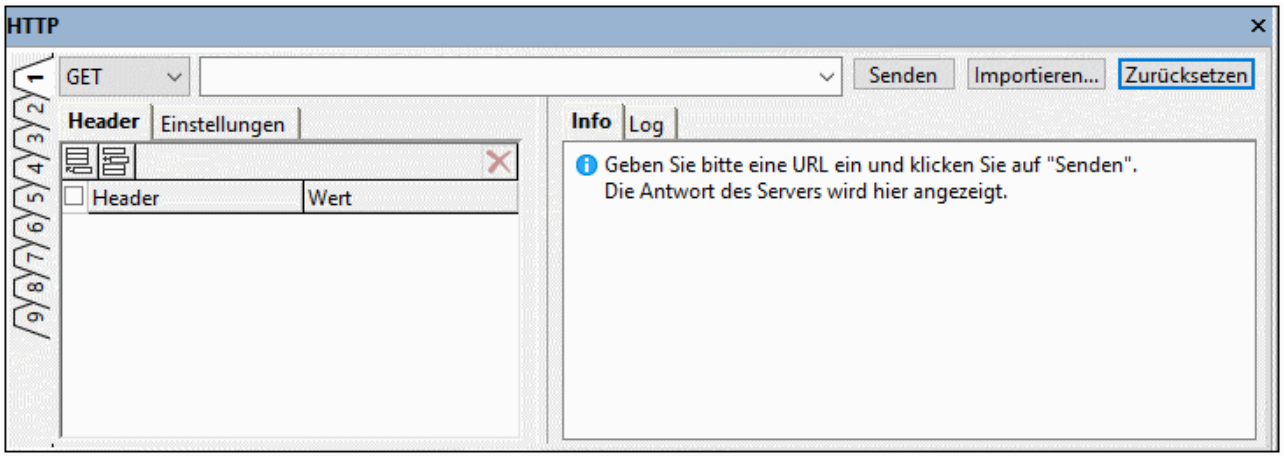
Anmerkung: Dateibezogene Informationen über das Stylesheet sowie dateibezogene Befehle finden Sie auf dem Register "XSLT" des Info-Fensters. Eine Beschreibung dazu finden Sie im Abschnitt XSLT und XQ | XSLT | XSL-Übersicht 504.

3.1.8 Ausgabefenster: HTTP

Im HTTP-Ausgabefenster (*Abbildung unten*) können Sie HTTP-Befehle testen: Sie können einen HTTP-Request erstellen und an einen Webserver senden und die Antwort empfangen und überprüfen.

Das HTTP-Ausgabefenster hat neun Register (*siehe Abbildung unten*). Sie können auf jedem Register einen separaten Request speichern und zwischen den Registern wechseln. Nach Erstellung eines Request können Sie diesen durch Klick auf die Schaltfläche **Senden** senden. Die Antwort (Response) wird direkt im Fenster angezeigt.

© 2016-2022 Altova GmbH



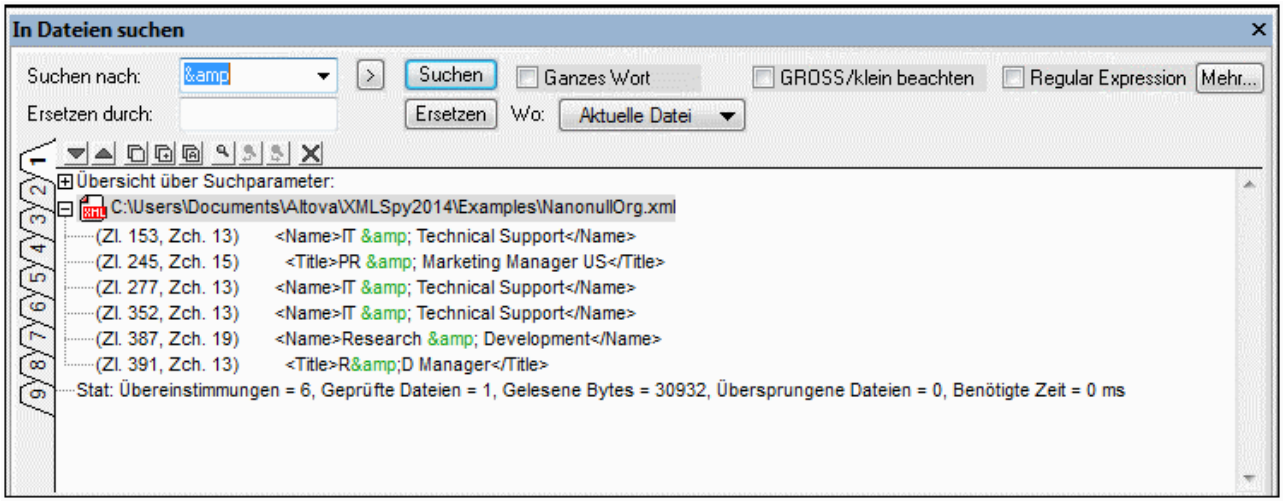
Das Fenster besteht aus den folgenden Teilen:

- Im oberen Bereich finden Sie: (i) eine Auswahlliste, in der Sie die gewünschte HTTP-Methode auswählen können; (ii) ein Eingabefeld für die URL des Webservers; (iii) Schaltflächen im Zusammenhang mit der Ausführung von HTTP Requests (**Senden**, **Importieren** und **Zurücksetzen**).
- Einem Bereich auf der linken Seite zum Erstellen des Request. 818
- Einem Bereich auf der rechten Seite zur Anzeige der Response 825.

Nähere Informationen zur Verwendung des HTTP-Ausgabefensters finden Sie im Abschnitt HTTP 817.

3.1.9 Ausgabefenster "In Dateien suchen"

Im Ausgabefenster "In Dateien suchen" (*Abbildung unten*) können Sie Such- und Ersetzungsvorgänge schnell in mehreren Dokumenten gleichzeitig durchführen. Es bietet Funktionen zum raschen Navigieren in den gefundenen Instanzen. Die Ergebnisse der einzelnen Such- und Ersetzungsvorgänge werden auf einem der von 1 bis 9 durchnummerierten Register angezeigt. Wenn Sie in der Ergebnisliste auf eine der gefundenen Instanzen klicken, gelangen Sie in der Text-Ansicht des Dokuments zu dieser Instanz.



Suchkriterien

Es gibt zwei grundlegende Suchkriterien: (i) Was soll gesucht werden und (ii) wo soll gesucht werden?

Was soll gesucht werden: Der zu suchende String wird in das Textfeld "Suchen nach" eingegeben. Wenn mit diesem String ein ganzes Word gefunden werden soll, muss das Kontrollkästchen "Ganzes Wort" aktiviert werden. Um z.B. den String `fit` zu finden wird bei aktivierten Kontrollkästchen "Ganzes Wort" nur das Wort `fit` gefunden; `fit` in `fitness` würde z.B. nicht gefunden. Mit Hilfe des Kontrollkästchens "GROSS/klein beachten" können Sie angeben, ob die Groß- und Kleinschreibung beachtet werden soll. Wenn es sich bei dem im Textfeld "Suchen nach" eingegebenen Text um eine Regular Expression handelt, muss das Kontrollkästchen "Regular Expression" aktiviert werden. Um eine Eingabehilfe zur Eingabe von Regular Expression-Zeichen aufzurufen, klicken Sie auf die Schaltfläche . Die Verwendung von Regular Expressions zur Suche wird unter Suchen[1307] erklärt. Über die Schaltfläche **Mehr** wird das Dialogfeld In Dateien suchen[1312] geöffnet. Hier können Sie weitere Suchbedingungen und Aktionen definieren. Nähere Informationen finden Sie unter Bearbeiten | In Dateien suchen[1312].

Wo soll gesucht werden: Die Suche kann in folgenden Dateien durchgeführt werden: (i) in allen auf der Benutzeroberfläche geöffneten Dateien; (ii) in den Dateien des aktuellen Projekts; und (iii) in den Dateien eines ausgewählten Ordners. Im Dialogfeld "In Dateien suchen"[1312] (das durch Klicken auf **Mehr** aufgerufen wird) können Sie zusätzliche Bedingungen festlegen.

Ersetzen durch

Der String, durch den der gefundene String ersetzt werden soll, wird in das Textfeld "Ersetzen durch" eingegeben. Beachten Sie: Wenn das Textfeld "Ersetzen durch" leer ist und Sie auf die Schaltfläche "Ersetzen" klicken, wird der gefundene Text durch einen leeren String ersetzt.

Die Ergebnisse

Nachdem Sie auf die Such- oder Ersetzungsschaltflächen geklickt haben, werden die Ergebnisse des Such- bzw. Ersetzungsvorgangs im Ausgabefenster "In Dateien suchen" angezeigt. Die Ergebnisse sind in vier Teile untergliedert:

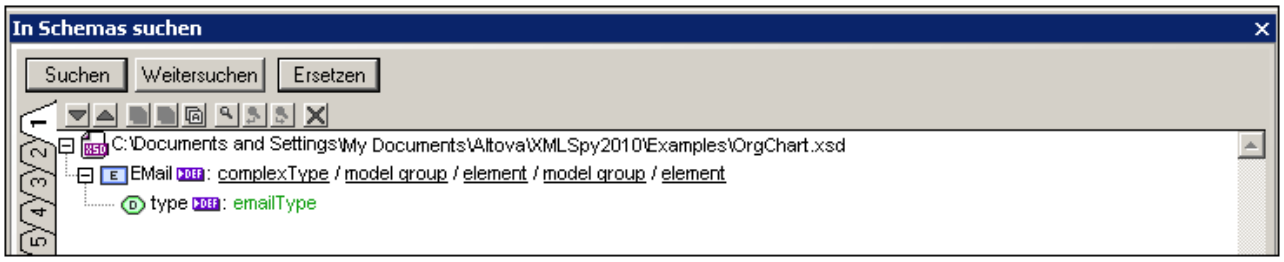
- Eine Zusammenfassung der Suchparameter, in der aufgelistet ist, welcher Suchstring gesucht wurde und welche Dateien durchsucht wurden.
- Eine Liste der gefundenen bzw. ersetzten Strings (je nachdem, ob Sie die Schaltfläche **Suchen** oder **Ersetzen** betätigt haben). Die Einträge in dieser Liste sind Links zum gefundenen/ersetzten Text in der Text-Ansicht des Dokuments. Wenn das Dokument nicht geöffnet ist, wird es in der Text-Ansicht geöffnet und der gefundene/ersetzte Text wird markiert.
- Eine Liste der Dateien, die durchsucht wurden, in denen jedoch keine Instanzen gefunden wurden.
- Eine Zusammenfassung der Statistiken für die Suchaktion einschließlich der Anzahl der Treffer und der überprüften Dateien.

Anmerkung: Beachten Sie: Mit der Funktion "In Dateien suchen" werden die Befehle zum Suchen und Ersetzen gleichzeitig an mehreren Dateien durchgeführt. Die Ergebnisse werden im Ausgabefenster "In Dateien suchen" angezeigt. Um eine Suche durchzuführen, bei der Sie von einer gefundenen Instanz zur nächsten springen, verwenden Sie den Befehl Suchen[1307].

© 2016-2022 Altova GmbH

3.1.10 Ausgabefenster "In Schemas suchen"

Wenn in der Schema-Ansicht ein XML-Schema aktiv ist, kann es mit der "Suchen & Ersetzen" Schema-Ansichtsfunktion von XMLSpy auf intelligente Weise durchsucht werden. Die Funktion "Suchen und Ersetzen" wird auf folgende Weise aufgerufen: (i) über die Befehle **Suchen** und **Ersetzen** aus dem Menü **Bearbeiten** und (ii) über die Schaltflächen **Suchen** und **Ersetzen** im Fenster "In Schemas suchen" (*Abbildung unten*).



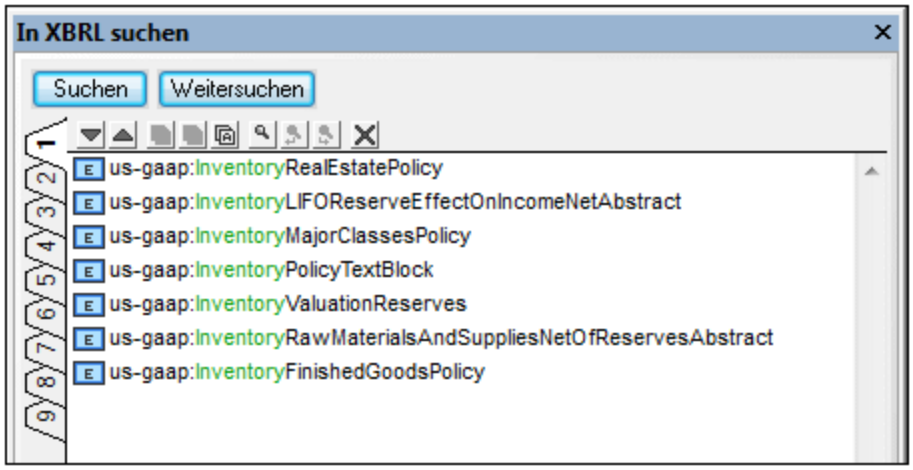
Die Ergebnisse der Such- und Ersetzungsfunktion in der Schema-Ansicht werden im Fenster "In Schemas suchen" angezeigt (jedes Mal, wenn der Befehl "Suchen" oder "Ersetzen" ausgeführt wird). Der Suchbegriff wird grün angezeigt; (in der Abbildung oben sehen Sie, dass der Suchbegriff `email` war, wobei die Groß- und Kleinschreibung ignoriert wurde). Beachten Sie, dass auch der Pfad der Schema-Datei angegeben wird.

Die Ergebnisse werden auf neun separaten Registern (mit einer Nummer von 1 bis 9) angezeigt. Dadurch können sie die Ergebnisse auf einem Register beibehalten, auf einem neuen Register eine neue Suche durchführen und die Ergebnisse miteinander vergleichen. Um die Ergebnisse einer neuen Suche auf einem neuen Register anzuzeigen, wählen Sie es vor Beginn der Suche aus. Wenn Sie im Fenster "In Schemas suchen" auf ein Ergebnis klicken, wird die entsprechende Komponente im Hauptfenster der Schemaansicht angezeigt und markiert. Auf diese Art können Sie schnell zur gewünschten Komponente navigieren und Meldungen in die Zwischenablage kopieren. Nähere Informationen dazu finden Sie unter der Beschreibung der Funktion In Schemas suchen[482] im Abschnitt Ergebnisse & Informationen[494].

3.1.11 Ausgabefenster "In XBRL suchen"

Im Fenster "In XBRL suchen" (*Abbildung unten*) werden die Ergebnisse der Suche in einem XBRL-Taxonomiedokument angezeigt. Das Fenster enthält neun Registerkarten, sodass die Ergebnisse auf einem Register mit denen auf einem anderen Register verglichen werden können.

Die Suche in XBRL-Dokumenten kann durchgeführt werden, wenn ein XBRL-Taxonomiedokument in der XBRL-Ansicht geöffnet ist. Eine Anleitung zur Suche finden Sie im Abschnitt In XBRL suchen[947] im Abschnitt "XBRL"[829] dieser Dokumentation.



Über die Schaltfläche "Suchen" wird das Suchdialogfeld aufgerufen. Wenn Sie auf die Schaltfläche "Weitersuchen" klicken, wird die nächste Instanz des Suchbegriffs im Dokument - beginnend von der Zelle unmittelbar nach der Zelle in der XBRL-Ansicht, in der der Cursor sich gerade befindet - gesucht.

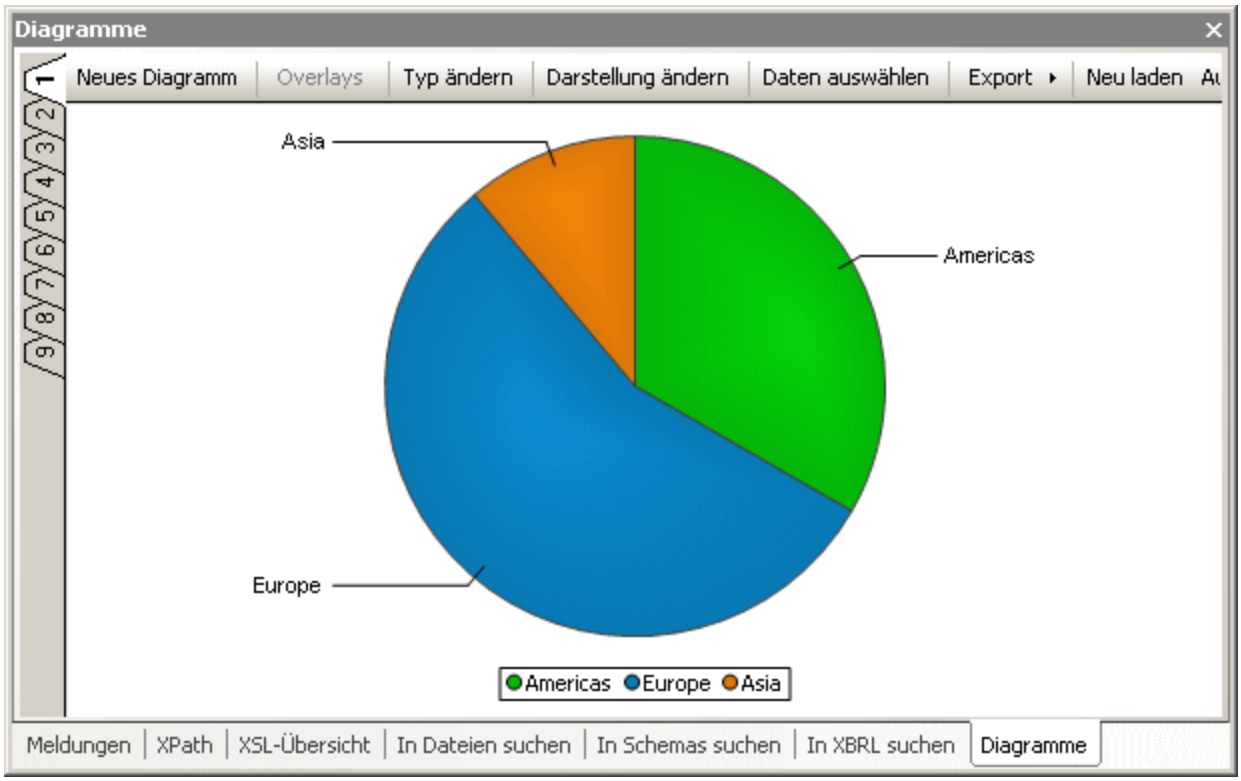
In der Symbolleiste "In XBRL suchen" stehen die folgenden Befehle zur Verfügung:

- Mit den Schaltflächen **Nächstes** und **Vorheriges** wird das nächste bzw. vorherige Suchergebnis nach bzw. vor dem gerade ausgewählten Ergebnis ausgewählt.
- Mit den Befehlen zum **Kopieren von Meldungen** kann bzw. können die ausgewählte Meldung, bzw. die ausgewählte Meldung und ihre untergeordneten Einträge bzw. alle Meldungen in die Zwischenablage kopiert werden.
- Mit dem Befehl **Suchen** werden Textstrings im Fenster "Suchen in XBRL" gesucht.
- Mit dem Befehl **Löschen** werden alle Meldungen auf dem gerade aktiven Register gelöscht.

3.1.12 Ausgabefenster "Diagramme"

Wenn ein XML-Dokument in der Text- oder Grid-Ansicht geöffnet ist, kann im Diagrammfenster ein Diagramm (Kreisdiagramm, Balkendiagramm, usw.) generiert werden, in dem ausgewählte Daten aus dem XML-Dokument dargestellt werden (Abbildung unten).

© 2016-2022 Altova GmbH



Erstellung des Diagramms

Die grundlegenden Schritte zur Erstellung eines Diagramms sind die folgenden:

1. Platzieren Sie den Cursor in das XML-Dokument um einen Kontext-Node auszuwählen.
2. Klicken Sie im Fenster "Diagramme" auf die Schaltfläche **Neues Diagramm** (*siehe Abbildung oben*) oder klicken Sie mit der rechten Maustaste in das Hauptfenster und wählen Sie im Kontextmenü den Befehl **Neues Diagramm**.
3. Wählen Sie im Dialogfeld "Spalten auswählen" die Daten für die Diagrammdatentabelle aus und klicken Sie auf **OK**. Das Diagramm wird im Fenster "Diagramme" erstellt (*Abbildung oben).*

Nähere Informationen dazu finden Sie im Benutzerhandbuch im Abschnitt "XML"[288] unter Diagramme[350].

Ändern und Verwalten von Diagrammen

Ein Diagramm kann auf jedem beliebigen der neun Diagrammfensterregister (die auf der linke Seite des Fensters nummeriert sind) erstellt werden. Auf diese Art können Diagramme auf unterschiedlichen Registern miteinander verglichen werden. Ein Diagramm, das auf einem Register erstellt wurde, kann nur überschrieben werden, wenn auf diesem Register ein neues Diagramm erstellt wird. Ein Diagramm kann anders nicht gelöscht werden. Das Diagramm bleibt selbst dann auf dem Register, auf dem es erstellt wurde, erhalten, wenn das dafür verwendete XML-Dokument geschlossen wurde.

Die Schaltflächen am oberen Rand des Fensters führen folgende Operationen aus:

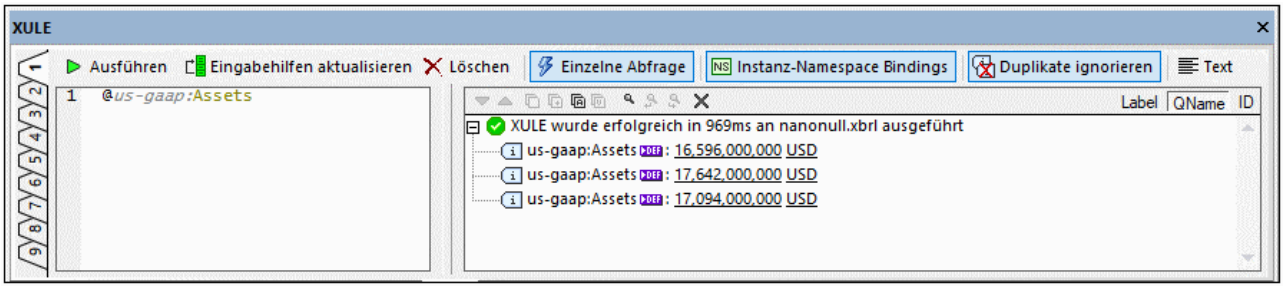
- *Neues Diagramm*: Ruft das Dialogfeld "Spalten auswählen" auf, in dem die Diagrammdatentabelle konfiguriert wird.

- *Overlays:* Dient zum Hinzufügen und Löschen von Ebenen über dem Hauptdiagramm. Nachdem Sie eine neue Ebene erstellt haben, können Sie durch Klick auf das Register für die neue Ebene und Definition der für diese Overlay-Ebene zu verwendenden Daten ein neues Diagramm zu dieser Ebene hinzufügen.
- *Typ ändern*: Dient zum Ändern des Diagrammtyps, z.B. von einem Balken- in ein Kreisdiagramm.
- *Darstellung ändern*: Dient zum Ändern von Einstellungen wie Schriftgröße und Farbschema für das Diagramm.
- *Daten auswählen*: Ruft das Dialogfeld "Daten auswählen" auf, das die Diagrammdatentabelle und die endgültige Auswahl der Daten, die im Diagramm dargestellt werden, enthält. Die Daten für die Datenreihe, die X- und die Y-Achse können in diesem Dialogfeld geändert werden. Die Daten für die X- und die Y-Achse können grafisch über die Diagrammdatentabelle ausgewählt werden. Wenn Sie auf **OK** klicken, wird das geänderte Diagramm im Fenster "Diagramme" generiert.
- *Export*: Das Diagramm kann als Bilddatei, als XSLT- oder XQuery-Fragment in die Zwischenablage kopiert werden. Das XSLT- oder XQuery-Fragment kann in einem XSLT- bzw. XQuery-Dokument verwendet werden, das das Diagramm bei Verarbeitung mit dem Altova XSLT 2.0- oder dem Altova XQuery-Prozessor korrekt darstellt.
- *Neu laden/Automatisch*: Wenn die Schaltfläche **Automatisch** aktiv ist, werden alle Änderungen am zugrunde liegenden XML-Dokument in dem Diagramm, das auf diesem XML-Dokument basiert, automatisch aktualisiert. Andernfalls wird ein Diagramm nur dann aktualisiert, wenn Sie auf die Schaltfläche **Neu laden** klicken.

Nähere Informationen dazu finden Sie im XML-Abschnitt[288] dieses Benutzerhandbuchs im Abschnitt Diagramme[350].

3.1.13 Ausgabefenster: XULE

Über das Fenster "XULE" können Sie das aktive XBRL-Instanzdokument interaktiv abfragen und die Ergebnisse Ihrer Abfrage anzeigen. Das Fenster "XULE" verfügt über neun Register. Jedes davon ist in zwei Bereiche gegliedert: (i) einen Bereich für den XULE-Ausdruck, in den Sie den gewünschten XULE-Ausdruck (oder die XULE-Regel) eingeben und (ii) einen Ergebnisbereich, in dem das Ergebnis der Ausführung angezeigt wird.



Um einen XULE-Ausdruck am aktiven XBRL-Instanzdokument interaktiv auszuführen, gehen Sie folgendermaßen vor:

1. Machen Sie das XBRL-Instanzdokument, das Sie abfragen möchten, im Hauptfenster zum aktiven Dokument.
2. Geben Sie den XULE-Ausdruck in den XULE-Ausdrucksbereich (*linke Seite*) ein.
3. Klicken Sie in der Symbolleiste des Fensters auf **Ausführen**, um den Ausdruck auszuführen.
4. Die Ausführungsergebnisse werden im Ergebnisbereich (*rechte Seite*) angezeigt.

© 2016-2022 Altova GmbH

Nähere Informationen dazu finden Sie im Abschnitt XBRL unter der Beschreibung zum Fenster "XULE" 941.

3.1.14 Menüleiste, Symbolleisten und Statusleiste

Menüleiste

Die Menüleiste (*siehe Abbildung* 115) enthält die verschiedenen Menüs der Applikation. Es gelten die folgenden Konventionen:

- Wenn Befehle in einem Menü in einer Ansicht oder an einer bestimmten Stelle im Dokument **nicht** zur Verfügung stehen, sind sie ausgegraut und deaktiviert.
- Bei Auswahl einiger Menübefehle erscheint ein Untermenü mit einer Liste zusätzlicher Optionen. Solche Menübefehle weisen rechts neben dem Befehlsnamen einen nach rechts weisenden Pfeil auf.
- Bei einigen Menübefehlen wird ein Dialogfeld aufgerufen, in dem Sie weitere Informationen eingeben müssen, damit der Befehl ausgeführt werden kann. Diese Befehle sind durch drei Auslassungspunkte (...) nach dem Namen des Befehls gekennzeichnet.
- Um einen Menübefehl aufzurufen, klicken Sie auf den Menünamen und anschließend auf den Befehl. Enthält ein Menüeintrag ein Untermenü, wird dieses geöffnet, wenn Sie die Maus über den Menüeintrag positionieren. Klicken Sie auf den gewünschten Untermenüeintrag.
- Sie können ein Menü auch durch Drücken der entsprechenden Tastenkombination öffnen. Die Tastenkombination für die einzelnen Menüs lautet **Alt+TASTE**, wobei TASTE für den unterstrichenen Buchstaben im Menünamen steht. Die Tastenkombination für das Menü **Datei** ist z.B. **Alt+a**.
- Die Auswahl eines Menübefehls (also eines Befehls in einem Menü) erfolgt durch das nacheinander-Auswählen (i) des Menüs durch die entsprechende Tastenkombination (siehe voriger Punkt) und (ii) anschließendes Drücken der Tastenkombination für den jeweiligen Befehl (**Alt+TASTE**, wobei TASTE dem unterstrichenen Buchstaben im Befehlsnamen entspricht). Um z.B. eine neue Datei zu erstellen, (**Datei | Neu**), drücken Sie **Alt+a** und dann **Alt+N**.
- Einige Menübefehle können **direkt** durch Drücken eines speziellen **Tastenkürzels** oder einer Tastenkombination (**Strg+TASTE**) aufgerufen werden. Bei Befehlen, denen ein Tastenkürzel zugewiesen wurde, wird rechts vom Befehl das Tastenkürzel oder die Tastenkombination angezeigt. So können Sie z.B. eine neue Datei mit der Tastenkombination **Strg+N** erstellen und eine XML-Datei durch Drücken von **F8** validieren. Im Dialogfeld "Anpassen" (**Extras | Anpassen**) können Sie auf dem Register "Tastatur" Ihre eigenen Tastenkürzel erstellen 1589.

Symbolleisten

Die Symbolleisten (*siehe Abbildung* 115) enthalten Schaltflächen zum Aufrufen von Menübefehlen. Wenn Sie den Mauszeiger über die Schaltfläche halten, wird der Name des Befehls angezeigt. Um den Befehl auszuführen, klicken Sie auf die Schaltfläche.

Die Symbolleisten-Schaltflächen sind in Gruppen angeordnet. Im Dialogfeld **Extras | Anpassen | Symbolleisten** 1586 können Sie einstellen, welche Symbolleistengruppen angezeigt werden sollen. Diese Einstellungen gelten für die aktuelle Ansicht. Um Einstellungen für eine andere Ansicht vorzunehmen, wechseln Sie zu dieser Ansicht und nehmen Sie anschließend unter **Extras | Anpassen | Symbolleisten** 1586 Ihre Einstellungen vor. Sie können Symbolleistengruppen auf der GUI auch an eine andere Stelle ziehen, indem Sie darauf klicken und mit dem Ziehpunkt (oder an der Titelleiste) an die gewünschte Stelle verschieben. Wenn Sie auf den Ziehpunkt doppelklicken, wird die Symbolleiste abgedockt und frei schwebend angezeigt; wenn Sie auf die Titelleiste doppelklicken, wird die Symbolleiste an der Stelle angedockt, an der sie sich vorher befunden hat.

Statusleiste

Die Statusleiste befindet sich am unteren Rand des Applikationsfensters (*siehe Abbildung* 115). In ihr werden die folgenden Informationen angezeigt: (i) Statusinformationen über das Laden von Dateien und (ii) Informationen über Menübefehle und Befehlsschaltflächen in den Symbolleisten, wenn Sie den Cursor darüber platzieren. Wenn Sie die 64-Bit-Version von%APPNAME%> verwenden, wird in der Statusleiste das Suffix (x64) hinter dem Applikationsnamen angezeigt. Für die 32-Bit-Version gibt es kein Suffix.

© 2016-2022 Altova GmbH

3.2 Die Applikationsumgebung

In diesem Abschnitt beschreiben wir verschiedene Aspekte der Applikation, die für den Anfang wichtig sind. Gehen Sie diesen Abschnitt durch, um sich mit XMLSpy vertraut zu machen. Dieser Abschnitt enthält wichtige Informationen über Einstellungen und Anpassungen und verschafft Ihnen einen allgemeinen Überblick darüber.

Dieser Abschnitt ist folgendermaßen gegliedert:

- *Einstellungen und Anpassung*[135]*:* Hier wird beschrieben wie und wo wichtige Einstellungen und anpassbare Optionen definiert werden können.
- *Tutorials, Projekte, Beispiele:*[138] Hier finden Sie die Pfade der verschiedenen im Applikationspaket enthaltenen Dateien, die nicht Teil des Programms selbst sind.
- *Produktfeatures, Dokumentation und Altova-Produkte*[138]*:* Hier finden Sie Links zur Altova Website, wo Sie Informationen über Produktfeatures, ein Hilfe in zusätzlichen Formaten und andere Altova-Produkte finden.

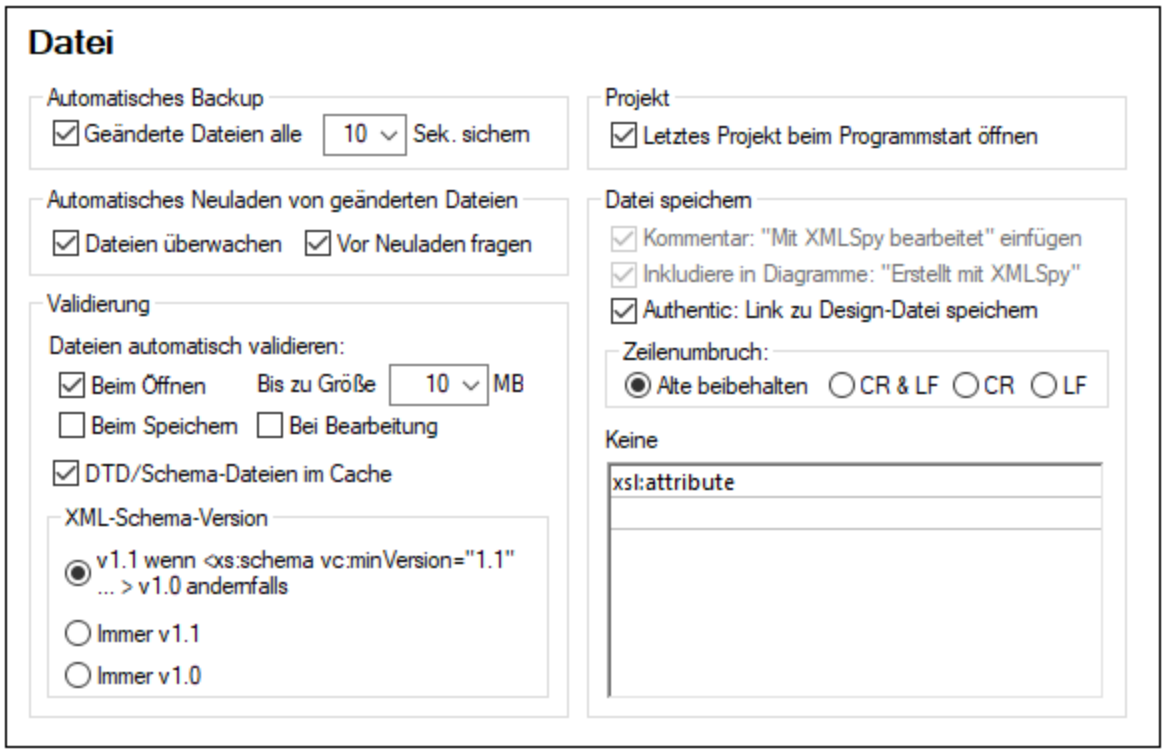
3.2.1 Einstellungen und Anpassung

In XMLSpy stehen verschiedene Einstellungen und Optionen zur Auswahl. In diesem Abschnitt erhalten Sie einen kurzen Überblick über diese Optionen. Außerdem werden einige Aspekte der XMLSpy Menüs kurz erläutert. Dieser Abschnitt ist in die folgenden Teile gegliedert.

- Einstellungen[135]
- Anpassung[137]
- Menüs[137]

Einstellungen

Auf den verschiedenen Registern im Dialogfeld "Optionen" wird eine Reihe wichtiger XMLSpy-Einstellungen definiert (*Abbildung unten*, Aufruf des Dialogfelds mit dem Befehl **Extras | Optionen**[1602]). Gehen Sie die verschiedenen Optionen selbst durch, um zu sehen, welche Optionen zur Verfügung stehen.



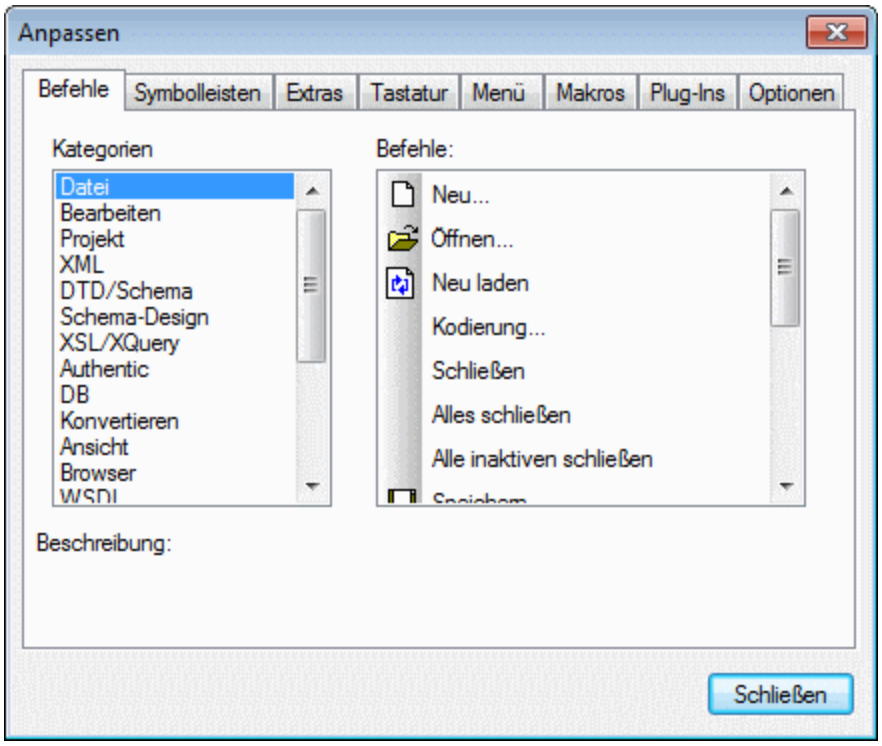
Im Folgenden finden Sie eine Zusammenfassung der wichtigsten Einstellungen. Nähere Informationen dazu finden Sie im Abschnitt "Benutzerreferenz" in der Beschreibung des Dialogfelds "Optionen"[1602].

- *Dateiarten und Standardansichten:* Auf dem Register "Dateiarten" können Sie eine Dateiart hinzufügen, damit XMLSpy sie erkennt. Fügen Sie dazu eine Dateierweiterung zur Liste der Dateierweiterungen hinzu. Anschließend können Sie für jede Dateiart festlegen, welcher Spezifikation Dateien dieses Typs entsprechen sollen und in welcher XMLSpy-Ansicht Dateien dieses Typs geöffnet werden sollen (die Standardansicht für diesen Dateityp).
- *Dateivalidierung:* Auf dem Register "Datei" können Sie festlegen, ob Dateien beim Öffnen und/oder Speichern automatisch validiert werden sollen. Anschließend können Sie auf dem Register "Dateiarten" die Dateivalidierung für bestimmte Dateiarten deaktivieren.
- *Bearbeitungsfunktionen:* Auf dem Register "Bearbeiten" können Sie festlegen, wie Eingabehilfen organisiert sein sollen, wie neue Elemente generiert werden und ob die Autokomplettierungsfunktion aktiviert werden soll. Für einzelne Ansichten stehen auf dem Register "Ansicht" zusätzliche Optionen zur Verfügung. Auf den Schriftartregistern für die einzelnen Ansichten können Sie die Schriftarteigenschaften für einzelne Node-Typen in den einzelnen Ansichten definieren.
- *XSLT- und FO-Prozessor:* Auf dem Register "XSL" können Sie festlegen, dass ein externer XSL-Prozessor für Transformationen verwendet werden soll, die über die Benutzeroberfläche gestartet werden. Dabei müssen Sie außerdem den Pfad zur ausführbaren Datei für den FO-Prozessor, der für die FO-Verarbeitung in XMLSpy verwendet werden soll, festlegen. Nähere Informationen dazu finden Sie im Abschnitt XSLT-Verarbeitung.[501]
- *Kodierung:* Auf dem Register "Kodierung" wird die Standardkodierung für XML- und nicht-XML-Dateien definiert.

© 2016-2022 Altova GmbH

Anpassung

Sie können verschiedene Aspekte von XMLSpy anpassen, wie unter anderem auch das Aussehen der Benutzeroberfläche. Die entsprechenden Optionen dazu finden Sie im Dialogfeld "Anpassen" (*Abbildung unten*, Aufruf über den Menübefehl **Extras | Anpassen**[1584]).



Die verschiedenen Konfigurationsoptionen sind im Abschnitt **Benutzerreferenz**[1584] beschrieben.

Menüs

Menübefehle sind abhängig von drei Faktoren aktiv/deaktiviert: (i) Dateiart, (ii) aktiver Ansicht und (iii) aktueller Cursorposition oder aktuellem Dokumentstatus. z.B:

- *Dateiarten:* Der Befehl **DTD/Schema | Weitere DTD einfügen** ist nur dann aktiv, wenn es sich bei der aktiven Datei um eine DTD handelt; so sind auch Befehle im Menü **WSDL** nur aktiv, wenn eine WSDL-Datei aktiv ist.
- *Aktive Ansicht:* Die meisten Befehle im Menü **Schema-Design** werden erst aktiv, wenn die aktive Ansicht die Schema-Ansicht ist.
- *Aktuelle Cursorposition, Dokumentstatus:* Ob in der Grid-Ansicht der Befehl zum Hinzufügen eines Attributs als Child-Node (**XML | Child hinzufügen | Attribute**) aktiv ist, hängt davon ab, ob es sich beim ausgewählten Node in der Grid-Ansicht um ein Element handelt oder nicht (*aktuelle Cursorposition*). Wenn ein XSLT-Dokument aktiv ist, steht der Befehl **Debugger anhalten** erst nach Beginn der Debugging-Sitzung zur Verfügung (*aktueller Dokumentstatus*).

Beachten Sie, dass Sie auch Menüs anpassen können (**Extras | Anpassen**[1602]) und sie durch Ziehen auch an eine andere Stelle der Benutzeroberfläche ziehen können (siehe Menüleiste, Symbolleisten, Statusleiste[133]).

3.2.2 Tutorials, Projekte, Beispiele

Das XMLSpy Installationspaket enthält Tutorials, Projekte und Beispieldateien.

Pfade von Tutorials, Projekten und Beispieldateien

Die XMLSpy-Tutorials, -Projekte und -Beispieldateien sind in folgendem Ordner installiert:

```
C:\Benutzer\<Benutzername>\Dokumente\Altova\XMLSpy2023\Examples\
```

Der Ordner `My Documents\Altova\XMLSpy2023` wird für jeden Benutzer, der auf dem PC einen Account hat, im entsprechenden Ordner dieses Benutzers unter `<username>` installiert. Bei diesem Installationssystem stehen jedem Benutzer seine eigenen `Examples`-Ordner in einem eigenen Arbeitsbereich zur Verfügung.

Pfade der Tutorial-, Projekt- und Beispieldateien

Alle Tutorial-, Projekt- und Beispieldateien befinden sich im Ordner `Examples`. Die genauen Pfade lauten wie folgt:

- *XMLSpy-Tutorial:* Ordner `Tutorial`.
- *Authentic-Tutorial:* Ordner `Examples`.
- *WSDL-Tutorial:* Ordner `Examples`.
- *Projektdatei:* Das Beispielprojekt, mit dem XMLSpy geöffnet wird, ist in `Examples.spp` definiert. Diese Datei befindet sich im Ordner `Examples`.
- *Beispieldateien:* befinden sich im Ordner `Examples` und in den Unterordnern dieses Ordners.

3.2.3 XMLSpy Features, Hilfe und Altova-Produkte

Auf der Altova-Website, www.altova.com, finden Sie eine Unmenge an Informationen zu XMLSpy und Ressourcen. Darunter die folgenden:

Liste der XMLSpy-Features

Die Altova-Website bietet eine aktuelle Liste der XMLSpy Features, in der Sie auf einen Blick sehen, welche Funktionen von den verschiedenen Versionen von XMLSpy (Enterprise und Professional) unterstützt werden

XMLSpy Hilfe

Diese Dokumentation ist die von Altova zur Verfügung gestellte Hilfe zu XMLSpy. Sie steht in Form des in XMLSpy integrierten Hilfesystems zur Verfügung, die mit dem Befehl **Hilfe** oder durch Drücken von **F1** aufgerufen werden kann. Des Weiteren stehen die Benutzerhandbücher für alle Altova-Produkte auch in den folgenden Formaten zur Verfügung:

- Online-HTML-Handbücher: Werden über die Support-Seite der Altova-Website aufgerufen

- Druckbare PDF-Dokumente: können von der Altova-Website heruntergeladen und lokal ausgedruckt werden
- Druckausgabe: können über einen Link auf der Altova-Website bestellt werden

Support-Optionen

Wenn Sie weitere Informationen darüber benötigen, welche Informationen im Benutzerhandbuch (also in dieser Dokumentation) zur Verfügung stehen, oder wenn Sie eine Frage zu Altova-Produkten haben, besuchen Sie unser Support Center auf der Altova-Website. Hier finden Sie folgende Informationen:

- Links zu unseren FAQ-Seiten
- Diskussionsforen zu Altova-Produkten und allgemeinen XML-Themen
- Online Support-Formulare, über die Sie Anfragen an den Support senden können, falls Sie über ein Support- und Wartungspaket verfügen. Unser Support-Team wird sich um Ihre Anfrage kümmern.

Altova-Produkte

Eine Liste aller Altova-Produkte finden Sie auf der Altova Website.

4 Bearbeitungsansichten

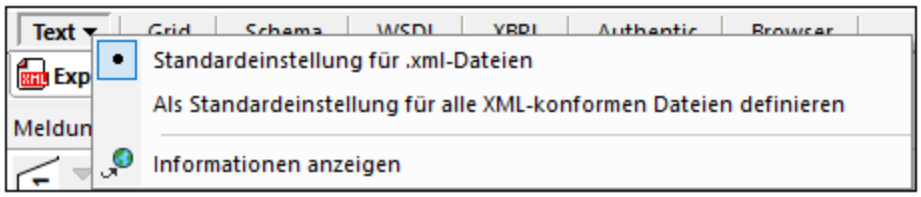
XMLSpy enthält leistungsstarke Bearbeitungsansichten. Zusätzlich zur Textansicht mit ihren intelligenten Bearbeitungsfunktionen gibt es grafische Ansichten, mit denen die Bearbeitung von Dokumenten einfacher wird. Abhängig davon, welche Dokumentart gerade in XMLSpy aktiv ist, stehen im Hauptfenster eine oder mehrere der Bearbeitungsansichten von XMLSpy zur Verfügung. Wenn z.B. eine Office Open XML-Datei oder eine ZIP-Datei geöffnet ist, so enthält das Hauptfenster nur eine Bearbeitungsansicht: die Archivansicht. Wenn ein HTML-Dokument aktiv ist, gibt es zwei Bearbeitungsansichten: die Text- und die Browser-Ansicht. Wenn ein XML-Dokument aktiv ist, stehen sieben Bearbeitungsansichten zur Verfügung: die Textansicht, die Grid-Ansicht, die Schema-Ansicht, die WSDL-Ansicht, die XBRL-Ansicht, die Authentic-Ansicht und die Browser-Ansicht; von diesen Ansichten ist die Schema-Ansicht nur bei XML-Schema-Dokumenten aktiv und die WSDL-Ansicht nur bei WSDL-Dokumenten.

In diesem Abschnitt beschreiben wir die verschiedenen Bearbeitungsansichten in XMLSpy:

- Textansicht 144
- Grid-Ansicht 169
- Schema-Ansicht 171
- WSDL-Ansicht 253
- XBRL-Ansicht 266
- Authentic-Ansicht 281
- Browser-Ansicht 282
- Archivansicht 284

Auswahl der Standardansicht

Ein Dokument eines bestimmten Typs (z.B. ein XML- oder ein JSON-Dokument) kann in verschiedenen Ansichten angezeigt werden. Sie können die Standardbearbeitungsansicht direkt auf der Benutzeroberfläche auswählen, indem Sie entweder mit der linken oder rechten Maustaste auf das Register der aktuellen Ansicht klicken und die entsprechende Option aus dem daraufhin angezeigten Menü auswählen *(siehe Abbildung unten):*



Die folgenden Menüoptionen stehen zur Verfügung:

- *Standardeinstellung für Dateierweiterung:* Die aktuelle Bearbeitungsansicht wird zur Standardansicht für Dateien, die **dieselbe Dateierweiterung** wie die aktive Datei haben (z.B. `.xml`-Dateien oder `.xq`-Dateien).
- *Als Standardansicht für alle ähnlich konformen Dateien definieren:* Die aktuelle Bearbeitungsansicht wird zur Standardansicht für Dateien, die **denselben Konformitätstyp** wie die aktive Datei haben (z.B. XML-konforme Dateien oder alle XQuery-konformen Dateien). Dabei kann es sich um eine größere Gruppe von Dateien als bei einer Dateierweiterung handeln. So könnten etwa die zwei unterschiedlichen Dateierweiterungen `.xq` und `.xquery` beide XQuery-konform sein.
- *Informationen anzeigen:* Verlinkt auf eine Seite auf der Altova Website, auf der die Funktionalitäten der aktuellen Bearbeitungsansicht beschrieben sind.

© 2016-2022 Altova GmbH

Wenn eine Standardansichtsoption ausgewählt wurde, wird daneben ein Optionsfeld angezeigt (*siehe Abbildung oben*).

In den folgenden Fällen kann die Standardbearbeitungsansicht nicht geändert werden:

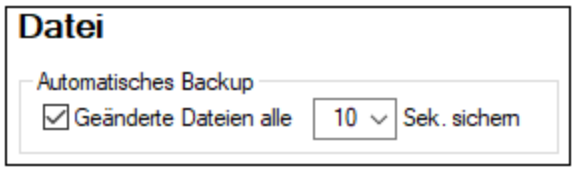
- *JSON-Schema-Dateien:* Diese Dateien haben wie JSON-Instanzdateien die Erweiterung **.json**. Anhand ihres Inhalts werden sie jedoch automatisch als JSON-Schemas erkannt und in der JSON-Schema-Ansicht 726 geöffnet.
- *Authentic-fähige XML-Dateien:* Dies sind XML-Dateien (mit der Dateierweiterung **.xml**), denen ein Altova StyleVision Stylesheet zugewiesen wurde. Mit Hilfe dieses Stylesheet kann das XML-Dokument als XML-Authentic-Dokument angezeigt werden. Solche Dateien werden immer in der Authentic-Ansicht 281 geöffnet.
- *XSD-Dateien für XBRL (XBRL-Taxonomien):* Diese Dateien haben die Dateierweiterung **.xsd**. Anhand ihres Inhalts werden sie automatisch als XBRL-Taxonomien (und nicht als XML-Schemas) erkannt und daher immer in der XBRL-Ansicht 266 geöffnet.

Anmerkung: Die Dateikonformität einzelner Dateierweiterungen sowie die Standardbearbeitungsansicht für einzelne Dateitypen kann im Dialogfeld "Optionen" im Abschnitt "Dateiarten" 1606 definiert werden. Schneller und einfacher ist dies jedoch über die hier beschriebene Auswahl der Standardansicht in der Bearbeitungsansicht selbst möglich.

4.1 Automatisches Backup von Dateien

In XMLSpy geänderte Dateien werden in regelmäßigen Abständen automatisch gesichert. Im Dialogfeld "Optionen" (**Extras | Optionen | Datei**[1603]) auf dem Register *Datei* (siehe Abbildung unten) können Sie:

- automatische Backups aktivieren/deaktivieren
- die Backup-Intervalle (5 Sekunden bis 300 Sekunden) definieren



Alle Dateiarten, die in XMLSpy bearbeitet werden können, werden gesichert; ZIP-Archive werden nicht gesichert.

Anzeigesymbole

Die Dateiregister am unteren Rand des Hauptfensters enthalten jeweils ein Symbol rechts vom Dateinamen, das den gespeicherten/nicht gespeicherten Status und den Backup-Status der Datei angibt (*Abbildung unten*).



Gespeichert / nicht gespeichert
Wenn eine Datei geändert wurde, wird ein farbiges Kreissymbol angezeigt. Wenn kein solches Symbol angezeigt wird, bedeutet dies, dass die Datei seit der Öffnung oder der letzten Speicherung nicht geändert wurde. So sehen Sie etwa in der Abbildung oben, dass `address.xsd` seit der letzten Speicherung nicht geändert wurde. `Unbenannt1.xml` ist eine neue Datei, die seit ihrer Erstellung nicht bearbeitet oder gespeichert wurde.

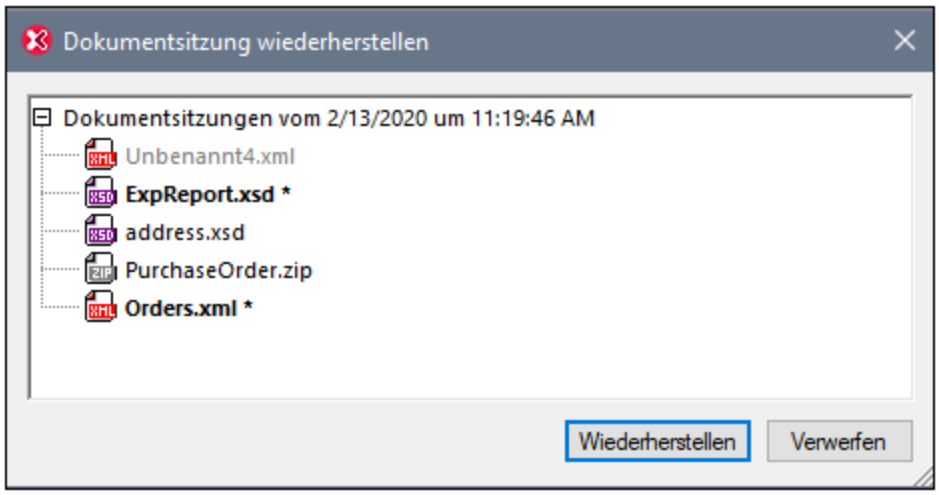
Backup-Status
Die Farben der Kreissymbole geben den Backup-Status der Datei an.

- *Gelb:* Die Datei wurde geändert, doch wurde die letzte Änderung nicht gesichert (oder gespeichert).
- *Grün:* Die Datei wurde gesichert und wurde seither nicht geändert. Jedoch wurde die Datei noch nicht gespeichert. (Wäre sie gespeichert worden, würde kein Kreissymbol angezeigt).
- *Rot:* Ein Backup wird für diese Datei nicht unterstützt (z.B. wenn es sich um die Archivansicht[284] einer ZIP-Datei handelt) oder das Backup ist fehlgeschlagen.
- *Grau:* Die automatische Backup-Funktion wurde (über das Dialogfeld Optionen[1603], *siehe oben*) deaktiviert. Dass ein Symbol angezeigt wird, gibt an, dass die Datei seit der letzten Änderung noch nicht gespeichert wurde. (Wäre sie gespeichert worden, würde kein Kreissymbol angezeigt).

Wiederherstellung anhand von Backups

Wenn XMLSpy unerwartet beendet wird, wird beim nächsten Applikationsstart ein "Dokument wiederherstellen"-Dialogfeld angezeigt, welches eine Liste aller zum Zeitpunkt der Applikationsbeendung geöffneten Dokumente enthält (*Abbildung unten*). Wenn Sie den Mauszeiger über eine Datei platzieren, sehen Sie ihren Pfad. Bei temporären Dateien, die noch nicht gespeichert wurden, wird als aktueller Standardpfad der Dateipfad angezeigt, der bei Öffnung des Dialogfelds "Speichern unter" angezeigt würde.

© 2016-2022 Altova GmbH



Bei jeder Datei in der Liste geben die Schriftart und das Vorhandensein oder Fehlen von Sternchen Aufschluss über die folgenden Dinge:

- Ein fetter Stil und ein Sternchen geben an, dass die Datei nicht gespeicherte Änderungen enthält. Derartige Dateien werden im Zustand der letzten Sicherung wiederhergestellt.
- Ein normaler Stil gibt an, dass die Datei gespeichert wurde und es keine nicht gespeicherten Änderungen gibt. Derartige Dateien werden zu dem Zustand der Speicherung wiederhergestellt.
- Ein ausgegrauter Stil zeigt an, dass die Datei weder gespeichert noch gesichert wurde (z.B. weil es sich um eine neue Datei handelt, die nicht bearbeitet wurde). Derartige Dateien werden nicht wiederhergestellt.

Sie haben nun eine der folgenden Möglichkeiten:

- Klicken Sie auf **Wiederherstellen**, um die Dateien auf der Benutzeroberfläche anhand ihres letzten Backups wiederherzustellen.
- Klicken Sie auf **Verwerfen**, um keine der aufgelisteten Dateien zu öffnen und alle vorhandenen Backups zu verwerfen.

4.2 Textansicht

In der Textansicht (*Abbildung unten*) können Sie den Text Ihres Dokuments - sowohl den Markup-Code als auch den Inhalt - direkt eingeben. Jede beliebige Textdatei, auch Nicht-XML-Dokumente (wie z.B. XQuery- und HTML-Dokumente), kann in der Textansicht bearbeitet werden. Mit Hilfe einer Reihe von Funktionen können Sie Ihr Dokument schnell und korrekt eingeben.



In diesem Abschnitt werden allgemeine Funktionen der Textansicht beschrieben, die für alle Arten von Dokumenten zur Verfügung stehen. Bestimmte Dokumenttypen wie z.B. XML-, XQuery- und CSS-Dokumente haben bestimmte dokumentspezifische Funktionen, die in den entsprechenden Abschnitten zu diesen Dokumenttypen beschrieben sind. So werden etwa zusätzliche XML-spezifische Funktionen der Textansicht im Abschnitt XML | Bearbeiten in der Textansicht 293 beschrieben.

Die allgemeinen Funktionen der Textansicht wurden folgendermaßen gegliedert:

- Formatieren in der Textansicht 145: Hier wird beschrieben, wie die Schriftarteigenschaften, die Einrückung und der Zeilenumbruch im Dokument definiert werden.
- Anzeige des Dokuments 148: Enthält Informationen zur Zeilennummerierung, Lesezeichen, zum Erweitern/Reduzieren von Nodes und anderen Funktionen, die mit der Anzeige von Dokumenten in Zusammenhang stehen.
- Bearbeiten in der Textansicht 151: Hier werden die Funktionen beschrieben, die bei der Bearbeitung zur Verfügung stehen, insbesondere die intelligenten Bearbeitungsfunktionen.
- Navigation im Dokument 154: Hier werden die verschiedenen Methoden, mit denen Sie in der Textansicht in einem Dokument navigieren können, erläutert.

 © 2016-2022 Altova GmbH

- Suchen und Ersetzen[157]: Hier werden die Funktionen zum Suchen und Ersetzen in der Textansicht beschrieben.
- Eingabehilfen[165] sind die Fenster, in denen kontextsensitive Dateneingabeoptionen zur Verfügung stehen. So werden z.B. in einer Eingabehilfe die Elemente bzw. Attribute angezeigt, die an einer bestimmten Stelle des Dokuments gültig sind und hinzugefügt werden können. Jede dieser Optionen kann durch Doppelklick eingefügt werden.
- Geteilte Ansicht[166] teilt das Hauptfenster der Textansicht in zwei Teile und zeigt das aktive Dokument in beiden Ansichten an. Sie können dadurch verschiedene Teile eines langen Dokuments Seite an Seite nebeneinander anzeigen.
- Tastaturkürzel für die Textansicht[167]: Enthält eine Liste der Standardtastaturkürzel häufig verwendeter Textansichtsbefehle.

Wechseln in die Textansicht

Um die Textansicht eines Dokuments zu öffnen, klicken Sie auf die Schaltfläche **Text** am unteren Rand des Dokumentfensters oder wählen Sie **Ansicht | Textansicht**.

4.2.1 Formatieren in der Textansicht

Die Textansicht bietet eine Reihe von Textformatierungsoptionen. Diese sind unten aufgelistet.

Schriftarten

Die Schriftart, Schriftgröße, der Schriftgrad und der Texthintergrund können für die folgenden Dokumentgruppen separat angepasst werden: (i) allgemeine XML-Dokumente (einschließlich HTML-Dokumente); (ii) XQuery-Dokumente; und (iii) CSS-Dokumente.

Für einzelne Textelemente in einem Dokument, die eine unterschiedliche Semantik haben, können unterschiedliche Textfarben verwendet werden. So können Sie z.B. unterschiedliche Farben für Elementnamen und Attributnamen und den Inhalt von Elemente verwenden. Wenn Sie unterschiedliche Farben für unterschiedliche Textelemente verwenden, ist die Syntaxfärbungsfunktion aktiviert. Textschriftarten werden im Dialogfeld "Optionen" im Abschnitt "Schriftarten"[1623] angepasst.

Einrückung

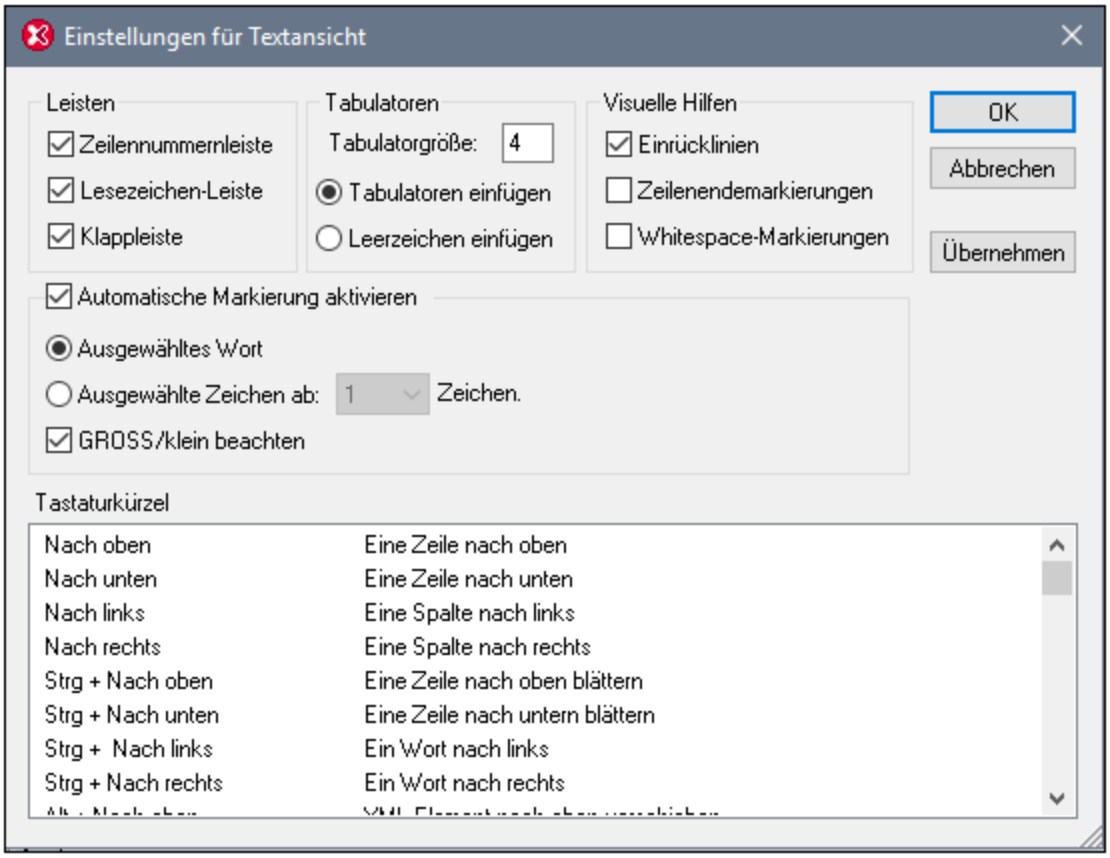
Sie können ein Dokument mit Einrückungen versehen, um seine Struktur deutlich zu machen, wie in der Abbildung unten gezeigt. Wenn das Dokument solcherart hierarchisch gegliedert angezeigt wird, wird es im Pretty-Print-Modus angezeigt. In einem mit Pretty-Print angezeigten Dokument wird jede untergeordnete Ebene mit einer weiteren Einrückung gekennzeichnet. Um ein Dokument mit Pretty-Print anzuzeigen, müssen Sie Folgendes tun: (i) Sie müssen die Einrückung (bedarfsgemäß) konfigurieren und (ii) Pretty-Print auf das Dokument anwenden.



Um Pretty-Print zu konfigurieren und anzuwenden, gehen Sie folgendermaßen vor. Beachten Sie, dass die Pretty-Print-Konfiguration für die gesamte Applikation gilt. Dies bedeutet, dass Pretty-Print-Einstellungen nach Speicherung der Einstellungen auf alle XML- und JSON-Dokumente angewendet werden. Die Einstellungen müssen daher nur bearbeitet werden, wenn Sie die Pretty-Print-Anzeige Ihrer Dokumente ändern möchten. Sobald Sie die applikationsweiten Einstellungen vorgenommen haben, müssen Sie Pretty-Print einzeln auf jedes Dokument anwenden. Ein Dokument wird immer mit den Pretty-Print-Einstellungen, die darauf angewendet wurden, angezeigt. Die Formatierung der Anzeige ändert sich erst, wenn Pretty-Print das nächste Mal auf das Dokument angewendet wird.

1. Aktivieren Sie im Pretty-Print-Bereich[1611] des Dialogfelds Optionen (**Extras | Optionen**[1602]) Pretty-Print durch Aktivierung der Option *Einrückung aus der Tab-Konfiguration der Textansicht verwenden*. Mit Hilfe dieser Einstellung werden die Einrückungen, die in einem Pretty-Print-Dokument angezeigt werden, aktiviert. Wenn die Option *Einrückung verwenden* nicht aktiviert ist, beginnt jede Zeile im Dokument mit der Einrückung Null.
2. Klicken Sie im Pretty-Print-Bereich[1611] (*siehe vorheriger Punkt*) rechts oben auf die Schaltfläche **Einstellungen für Textansicht**, um das Dialogfeld Einstellungen für Textansicht (*Abbildung unten*) aufzurufen und legen Sie die Einrückungszahl und die Art der Einrückung fest. (Das Dialogfeld "Einstellungen für Textansicht" kann auch mit dem Menübefehl **Ansicht | Einstellungen für Textansicht**[1506] und (ii) über die Schaltfläche **Einstellungen für Textansicht** in der Symbolleiste "Text" aufgerufen werden). Das Feld *Tabulatorgröße* im Dialogfeld "Einstellungen für Textansicht" definiert die Anzahl der Leerzeichen, als die eine Tabulatoreinrückung definiert wird. Wählen Sie aus, ob die Pretty-Print-Einrückungen aus Tabulatoren oder Leerzeichen gebildet werden sollen, indem Sie entweder *Tabulatoren einfügen* oder *Leerzeichen einfügen* auswählen. In beiden Fällen entspricht die Größe der Einrückung der Anzahl der **x** Leerzeichen, die im Feld *Tabulatorgröße* definiert wurden (da 1 Tabulator = X Leerzeichen).

 © 2016-2022 Altova GmbH



3. Nachdem Sie das applikationsweite Pretty-Printing, wie in den beiden vorherigen Schritten beschrieben, definiert haben, können Sie Pretty-Print auf einzelne Dokumente anwenden. Machen Sie das gewünschte Dokument zum aktiven Dokument. Wählen Sie den Befehl **Bearbeiten | Pretty-Print**[1306] aus oder klicken Sie in der Symbolleiste auf die Schaltfläche **Pretty Print**. Daraufhin wird der Dokumenttext, gemäß der im Dialogfeld "Optionen" im Abschnitt Pretty-Print[1611] definierten Formatierung angezeigt. Beachten Sie, dass mit dem **Pretty-Print**-Befehl unnötige vorangestellte oder nachgestellte Leerzeichen entfernt werden.

Anmerkung: Bei den Einrücklinien handelt es sich um gepunktete vertikale Linien (*siehe Abbildung am Anfang dieses Abschnitts*), die im Dialogfeld "Einstellungen für Textansicht" im Bereich *Visuelle Hilfe* über das Kontrollkästchen "Einrücklinien" (*siehe Abbildung oben*) ein- und ausgeblendet werden können.

Anmerkung: Informationen zur Behandlung von Whitespaces finden Sie im Abschnitt Whitespace[341].

Verwendung von Tabulatoren und Leerzeichen zum Formatieren

Sie können Text, v.a. Nicht-XML-Dokumente, für die die Pretty-Print-Option nicht zur Verfügung steht, mit Hilfe von Tabulatorzeichen und Leerzeichen formatieren. Bei Drücken der **Eingabetaste** bzw. von **Umschalt + Eingabe** springt der Cursor an die Stelle in der nächsten Zeile, an der die vorige Zeile begonnen hat.

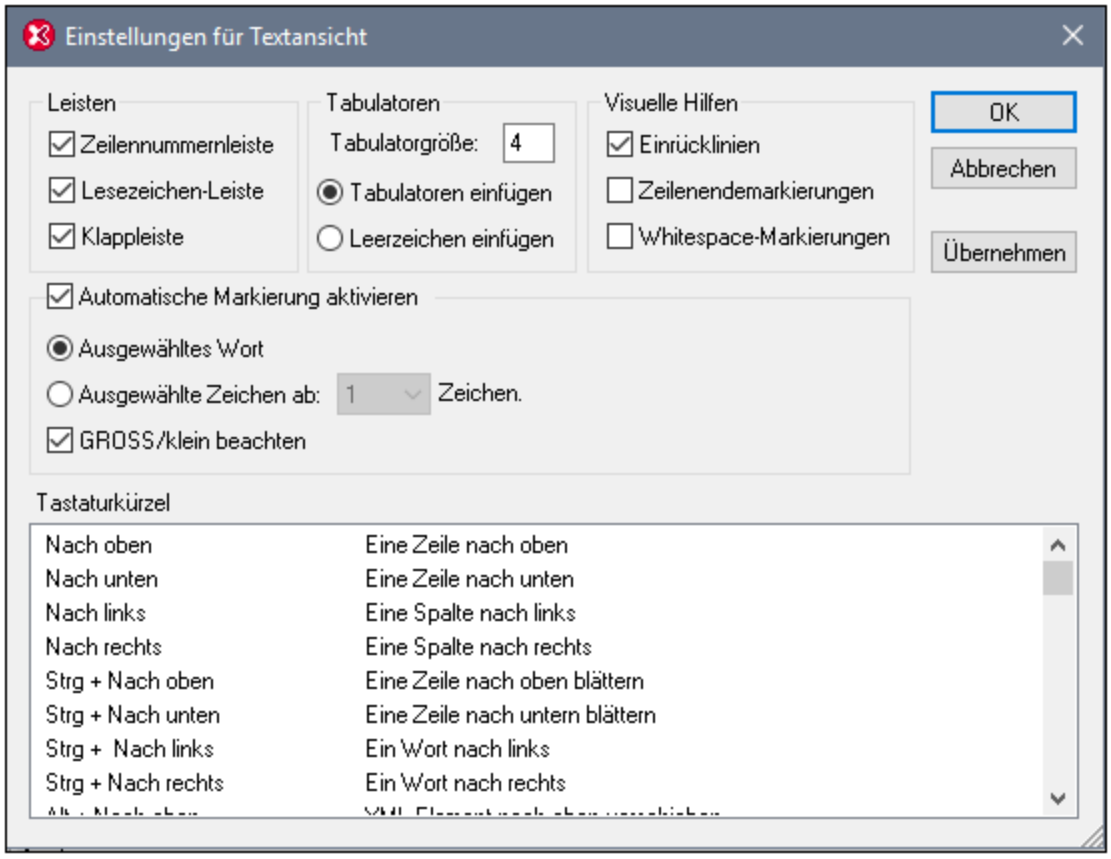
Zeilenumbruch

Textzeilen, die über die Fensterbreite hinausgehen, können umbrochen werden. Aktivieren Sie dazu den Befehl **Ansicht | Zeilenumbruch**[1505]. Die entsprechende Schaltfläche dazu befindet sich in der Symbolleiste "Text"[148].

4.2.2 Anzeige des Dokuments

Die Textansicht enthält einige nützliche Funktionen, die die Bearbeitung großer Textpassagen erleichtern: (i) Zeilennummerierung[149], (ii) Lesezeichen[149], (iii) Klappleiste[149] (zum Ein- und Ausklappen von Nodes), (iv) Einrücklinien[150] und Zeilenendezeichen und Whitespace-Markierungen[150]. Diese Befehle stehen im Dialogfeld "Einstellungen für Textansicht"(*erste Abbildung unten)* und in der Symbolleiste "Text" (*zweite Abbildung unten*) zur Verfügung.

Das Dialogfeld "Einstellungen für Textansicht"[1506] wird über den Befehl **Ansicht | Einstellungen für Textansicht** oder die gleichnamige Schaltfläche in der Symbolleiste "Text" oder das Textansicht-Kontextmenü aufgerufen. Die Einstellungen im Dialogfeld "Einstellungen für Textansicht" gelten für die gesamte Applikation - nicht nur für das aktive Dokument.



 © 2016-2022 Altova GmbH



Weitere nützliche Funktionen sind die Vergößern/Verkleinern- 151 und die Funktion zum Navigieren und Suchen im Text 154.

Zeilennummerierung

Die Zeilennummern werden in der Zeilennummernleiste (*Abbildung unten*) angezeigt, die über das Dialogfeld "Einstellungen für Textansicht" (siehe Abbildung oben) ein- und ausgeblendet werden können. Wenn ein Textabschnitt eingeklappt ist, werden auch die Zeilennummern der entsprechenden Textzeilen ausgeblendet. Dazu gibt es auch den Befehl Gehe zu Zeile/Zeichen 157.

Lesezeichen

Für den raschen Zugriff darauf können Zeilen in einem Dokument separat mit Lesezeichen versehen werden. Wenn die Lesezeichenleiste eingeblendet ist, werden die Lesezeichen in der Lesezeichenleiste angezeigt. Andernfalls werden mit Lesezeichen versehene Zeilen in Zyan markiert.

Die Lesezeichenleiste kann im Dialogfeld "Einstellungen für Textansicht" (*Abbildung oben*) ein- und ausgeblendet werden.

Sie können Lesezeichen mit Hilfe der Befehle im Menü **Bearbeiten** und in der Symbolleiste "Text" bearbeiten und darin navigieren. Lesezeichen können mit dem Befehl **Bearbeiten | Lesezeichen einfügen/löschen** eingefügt werden. Sie können damit eine Zeile im Dokument zu Referenzzwecken markieren. Um ein Lesezeichen zu löschen, wählen Sie die gekennzeichnete Zeile aus und wählen Sie anschließend den Befehl **Bearbeiten | Lesezeichen einfügen/löschen.** Um zwischen den Lesezeichen in einem Dokument zu navigieren, wählen Sie die Befehle **Bearbeiten | Zum nächsten Lesezeichen** und **Bearbeiten | Zum vorhergehenden Lesezeichen**. Diese Lesezeichenbefehle stehen auch in Form von Schaltflächen in der Symbolleiste "Text" (*Abbildung oben*) zur Verfügung.

Klappleiste

Mit Hilfe der Klappleiste können Sie Nodes in XML-, XQuery-, JSON- und CSS-Dokumenten erweitern und reduzieren. Nodes, die erweitert/reduziert werden können, werden in der Klappleiste durch ein +/−-Zeichen markiert (*siehe Abbildung unten*). Die Klappleiste kann über das Dialogfeld "Einstellungen für Textansicht" (*siehe Abbildung oben*) ein- und ausgeblendet werden. In der Abbildung unten sehen Sie, dass drei Nodes reduziert wurden: das Element `shipTo` und zwei `item`-Elemente. Wenn ein Node reduziert wurde, wird dies durch Auslassungspunkte gekennzeichnet (*grün markierter Bereich in der Abbildung unten*). Wenn Sie den Mauszeiger über die Auslassungspunkte platzieren, wird der Inhalt des eingeklappten Node in einem Popup-Fenster angezeigt (*in der Abbildung unten blau markiert*). Falls der Inhalt für das Popup-Fenster zu groß ist, so wird dies durch Auslassungspunkte am unteren Rand des Popup-Fenster angezeigt.



Mit der Schaltfläche **Alle ein-/ausklappen** in der Symbolleiste "Text" werden **alle** Nodes in ihrer erweiterten Form angezeigt bzw. auf das Dokumentelement der obersten Ebene reduziert.
Wenn Sie auf das +/−-Symbol eines Node klicken, stehen die folgenden Optionen zur Verfügung:

Klick [−]	Klappt den Node ein.
Klick [+]	Zeigt den Node ausgeklappt an, sodass seine untergeordneten Nodes, je nachdem, in welchem Zustand sie sich vor dem Einklappen befunden haben, im erweiterten bzw. reduzierten Zustand zu sehen sind.
Umschalt+Klick [−]	Reduziert alle untergeordneten Nodes, belässt jedoch den angeklickten Node in seiner erweiterten Form.
Strg+Klick [+]	Erweitert den angeklickten Node sowie alle seine untergeordneten Nodes.

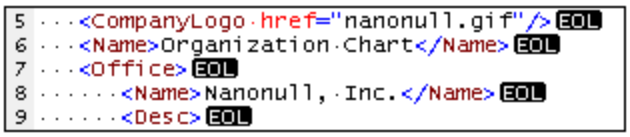
Einrücklinien

Einrücklinien sind vertikale gepunktete Linien, anhand derer Sie den Einrückungsgrad einer Zeile sehen können (*siehe Abbildung oben*). Sie können über das Dialogfeld "Einstellungen für Textansicht" ein- und ausgeblendet werden.

Zeilenendezeichen, Whitespace-Markierungen

Zeilenendezeichen (EOL = End Of Line) und Whitespace-Markierungen können über das Dialogfeld "Einstellungen für Textansicht" ein- und ausgeblendet werden. In der Abbildung unten sehen Sie diese Markierungen in der Dokumentanzeige; jeder Punkt steht für ein Leerzeichen.

© 2016-2022 Altova GmbH



Vergrößern/Verkleinern

Durch Drehen des Mausrads und gleichzeitiges Gedrückthalten der **Strg**-Taste können Sie die Textansicht vergrößern und verkleinern. Auf diese Art können Sie die Größe des angezeigten Texts in der Textansicht vergrößern und verkleinern. Wenn Sie die Schriftgröße ändern möchten, tun Sie dies über das Dialogfeld Optionen[145].

4.2.3 Bearbeiten in der Textansicht

Die folgenden Textbearbeitungsfunktionen stehen in der Textansicht im Allgemeinen für alle Dokumenttypen zur Verfügung. Diese Funktionen sind Ergänzungen zu allgemeinen Bearbeitungsfunktionen wie **Ausschneiden, Kopieren, Einfügen, Löschen** und **Alles markieren** (die im Menü **Bearbeiten** zur Verfügung stehen).

- Syntaxfärbung[151]
- Start-Tag and End-Tag-Suche[153]
- Intelligentes Editieren[152]
- Autokomplettierung[152]
- Verschieben von gleichrangigen Elementen relativ zu einander[153]
- Auswahl eines ganzen Elements und Springen zum übergeordneten Element[153]
- Drag-and-Drop und Kontextmenüs[153]
- Unbegrenztes Rückgängigmachen[153]
- Rechtschreibprüfung[154]

Für einige Dokumenttypen (wie z.B. XML[288] und XQuery[514]) stehen zusätzliche spezielle Funktionen zur Verfügung. Diese Funktionen werden in den entsprechenden Abschnitten zu diesen Dokumenttypen beschrieben.

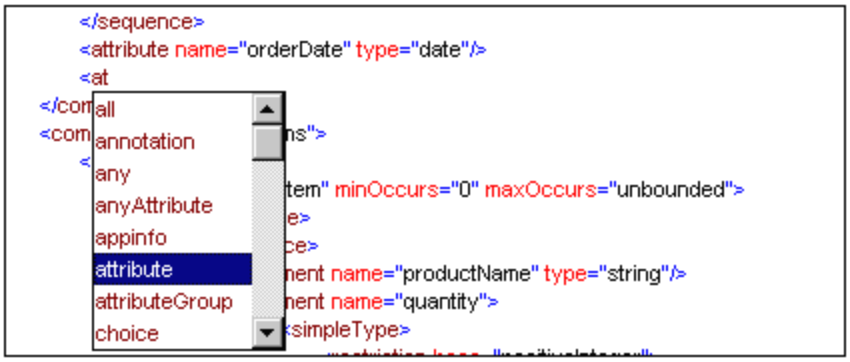
Anmerkung: Bei großen Dateien können die Autokomplettierung und die Eingabehilfen deaktiviert werden, um ein schnelleres Laden und eine schnellere Bearbeitung zu gewährleisten. Der Grenzwert kann vom Benutzer definiert werden. Nähere Informationen dazu finden Sie im Abschnitt "Referenz" unter Optionen | Bearbeitung[1609].

Syntaxfärbung

Die Syntaxfärbung richtet sich nach der Art des XML Nodes, d.h. je nachdem, ob es sich beim XML Node um ein Element, Attribut, Inhalt, einen CDATA-Abschnitt, Comment oder eine Processing Instruction handelt. Man unterscheidet zwischen einer Reihe von Gruppen von Dokumenttypen: (i) allgemeinen XML-Dokumenten (einschließlich HTML); (ii) XQuery, (iii) CSS und (iv) JSON. Die Texteigenschaften dieser Gruppen können im Dialogfeld "Optionen" (**Extras | Optionen**) im Abschnitt "Textschriftarten" eingestellt werden.

Intelligentes Editieren

Wenn Sie mit einem XML-Dokument arbeiten, das auf einem XML-Schema basiert, stehen Ihnen in XMLSpy in der Textansicht verschiedene intelligente Editierfunktionen zur Verfügung, mit Hilfe derer Sie schnell das richtige Element, Attribut oder den richtigen Attributwert einfügen können, und zwar dem Content Model entsprechend, das für das Element definiert wurde, das Sie gerade bearbeiten. Wenn Sie z.B. den Beginn des Start-Tags eines Elements eingeben, werden die Namen aller Elemente angezeigt, die laut Schema an dieser Stelle zulässig sind (*Abbildung unten*). Wenn Sie den Namen des gewünschten Elements auswählen und die Eingabetaste drücken, wird dieser Name in den Start-Tag eingefügt. Nachdem der Start-Tag erstellt wurde, wird automatisch ein End-Tag hinzugefügt (*siehe Autokomplettierung 152 weiter unten*).



Das Popup-Fenster wird auch in den folgenden Fällen angezeigt:

- Wenn Sie die Leertaste drücken, während sich der Cursor innerhalb des Start-Tags eines Elements befindet, für das ein Attribut definiert ist. Das Popup-Fenster enthält alle verfügbaren Attribute.
- Wenn sich der Cursor innerhalb der doppelten Anführungszeichen befindet, die einen Attributwert umschließen, für den eine Aufzählung mit Werten zur Auswahl steht. Im Popup-Fenster werden die aufgezählten Werte angezeigt.
- Wenn Sie das Zeichen `</` (den Beginn eines End-Tags) eingeben, wird im Popup-Fenster der Name des zu schließenden Elements angezeigt.
- Wenn Sie ein leeres Element als einzelnen Tag anlegen möchten oder ein leeres Element bestehend aus zwei Tags in ein leeres Element aus einem Tag konvertieren möchten, geben Sie nach dem Elementnamen den schließenden Schrägstrich ein: `<element/`. Daraufhin wird ein leeres Element mit einem einzigen Tag erstellt; wenn ein schließender Tag existiert, wird dieser entfernt: `<element/>`.

Autokomplettierung

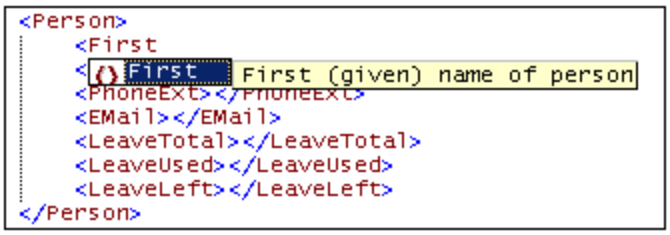
Wenn Sie XML-Dokumente (oder andere Dokumente, die mit Markup-Code versehen sind, wie z.B. HTML) in der Textansicht bearbeiten, wird oft die Wohlgeformtheit eines Dokuments verletzt, so können z.B. unter Umständen End-Tags fehlen, falsch geschrieben sein oder in der Struktur an einer falschen Stelle stehen. XMLSpy ergänzt den Start- und End-Tag von Elementen automatisch und fügt alle obligatorischen Attribute ein, sobald Sie auf der Tastatur den Elementnamen fertig eingegeben haben. Außerdem wird der Cursor automatisch zwischen den Start- und End-Tag des Elements gesetzt, sodass Sie gleich weitere Child-Elemente oder Inhalt eingeben können:

```
<img src="" alt="">|</img>
```

Zur Unterstützung der Autokomplettierungsfunktion bedient sich XMLSpy der XML-Regeln zur Wohlgeformtheit und Gültigkeit eines Dokuments. Die Informationen über die Struktur des Dokuments werden aus dem Schema abgerufen, auf dem das XML-Dokument basiert. (im Fall von häufig gebrauchten Schemas wie HTML und XSLT,

© 2016-2022 Altova GmbH

sind die Schemainformationen in XMLSpy bereits vordefiniert). Zur Autokomplettierung werden nicht nur Informationen über die Struktur des Dokuments verwendet, sondern auch die im Dokument gespeicherten Wert. So werden z.B. Enumerationen und Schema-Annotationen in einem XML-Schema aktiv von der Autokomplettierungsfunktion verwendet. Wenn im Schema Werte für einen bestimmten Node in einer Enumeration aufgezählt werden, werden diese Enumerationen als Autokomplettierungsoptionen angezeigt, wenn dieser Node im XML-Dokument bearbeitet wird. Ebenso werden auch Annotationen, die in einem Schema für einen Node bestehen, angezeigt, wenn der Node-Name in das Dokument eingegeben wird (*Abbildung unten*). (*First (given) name of person* ist die Schema-Annotation des Elements `First`.)



Die Autokomplettierungsfunktion kann im Dialogfeld "Optionen" im Abschnitt "Bearbeiten" 1609 (**Extras | Optionen | Bearbeiten**) aktiviert und deaktiviert werden.

Übereinstimmung von Start-Tag und End-Tag

Wenn Sie den Cursor in den Start- oder End-Tags eines XML-Elements setzen, wird bei Drücken von **Strg+E** der dazugehörige andere Tag ausgewählt. Bei wiederholtem Drücken von **Strg+E** können Sie zwischen dem Start- und End-Tag hin- und herwechseln. Dies ist ein gutes Hilfsmittel, um die Start- und End-Tags eines XML-Elements zu finden. Zusätzlich dazu werden die Namen von Elementen in unterschiedlichen Farben markiert 1623, je nachdem, ob die Namen im Start- und End-Tag miteinander übereinstimmen oder nicht. Dies dient als visuelle Bearbeitungshilfe. Die Markierungsfarben können im Dialogfeld "Optionen" 1623 definiert werden. Wenn Sie den Namen eines Elements in einem Start-Tag bearbeiten, so wird automatisch auch der End-Tag geändert.

Verschieben von gleichrangigen Elementen relativ zu einander

Wenn sich der Cursor innerhalb eines Elements befindet, können Sie das ausgewählte Element durch Drücken von **Alt+nach oben** bzw. **Alt+nach unten** relativ zu den gleichrangigen Elementen nach oben bzw. unten verschieben.

Auswahl eines ganzen Elements und Springen zum übergeordneten Element

Wenn sich der Cursor innerhalb eines Elements befindet, können Sie das ganze Element durch Drücken von **Strg+Umschalt+E** auswählen. Wenn Sie **Strg+Alt+E** drücken, wird der Start-Tag des übergeordneten Elements markiert. Mit Hilfe dieser beiden Tastaturkürzel sehen Sie schnell, wo sich Ihr Cursor gerade relativ zur Dokumentstruktur befindet.

Drag-and-Drop und Kontextmenüs

Sie können einen Textblock auch mit Drag-and-Drop an eine andere Stelle verschieben und häufig verwendete Editier-Befehle (wie Ausschneiden 1298, Kopieren 1298, Einfügen 1298, Löschen 1298, Als Mail senden 1292 und Gehe zu Zeile/Zeichen 1505) mit einem Rechtsklick direkt über ein Kontextmenü aufrufen.

Unbeschränktes Rückgängigmachen

XMLSpy bietet Ihnen für alle Editiervorgänge unbeschränktes Rückgängigmachen 1298 und Wiederherstellen 1298.

Rechtschreibprüfung

Dokumente können in der Textansicht mit Hilfe jedes der integrierten Sprachwörterbücher überprüft werden. Außerdem kann ein benutzerdefiniertes Wörterbuch erstellt und bearbeitet werden, mit dem auch Wörter, die nicht im Sprachwörterbuch enthalten sind, überprüft werden können. Nähere Informationen finden Sie unter den Beschreibungen zu den Befehlen Rechtschreibung[1558] und Rechtschreiboptionen[1562].

4.2.4 Navigation im Dokument

Mit Hilfe der folgenden Funktionen können Sie in einem Dokument in der Textansicht navigieren:

- Textmarkierung[154]: Damit können Sie alle Übereinstimmungen mit einem von Ihnen ausgewählten Text-String oder Wort suchen. Jede Übereinstimmung wird in der Bildlaufleiste markiert, so dass Sie einfach zwischen den einzelnen Treffern navigieren können.
- Dokumentübersicht in der Bildlaufleiste[156]: Zeigt die relative Position des Cursors und des ausgewählten Texts im Dokument an.
- Gehe zu Zeile/Zeichen[157]: Geht direkt zur angegebenen Zeile bzw. zum angegebenen Zeichen.

Textmarkierung

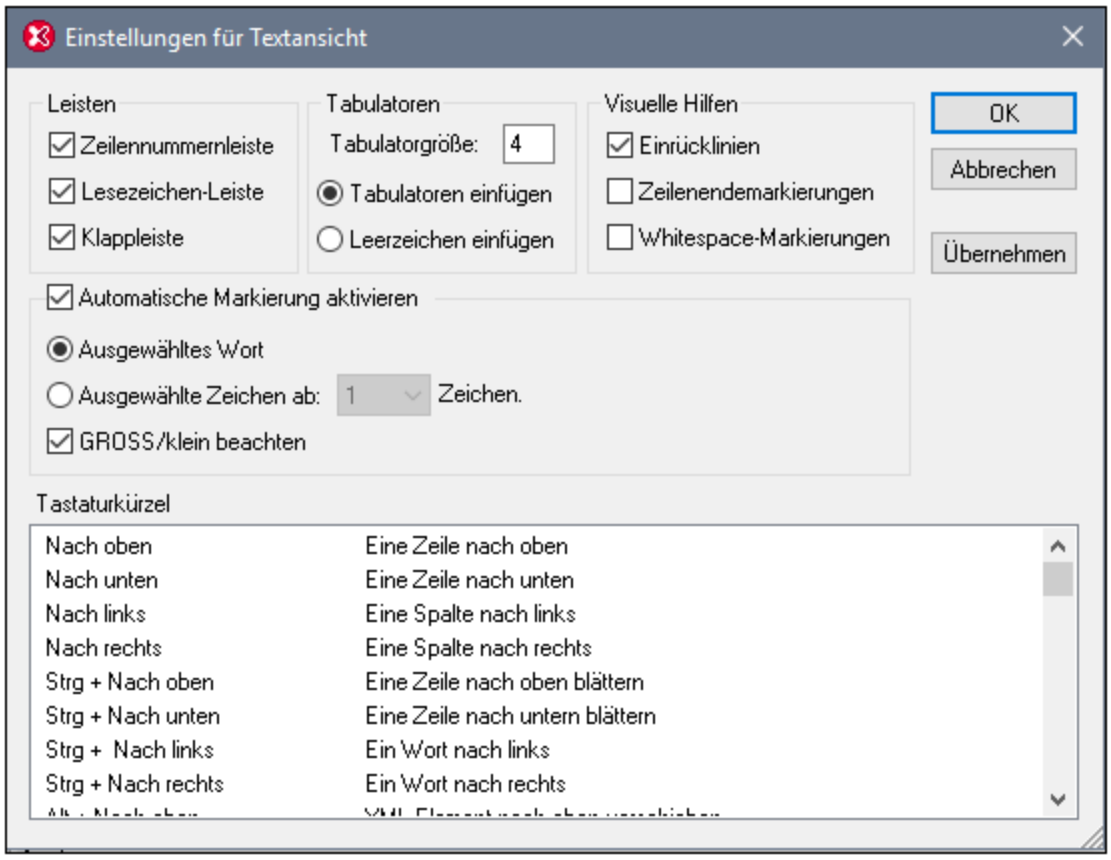
Wenn die Textmarkierung im Dialogfeld "Einstellungen für Textansicht" (**Ansicht | Einstellungen für Textansicht**[1506]) aktiviert ist, werden alle Übereinstimmungen mit einer vom Benutzer definierten Auswahl im Dokument in der Textansicht markiert. Der ausgewählte Text wird hellblau markiert und die Übereinstimmungen werden hellorange markiert (*siehe Abbildung unten*). Die Auswahl und die Übereinstimmungen werden in der Bildlaufleiste durch graue Markierungsquadrate gekennzeichnet. Beachten Sie außerdem, dass die aktuelle Cursorposition in der Bildlaufleiste durch die blaue Cursormarkierung angezeigt wird.

© 2016-2022 Altova GmbH



```
<expense-report xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ExpReport.xsd" currency="USD" detailed="false"
total-sum="556.9">
    <Person>
        <First>Fred</First>
        <Last>Landis</Last>
        <Title>Project Manager</Title>
        <Phone>123-456-7890</Phone>
        <Email>f.landis@nanonull.com</Email>
    </Person>
    <expense-item type="Lodging" expto="Sales">
        <Date>2003-01-01</Date>
        <expense>122.11</expense>
    </expense-item>
    <expense-item type="Lodging" expto="Development">
        <Date>2003-01-02</Date>
        <expense>122.12</expense>
        <description>Played penny arcade</description>
    </expense-item>
    <expense-item type="Lodging" expto="Marketing">
        <Date>2003-01-02</Date>
        <expense>299.45</expense>
        <description>Treated Clients</description>
    </expense-item>
    <expense-item type="Entertainment" expto="Development">
        <Date>2003-01-02</Date>
        <expense>13.22</expense>
        <Misc misctype="TeamBuilding"/>
        <description>Bought signed XMLSpy Handbook</description>
    </expense-item>
</expense-report>
```

Um die Textmarkierungsfunktion zu aktivieren, aktivieren Sie im Dialogfeld "Einstellungen für Textansicht" (**Ansicht | Einstellungen für Textansicht**[1506], *Abbildung unten*) das Kontrollkästchen *Automatische Markierung aktivieren*. Als Auswahl kann ein ganzes Wort oder eine festgelegte Anzahl von Zeichen definiert werden. Sie können außerdem definieren, ob die Groß- und Kleinschreibung berücksichtigt werden soll.



Beachten Sie die folgenden Punkte:

- Bei Auswahl einer Anzahl von Zeichen können Sie die Mindestanzahl der Zeichen ab dem ersten Zeichen in der Auswahl, für die eine Übereinstimmung gefunden werden soll, definieren. So können Sie z.B. nach zwei oder mehr Zeichen suchen. In diesem Fall wird für ein einziges ausgewähltes Zeichen keine Übereinstimmung gesucht, für eine Auswahl bestehend aus zwei oder mehr Zeichen werden Übereinstimmungen gefunden. Wenn Sie in diesem Fall den Buchstaben `t` auswählen, werden keine Übereinstimmungen angezeigt; wenn Sie `ty` auswählen, werden alle `ty`-Treffer angezeigt; wenn Sie `typ` auswählen, werden alle Übereinstimmungen mit `typ` angezeigt, usw.
- Bei Wortsuchen werden folgende Einträge als separate Wörter behandelt: Elementnamen (ohne spitze Klammern), die spitzen Klammern von Element-Tags, Attributnamen und Attributwerte ohne Anführungszeichen.

Anmerkung: Die Übereinstimmung von Element-Start- und End-Tags 151 ist eine separate Funktion, die von der Einstellung *Automatische Markierung aktivieren* nicht betroffen ist. Im Dialogfeld "Optionen" im Abschnitt "Textschriftarten" 1623 (**Extras | Optionen**) können Sie Farben für die Übereinstimmung von Element-Start- und End-Tags definieren.

Dokumentübersicht in der Bildlaufleiste

Die Bildlaufleiste bietet die folgenden Funktionen:

© 2016-2022 Altova GmbH

- Sie zeigt die Größe der folgenden Elemente im Vergleich zueinander an (*siehe Abbildung unten*): (i) das gesamte Dokument (Bildlaufleiste); (ii) der gerade im Fenster angezeigte Dokumentabschnitt (Ziehpunkt); (iii) der aktuell ausgewählte Text (blauer Balken), falls vorhanden und (iv) die aktuelle Cursorposition (Cursor-Markierung).
- Über die Bildlaufleiste können Sie folgendermaßen im Dokument navigieren: (i) durch Ziehen des Ziehpunkts der Bildlaufleiste nach oben und unten oder (ii) durch Klicken auf den **Nach oben**- und **Nach unten**-Pfeil (in der Abbildung unten grün umrandet).



Beachten Sie die folgenden Punkte:

- Die Länge der Bildlaufleiste entspricht der Länge des gesamten Dokuments.
- Wenn nur ein Teil des Dokuments in das Fenster passt, so entspricht der Ziehpunkt der Bildlaufleiste (*siehe Abbildung oben*) diesem Teil des Fensters. Sie können den Ziehpunkt auch nach oben und unten ziehen, um andere Teile des Dokuments im Fenster sichtbar zu machen. Der Ziehpunkt stellt also das Fenster dar und Sie ziehen das Fenster im Dokument nach oben und unten, um das Dokument zu sehen.
- Der aktuell ausgewählte Text wird in der Bildlaufleiste durch den blauen Balken markiert. Die Größe des blauen Balkens im Vergleich zur Bildlaufleiste entspricht der Größe der Textauswahl im Vergleich zur Größe des gesamten Dokuments. Wenn der ausgewählte Text nicht mehr als eine Zeile umfasst, ist der blaue Balken nicht zu sehen.
- Die Cursorposition wird durch eine dunkelblaue Cursormarkierung angezeigt. Die relative Position der Cursormarkierung in der Bildlaufleiste entspricht der relativen Position des Cursors im Dokument.

Gehe zu Zeile/Zeichen

Mit Hilfe dieses Befehls im Menü **Ansicht** und der Schaltfläche in der Symbolleiste "Text" können Sie zu einer bestimmten Zeile und einem bestimmten Zeichen im Dokumenttext navigieren.

4.2.5 Suchen und Ersetzen

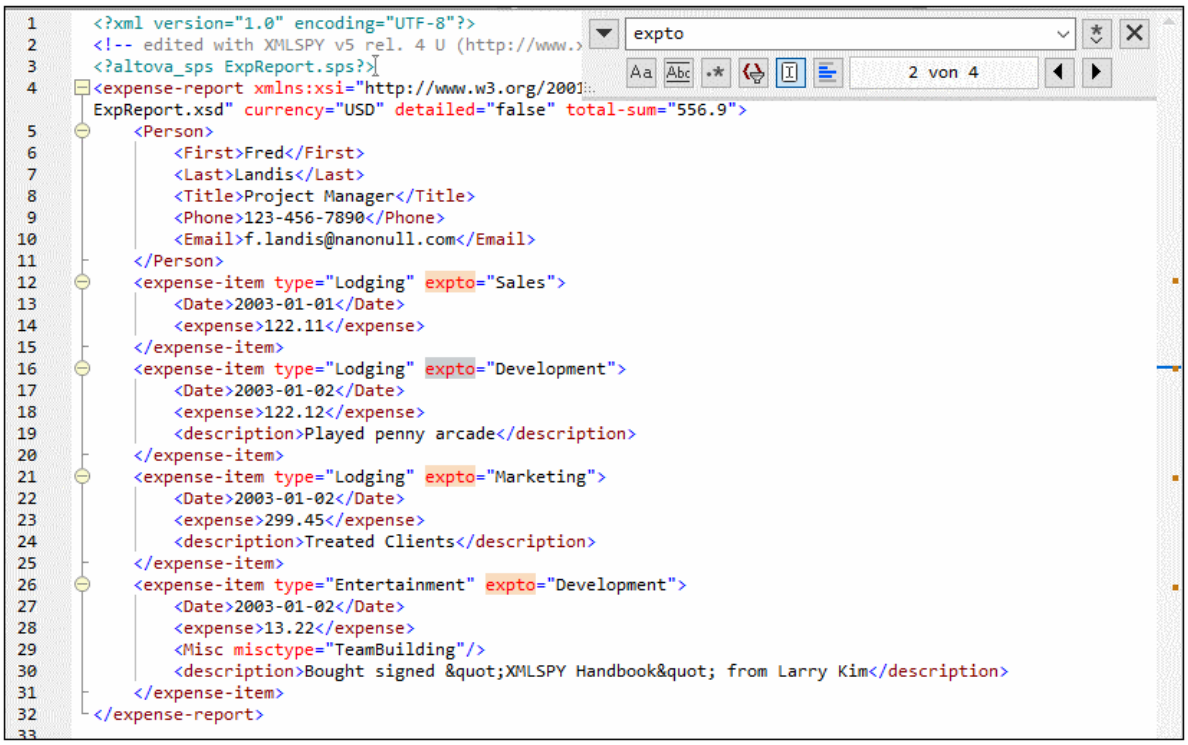
In diesem Abschnitt:

- Suchen nach Text in einem Dokument 158
- Suchoptionen 159
- Ersetzen von Text 160
- Verwendung von Regular Expressions 160
- Beispiele für Regular Expressions 162

Suchen nach Text in einem Dokument

Drücken Sie **Strg+F** (oder wählen Sie den Menübefehl **Bearbeiten | Suchen**), um das Suchdialogfeld (*rechts oben in der Abbildung unten*) aufzurufen. Sie können im gesamten Dokument oder im ausgewählten Text nach dem Begriff suchen, den Sie in das Dialogfeld eingeben.

- Geben Sie den gewünschten String ein oder verwenden Sie die Auswahlliste, um einen String aus den letzten 10 Such-Strings auszuwählen.
- Nach Eingabe bzw. Auswahl des gewünschten String werden alle Übereinstimmungen markiert und die Position der Treffer wird durch beige Markierungen in der Bildlaufleiste gekennzeichnet (*siehe Abbildung*).
- Die gerade ausgewählte Übereinstimmung hat eine andere Markierungsfarbe als die anderen Übereinstimmungen. Ihre Position wird in der Bildlaufleiste durch eine dunkelblaue Cursormarkierung gekennzeichnet.
- Unterhalb des Suchbegrifffelds wird die Gesamtzahl der Treffer sowie die Indexposition des aktuell ausgewählten Treffers aufgelistet (*siehe Abbildung unten*). `2 von 4` zeigt z.B. an, dass gerade der zweite von vier Treffern ausgewählt ist.
- Sie können nun durch Auswahl der beiden Schaltflächen **Gehe zu vorherigem Ergebnis** (**Umschalt+F3**) und **Gehe zu nächstem Ergebnis** (**F3**) auf der rechten Seite in beiden Richtungen von einem Treffer zum nächsten springen.



- Um zwischen dem Such- und Ersetzungsmodus zu wechseln, klicken Sie auf den Abwärtspfeil auf der linken Seite.

© 2016-2022 Altova GmbH

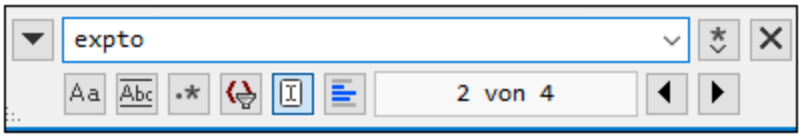
- Um das Dialogfeld zu schließen, klicken Sie auf die **Schließen**-Schaltfläche rechts oben.

Beachten Sie die folgenden Punkte:

- Das Suchdialogfeld hat keinen Modus, d.h. es bleibt offen, solange Sie die Textansicht verwenden.
- Wenn vor dem Öffnen des Dialogfelds Text ausgewählt ist, so wird der ausgewählte Text automatisch in das Suchfeld eingefügt.
- Um einen String innerhalb einer Auswahl zu suchen, gehen Sie folgendermaßen vor: (i) Markieren Sie die Auswahl; (ii) Aktivieren Sie die Option *In Auswahl suchen*[159], um die Auswahl zu sperren; (iii) Geben Sie den Suchbegriff ein. Um den Suchbegriff in einer anderen Auswahl zu suchen, heben Sie die Sperre der Auswahl durch Deaktivieren der Option *In Auswahl suchen*[159] auf, wählen Sie einen anderen Bereich aus und aktivieren Sie die Option *In Auswahl suchen*[159] wieder.
- Nachdem das Suchdialogfeld geschlossen wurde, können Sie die aktuelle Suche durch Drücken von **F3** (Weitersuchen) bzw. (**Umschalt+F3**) für eine Suche nach oben wiederholen. In diesem Fall wird das Suchdialogfeld wieder angezeigt.

Suchoptionen

Über Schaltflächen unterhalb des Suchfelds (*siehe Abbildung unten*) stehen Optionen zum Festlegen der Suchkriterien zur Verfügung. Wenn eine Option aktiv ist, ändert sich die Schaltflächenfarbe in Blau (*siehe erste Option (Groß-/Kleinschreibung) in der Abbildung unten*).



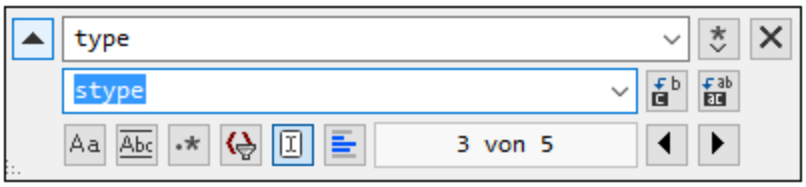
Die folgenden Optionen stehen zur Auswahl:

- *GROSS/klein beachten*: Wenn die Schaltfläche aktiv ist, wird die Groß- und Kleinschreibung bei der Suche berücksichtigt (`Address` ist nicht gleich `address`).
- *Ganzes Wort*: Nur die exakte Wortentsprechung im Text wird gefunden. So wird z.B. bei Eingabe von `fit` bei Aktivierung der Option *ganzes Wort* nur das Wort `fit` gefunden; `fit` in `fitness` wird z.B. nicht gefunden.
- *Regular Expression*: Wenn die Option aktiv ist, wird der Suchbegriff als Regular Expression gelesen. Eine Beschreibung zur Verwendung von Regular Expressions finden Sie weiter unten unter *Regular Expressions*[160].
- *Ergebnisse filtern:* Wählen Sie eine oder mehrere Dokumentkomponenten, in denen gesucht werden soll.
- *Anker suchen:* Bei Eingabe eines Suchbegriffs werden die Übereinstimmungen im Dokument markiert und eine dieser Übereinstimmungen wird als aktuell ausgewählte markiert. Mit der Ein/Aus-Schaltfläche *Anker suchen* legen Sie fest, ob diese erste aktuelle Auswahl relativ zur Cursorposition gemacht werden soll. Wenn *Anker suchen* aktiviert ist, so ist die erste aktuell ausgewählte Übereinstimmung die nächste Übereinstimmung ab der aktuellen Cursorposition. Wenn *Anker suchen* deaktiviert ist, so ist die erste aktuell ausgewählte Übereinstimmung die erste Übereinstimmung im Dokument ab dem Dokumentanfang gesehen.
- *In Auswahl suchen:* Wenn diese Option aktiv ist, wird der aktuell ausgewählte Text gesperrt und die Suche auf die Auswahl eingeschränkt. Andernfalls wird das gesamte Dokument durchsucht. Bevor Sie einen neuen Textbereich auswählen, heben Sie die aktuelle Auswahl durch Deaktivieren der Option *In Auswahl suchen* auf.

Ersetzen von Text

Um einen Text-String zu ersetzen, gehen Sie folgendermaßen vor:

1. Drücken Sie **Strg+H** (oder wählen Sie den Menübefehl **Bearbeiten | Ersetzen**) um das Ersetzungsdialogfeld (*Abbildung unten*) aufzurufen. (Alternativ dazu können Sie auch durch Klick auf den Abwärtspfeil links oben im Dialogfeld zum Ersetzungsmodus des Suchen/Ersetzen-Dialogfelds wechseln).



2. Geben Sie den zu ersetzenden String in das Suchfeld und den neuen String in das Ersetzungsfeld ein. Die Anzahl der zu ersetzenden Treffer und der Index des gerade ausgewählten Treffers werden unterhalb des Ersetzungsfelds angezeigt. Die Position der Treffer wird außerdem auch in der Bildlaufleiste durch beige Markierungen gekennzeichnet (*siehe Suchen nach Text in einem Dokument*[158] *weiter oben*). In der Abbildung oben sehen Sie z.B., dass für den String `type` fünf Übereinstimmungen gefunden wurden und dass gerade die dritte dieser Übereinstimmungen ausgewählt ist..
3. Rechts vom Ersetzungsfeld befinden sich die Schaltflächen **Nächstes ersetzen** und **Alles ersetzen**. Wenn Sie auf **Nächstes ersetzen** klicken, geschieht Folgendes: (i) Wenn sich der Cursor neben einem oder innerhalb eines Treffers befindet, so wird dieser ersetzt; (ii) wenn sich der Cursor außerhalb eines Treffers befindet, springt er zum nächsten Treffer. Klicken Sie auf **Nächstes ersetzen**, um diese Übereinstimmung zu ersetzen. Klicken Sie auf **Alles ersetzen,** um alle Treffer zu ersetzen.

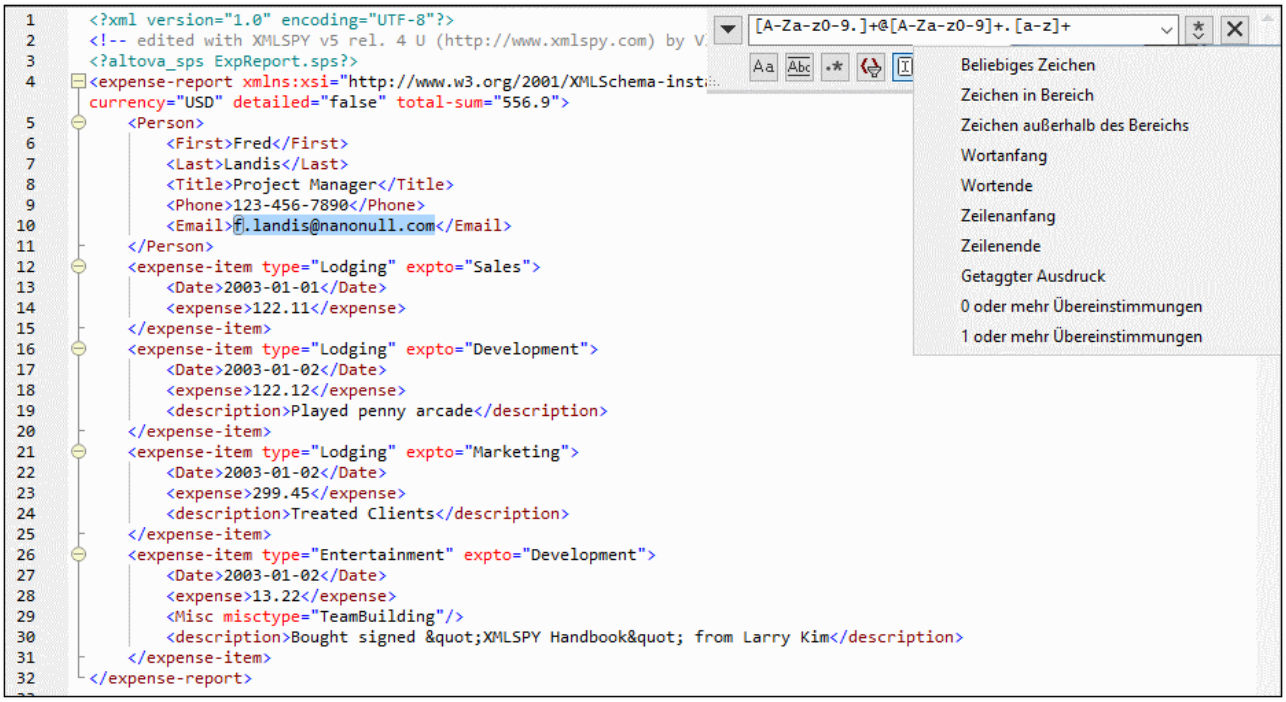
Beachten Sie die folgenden Punkte:

- Um Text innerhalb einer Auswahl - und nicht im gesamten Dokument - zu ersetzen, gehen Sie folgendermaßen vor: (i) Markieren Sie die Auswahl; (ii) Aktivieren Sie die Option *In Auswahl suchen*[159], um die Auswahl zu sperren; (iii) Geben Sie den Such- und den Ersetzungsstring ein; (iii) Klicken Sie je nach Bedarf auf **Nächstes ersetzen** oder **Alles ersetzen**. Um Text in einer anderen Auswahl zu ersetzen, heben Sie die aktuelle Auswahl durch Deaktivieren der Option *In Auswahl suchen*[159] auf, markieren Sie anschließend die neue Auswahl und aktivieren Sie die Option *In Auswahl suchen*[159], um die neue Auswahl zu sperren.
- Um eine Ersetzungsaktion rückgängig zu machen, drücken Sie **Strg+Z** oder wählen Sie den Befehl **Bearbeiten | Rückgängig**.

Verwendung von Regular Expressions

Sie können zum Suchen eines Text-String Regular Expressions (regex) verwenden. Aktivieren Sie dazu zuerst die Option *Regular Expression* (siehe *Suchoptionen*[159] oben). Damit geben Sie an, dass der Text im Suchfeld als Regular Expression ausgewertet werden soll. Geben Sie als Nächstes die Regular Expression in das Suchfeld ein. Hilfe zur Erstellung von Regular Expressions erhalten Sie durch Klick auf die Schaltfläche **Regular Expression Builder** rechts vom Suchfeld (*siehe Abbildung unten*). Klicken Sie auf einen Eintrag im Builder, um die/das entsprechende(n) regex-Metazeichen in das Suchfeld einzugeben. In der Abbildung unten sehen Sie eine einfache Regular Expression zum Suchen von E-Mail-Adressen. Eine kurze Beschreibung zu Metazeichen finden Sie im Abschnitt *Regular Expression-Metazeichen*[161] weiter unten.

© 2016-2022 Altova GmbH



Regular Expression Metazeichen

In der nachstehenden Liste sehen Sie eine Auflistung von Regular Expression-Metazeichen.

`.`	Steht für jedes beliebige Zeichen. Dies ist ein Platzhalter für ein einzelnes Zeichen.
`(`	Markiert den Beginn eines markierten Ausdrucks.
`)`	Markiert das Ende eines markierten Ausdrucks.
`(abc)`	Die Metazeichen `(` und `)` markieren Beginn und Ende eines markierten Ausdrucks. Markierte Ausdrücke eignen sich dazu, eine gesuchte Region zu markieren ("sich diese zu merken"), um diese später referenzieren zu können (Rückreferenz). Es können bis zu neun Unterausdrücke markiert werden (und später im Feld "Suchen" oder "Ersetzen" rückreferenziert werden). So wird etwa mit `(the) \1` der String `the the` gefunden. Dieser Ausdruck erklärt sich folgendermaßen: Suche den String "the" (und merke ihn Dir als markierte Region), gefolgt von einem Leerzeichen, gefolgt von einer Rückreferenz auf die zuvor gesuchte markierte Region.
`\n`	`n` ist eine **Variable**, die Ganzzahlwerte von `1` bis `9` haben kann. Der Ausdruck bezieht sich auf die erste bis neunte markierte Region bei der Ersetzung. Lautet der Suchstring beispielsweise `Fred([1-9])XXX` und der Ersetzungsstring `Sam\1YYY`, so bedeutet dies, dass sich im Suchstring ein markierter Ausdruck befindet, der (implizit) mit der Zahl `1` indiziert ist; im Ersetzungsstring wird der markierte Ausdruck mit `\1` referenziert. Wenn der Such- und Ersetzungsbefehl auf `Fred2XXX` angewendet wird, würde `Sam2YYY` generiert.
`\<`	Steht für den Beginn eines Worts.

\>	Steht für das Ende eines Worts.
\x	Damit können Sie ein Zeichen x verwenden, das sonst eine spezielle Bedeutung hätte. So würde z.B. \[als [und nicht als Beginn einer Zeichenmenge interpretiert werden.
[...]	Kennzeichnet eine *Zeichenmenge*, [abc] z.B. steht für jedes der Zeichen a, b oder c. Sie können auch Bereiche angeben, z.B. [a-z] für alle Kleinbuchstaben.
[^...]	Die invertierte Zeichenmenge. [^A-Za-z] z.B. steht für jedes Zeichen mit Ausnahme alphabetischer Zeichen.
^	Steht für den Zeilenanfang (es sei denn dieses Zeichen wird innerhalb einer Menge verwendet, *siehe oben*).
$	Steht für das Zeilenende. Beispiel: A+$ findet ein oder mehrere A's am Ende der Zeile.
*	Steht für 0 oder öfter. Mit Sa*m werden z.B. Sm, Sam, Saam, Saaam usw. gefunden.
+	Steht für 1 oder öfter. Mit Sa+m werden z.B. Sam, Saam, Saaam usw. gefunden.

Darstellung von Sonderzeichen
Beachten Sie die folgenden Ausdrücke.

\r	Wagenrücklauf (Carriage Return = CR). Zum Suchen oder Erstellen einer neuen Zeile können Sie entweder CR (**\r**) oder LF (**\n**) verwenden.
\n	Neue Zeile (LF). Zum Suchen oder Erstellen einer neuen Zeile können Sie entweder CR (**\r**) oder LF (**\n**) verwenden.
\t	Tabulatorzeichen
****	Verwenden Sie diese Zeichen, um Zeichen, die in Regular Expressions vorkommen, mit einem Escape zu versehen, z.B.: **\\\n**

Beispiele für Regular Expressions

In diesem Beispiel wird gezeigt, wie Sie Text mit Hilfe von Regular Expressions suchen und ersetzen können. In vielen Fällen ist das Suchen und Ersetzen von Text unkompliziert und es werden dafür keine Regular Expressions benötigt. Es gibt jedoch Fälle, in denen Sie Text auf eine Art und Weise bearbeiten müssen, wie es mit der normalen Such- und Ersetzungsoperation nicht möglich ist, z.B. wenn Sie eine XML-Datei mit mehreren tausend Zeilen haben, in der Sie bestimmte Elemente in einer einzigen Operation umbenennen müssen, ohne dabei den darin enthaltenen Inhalt zu ändern. Ein weiteres Beispiel: Sie müssen die Reihenfolge mehrerer Attribute eines Elements ändern. Hier sind Regular Expressions hilfreich, da Sie sich damit eine Menge Arbeit ersparen.

Beispiel 1: Umbenennen von Elementen

Der unten gezeigte XML-Beispielcode enthält eine Liste von Büchern. Angenommen, Sie möchten das Element <Category> jedes Buchs durch <Genre> ersetzen. Ein der Methoden, mit denen Sie dies erreichen, ist mit Hilfe von Regular Expressions.

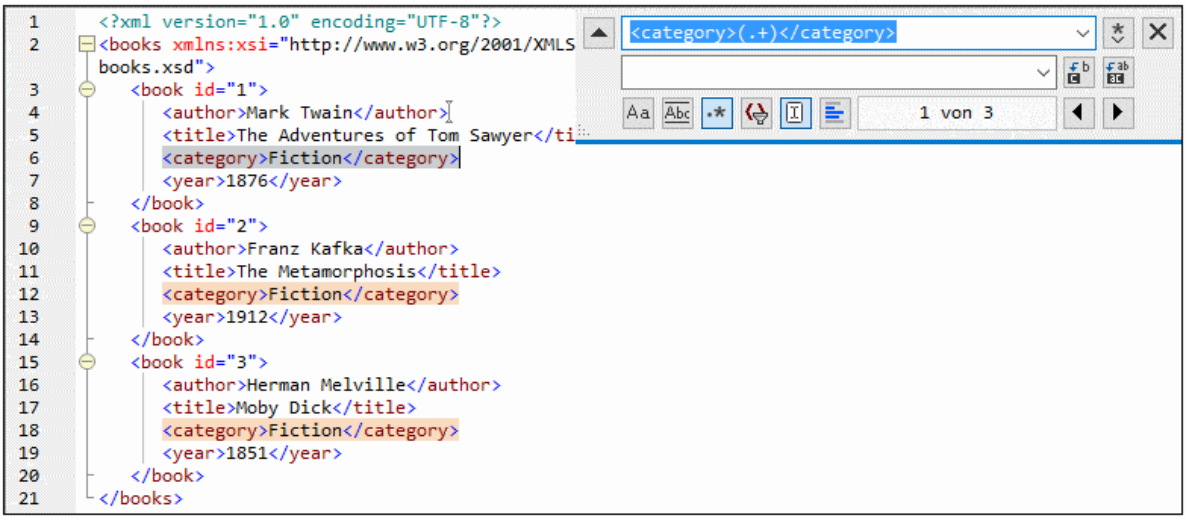
```
<?xml version="1.0" encoding="UTF-8"?>
<books xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="books.xsd">
   <book id="1">
      <author>Mark Twain</author>
```

 © 2016-2022 Altova GmbH

```
      <title>The Adventures of Tom Sawyer</title>
      <category>Fiction</category>
      <year>1876</year>
   </book>
   <book id="2">
      <author>Franz Kafka</author>
      <title>The Metamorphosis</title>
      <category>Fiction</category>
      <year>1912</year>
   </book>
   <book id="3">
      <author>Herman Melville</author>
      <title>Moby Dick</title>
      <category>Fiction</category>
      <year>1851</year>
   </book>
</books>
```

Gehen Sie dazu folgendermaßen vor:

1. Drücken Sie **Strg+H,** um das Such- und Ersetzungsdialogfeld aufzurufen.
2. Klicken Sie auf die Schaltfläche **Regular Expressions verwenden** .
3. Geben Sie in das Suchfeld den folgenden Text ein: `<category>(.+)</category>` . Mit dieser Regular Expression werden alle `category`-Elemente gefunden und markiert.



Um den Text im Inhalt der einzelnen Elemente zu suchen (der im Vorhinein nicht bekannt ist), haben wir den getaggten Ausdruck `(.+)` verwendet. Der getaggte Ausdruck `(.+)` bedeutet, "finde eine oder mehrere Instanzen eines beliebigen Zeichens, welches `.+` entspricht, und merke Dir diese Übereinstimmung". Wie Sie im nächsten Schritt sehen, benötigen wird die Referenz auf den getaggten Ausdruck später.

4. Geben Sie in das Ersetzungsfeld den folgenden Text ein: `<genre>\1</genre>` . Mit dieser Regular Expression wird der Ersetzungstext definiert. Beachten Sie, dass darin eine Rückreferenz `\1` auf den vorhin im Suchfeld getaggten Ausdruck verwendet wird, d.h. `\1` bedeutet in diesem Zusammenhang

"der Textinhalt der aktuellen <category>-Elementübereinstimmung".

5. Klicken Sie auf **Alle ersetzen** und überprüfen Sie die Ergebnisse. Alle `category`-Elemente wurden nun, wie geplant, in `genre` umbenannt.

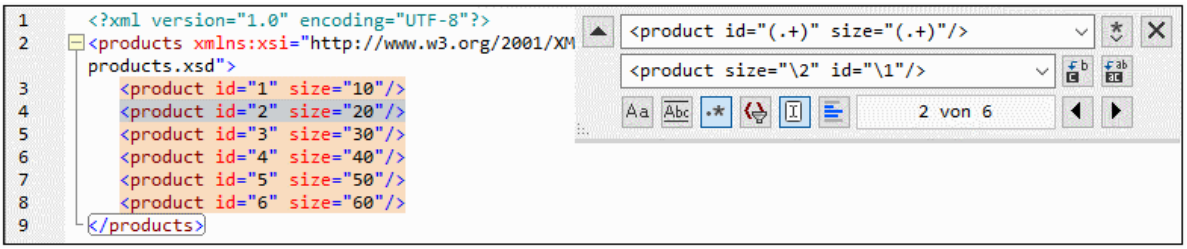
Beispiel 2: Ändern der Reihenfolge von Attributen

Der unten gezeigte XML-Beispielcode enthält eine Liste von Produkten. Jedes Produktelement hat zwei Attribute: `id` und `size`. Angenommen, Sie möchten die Reihenfolge der Attribute `id` und `size` in allen `product`-Elementen ändern (d.h. das Attribut `size` soll vor `id` stehen). Eine der Lösungsmethoden dafür ist mittels Regular Expressions.

```
<?xml version="1.0" encoding="UTF-8"?>
<products xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="products.xsd">
   <product id="1" size="10"/>
   <product id="2" size="20"/>
   <product id="3" size="30"/>
   <product id="4" size="40"/>
   <product id="5" size="50"/>
   <product id="6" size="60"/>
</products>
```

Gehen Sie dazu folgendermaßen vor:

1. Drücken Sie **Strg+H,** um das Such- und Ersetzungsdialogfeld aufzurufen.
2. Klicken Sie auf die Schaltfläche **Regular Expressions verwenden** .
3. Geben Sie ins Suchfeld den folgenden Ausdruck ein: `<product id="(.+)" size="(.+)"/>` . Mit dieser Regular Expression wird ein product-Element im XML-Dokument gefunden. Beachten Sie, dass zwei Mal ein getaggter Ausdruck `(.+)` verwendet wird, um den Wert der einzelnen Attribute (der im Vorhinein nicht bekannt ist) zu finden. Mit dem getaggten Ausdruck `(.+)` wird der Wert jedes einzelnen Attributs gefunden (Der Wert wird als eine oder mehrere Instanzen eines beliebigen Zeichens angenommen, also `.+`).
4. Geben Sie in das Ersetzungsfeld den folgenden Ausdruck ein: `<product size="\2" id="\1"/>` . Diese Regular Expression enthält den Ersetzungstext für jedes gefundene product-Element. Beachten Sie, dass darin zwei Referenzen `\1` und `\2` verwendet werden. Diese entsprechen den getaggten Ausdrücken aus dem Suchfeld. Anders ausgedrückt, bedeutet `\1` "der Wert des Attributs `id`" und `\2` "der Wert des Attributs `size`".



6. Klicken Sie auf **Alle ersetzen** und überprüfen Sie die Ergebnisse. Alle `product`-Elemente wurden nun aktualisiert, so dass das Attribut `size` jetzt vor dem Attribut `id` steht.

 © 2016-2022 Altova GmbH

4.2.6 Eingabehilfen in der Textansicht

Welche Eingabehilfe in der Textansicht zur Verfügung stehen, ist von der bearbeiteten Dokumentart abhängig. Unterhalb finden Sie eine Liste der Eingabehilfen für die gebräuchlichsten Dokumenttypen. Im Folgenden finden Sie eine Beschreibung 165 der allgemeinen Verwendung der Eingabehilfen. Zusätzliche Funktionen für bestimmte Dokumenttypen finden Sie in den Abschnitten zu den entsprechenden Dokumenttypen.

- *XML:* Elemente (*Abbildung unten*), Attribute, Entities



- *HTML:* Elemente, Attribute, Entities
- *CSS:* CSS-Übersicht, CSS-Eigenschaften, HTML-Elemente
- *DTD:* Keine
- *XQuery:* XQuery-Schlüsselwörter, XQuery-Variablen, XQuery-Funktionen
- *WSDL:* Übersicht, Details
- *Text:* Entities

Beachten Sie, dass es sich bei einer Reihe von Dokumentarten wie XSD, XSLT, XHTML und RDF im Grunde um XML-Dokumente handelt. Daher stehen hierfür die Eingabehilfen "Elemente", "Attribute" und "Entities" zur Verfügung.

Anzeige und Verwendung von Einträgen der Eingabehilfen

Die verschiedenen Einträge in den verschiedenen Eingabehilfen haben unterschiedliche Farben. Die Farbcodierung ist in der Dokumentation zu den Eingabehilfen der jeweiligen Dokumentart erklärt. Beachten Sie die folgenden Punkte zu Eingabehilfen:

- Die Eingabehilfen sind kontextsensitiv und zeigen Einträge an, die an dieser Stelle eingefügt werden können.
- Wenn das Element an der ausgewählten Stelle (oder einer anderen gültigen Stelle) bereits eingefügt wurde, und dort nicht noch einmal eingefügt werden darf (z.B. ein XML-Attribut), wird es grau angezeigt.
- Wenn das Element obligatorisch ist, wird ein Ausrufezeichen daneben angezeigt.
- Um den Eintrag einer Eingabehilfe im Text an der Cursorposition einzufügen, doppelklicken Sie in der Eingabehilfe auf den Eintrag.
- Wenn ein Element über die Eingabehilfe "Elemente" eingefügt wurde, werden sein Start- und End-Tag in den Dokumenttext eingefügt. Auch obligatorische Elemente werden eingefügt, wenn diese Option im Dialogfeld **Optionen** (**Extras | Optionen | Bearbeiten**) definiert wurde.
- Wenn ein Attribut über die Eingabehilfe "Attribute" eingefügt wird, wird das Attribut mit einem Ist-gleich-Zeichen und Anführungszeichen, in die der Attributwert gesetzt wird, eingefügt. Der Cursor wird zwischen die Anführungszeichen gesetzt, sodass Sie die Attributwert sogleich eingeben können.

Anmerkung: Bei großen Dateien können die Autokomplettierung und die Eingabehilfen deaktiviert werden, um ein schnelleres Laden und eine schnellere Bearbeitung zu gewährleisten. Der Grenzwert kann vom Benutzer definiert werden. Nähere Informationen dazu finden Sie im Abschnitt "Referenz" unter Optionen | Bearbeitung[1609].

4.2.7 Geteilte Ansicht

In der geteilten Ansicht wird das Hauptfenster der Textansicht in zwei Bereiche geteilt und das aktive Dokument wird in beiden Ansichten angezeigt. Sie können in jeder Ansicht separat navigieren, wodurch Sie verschiedene Teile des Dokuments Seite an Seite nebeneinander sehen können. Die Änderungen, die Sie vornehmen, werden jedoch am zugrunde liegenden Dokument gemacht und in beiden Ansichten angezeigt.

Wechseln zwischen der geteilten Ansicht und einer einzigen Ansicht

So zeigen Sie das **aktive Dokument** in einer geteilten Ansicht an:

- *Horizontale Teilung:* Ziehen Sie das Symbol für die horizontale Teilung in der rechten oberen Ecke (in der Abbildung unten grün umrandet) nach unten.
- *Vertikale Teilung:* Ziehen Sie das Symbol für die vertikale Teilung in der linken unteren Ecke (in der Abbildung unten blau umrandet) nach rechts.



Um wieder zur einfachen Ansicht zurückzukehren, wählen Sie eine der folgenden Methoden:

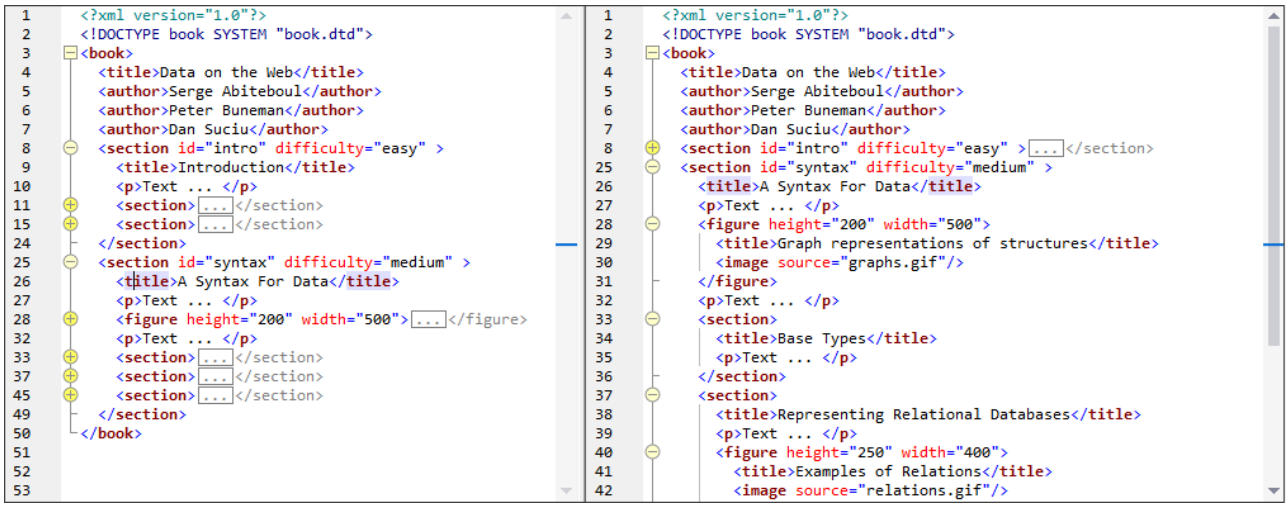
- Doppelklicken Sie auf die Teilungsleiste oder
- Schieben Sie die Teilungsleiste an den Rand des Hauptfensters.

 © 2016-2022 Altova GmbH

Anmerkung: Eine geteilte Ansicht wird für jedes Dokument einzeln erstellt.

Navigation und Bearbeitung in der geteilten Ansicht

Der wichtigste Vorteil bei der Arbeit in der geteilten Ansicht ist, dass Sie verschiedene Bereiche eines langen Dokuments Seite an Seite nebeneinander anzeigen und das Dokument währenddessen in beiden Ansichten bearbeiten können. In der Abbildung unten sehen Sie ein Dokument in einer vertikal geteilten Ansicht.



Beachten Sie die folgenden Punkte:

- Alle Anzeigefunktionen[148] (wie Zeilennummerierung und Klappleiste), Bearbeitungsfunktionen[151], Navigationsfunktionen[154], usw., die in der Einzelansicht eines Dokuments zur Verfügung stehen, stehen in beiden Ansichten der geteilten Ansicht zur Verfügung *(siehe Abbildung oben)*.
- Sie können in jeder Ansicht der geteilten Ansicht separat scrollen und navigieren.
- Sie können die Klappleiste in jeder Ansicht separat verwenden.
- Alle Bearbeitungsaktionen im Dokument, darunter auch alle Eingabehilfenaktionen, werden in beiden Ansichten angezeigt.

4.2.8 Tastaturkürzel für die Textansicht

Die folgende Liste zeigt die Standard-Tastaturkürzel häufig verwendeter Befehle in der Textansicht. Sie können die Standard-Tastaturkürzel im *Dialogfeld "Anpassen" auf dem Register "Tastatur"*[1589] ändern.

Befehle in der Textansicht

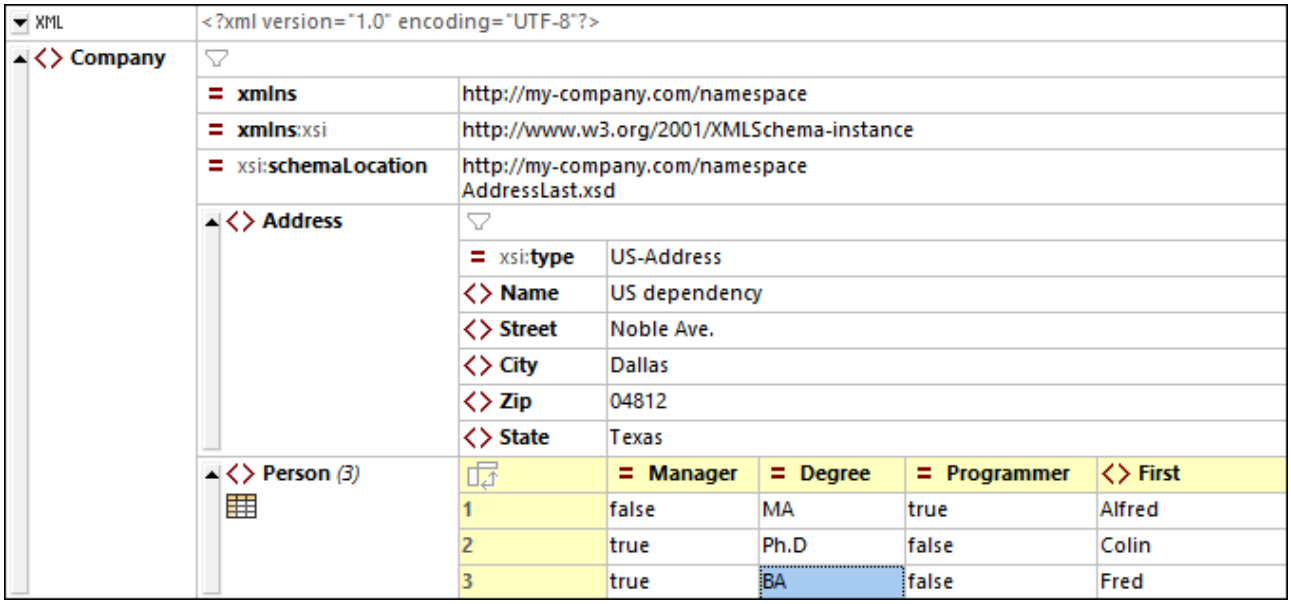
STRG + E	Zwischen Start- und End-Tag hin- und herspringen
STRG + Umschalt + E	Element, das den Cursor enthält, auswählen
STRG + Alt + E	Gehe zum übergeordneten Element
STRG + "+"	Vergrößern
STRG + "–"	Verkleinern

STRG + 0	Zoom zurücksetzen
STRG + Mausrad vorwärts	Vergrößern
STRG + Mausrad rückwärts	Verkleinern

© 2016-2022 Altova GmbH

4.3 Grid-Ansicht

Die Grid-Ansicht steht für XML-Dokumente, JSON-Dokumente und DTD-Dateien zur Verfügung. (*In der Abbildung unten sehen Sie die Grid-Ansicht eines XML-Dokuments.*) Die Grid-Ansicht zeigt die hierarchische Struktur von Dokumenten in Form von verschachtelten Behältern, die einfach erweitert und reduziert werden können, um ein klares Bild der Dokumentenstruktur zu erhalten. Dadurch können in der Grid-Ansicht sowohl Inhalt als auch Struktur einfach bearbeitet werden.

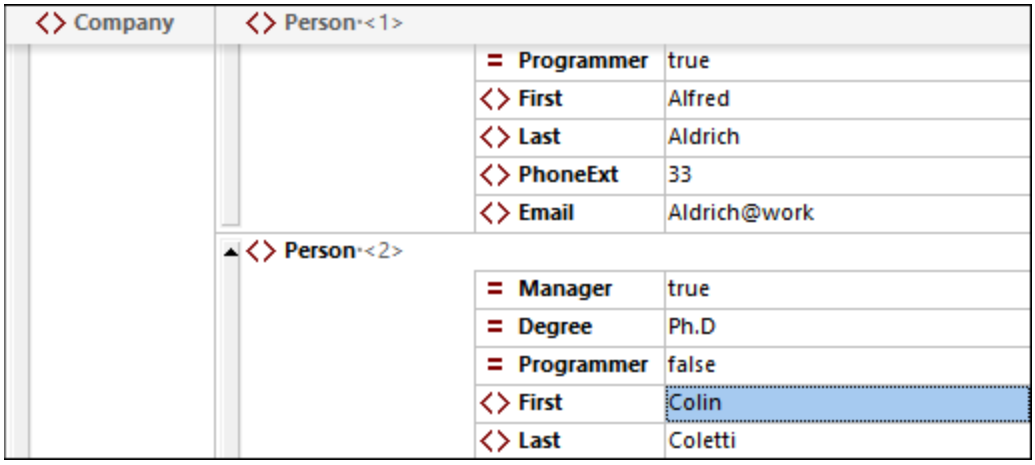


Nähere Informationen zum Arbeiten in der Grid-Ansicht finden Sie in den folgenden Abschnitten:

- XML in der Grid-Ansicht 296
- JSON-Grid-Ansicht 690
- DTDs 448

Bildlaufleistenüberschriften

Wenn Sie in einem XML-, JSON- oder DTD-Dokument in der Grid-Ansicht nach unten scrollen und der Node der obersten Ebene im Ansichtsfenster nicht mehr zu sehen ist, erscheint am oberen Rand der Grid-Ansicht eine Überschriftsleiste, in der dieser Node der obersten Ebene angezeigt wird. Wenn Sie noch weiter nach unten scrollen, werden in der Überschriftsleiste weitere Vorfahren-Elemente des derzeit im oberen Bereich des Fensters befindlichen Node angezeigt. So werden etwa in der Abbildung unten in der Überschriftsleiste die Vorfahren des `/Company/Person[1]/@Programmer`-Node angezeigt.



Wenn Sie auf einen der Nodes in der Überschriftsleiste klicken, wird dieser Node ausgewählt und im Ansichtsfenster angezeigt.

Anpassen der Grid-Ansicht

- Um die Größe von Spalten anzupassen, platzieren Sie den Cursor über den entsprechenden Rahmen und ziehen Sie diesen an die gewünschte Stelle.
- Um die Größe einer Spalte an die Breite des breitesten Eintrags anzupassen, doppelklicken Sie auf den rechten Rand der gewünschten Spalte.
- Um die Spaltenbreite so anzupassen, dass der gesamte Inhalt zu sehen ist, wählen Sie den Menüeintrag Ansicht | Optimale Breite[1505] oder klicken Sie auf das Symbol "Optimale Breite".
- Die Höhe der Zellen ist vom jeweiligen Inhalt abhängig. Sie kann über die Menüoption **Extras | Optionen | Ansicht | Grid-Ansicht** "Max. Zellenhöhe xxx Zeilen" angepasst werden.

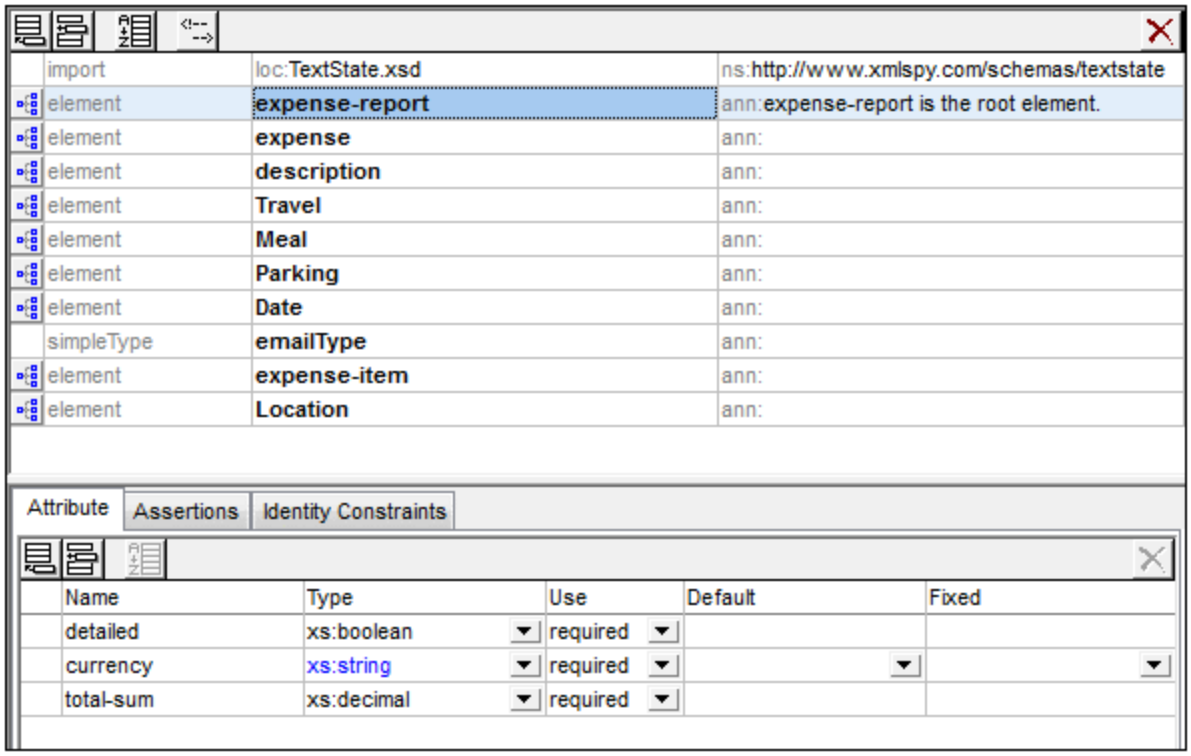
Anmerkung: Wenn Sie Daten in der Grid-Ansicht markieren und in die Textansicht wechseln, werden diese auch in der Textansicht markiert angezeigt.

© 2016-2022 Altova GmbH

4.4 Schema-Ansicht

Altova Website: XML-Schema-Editor

XML-Schemas können in der Schema-Ansicht (*Abbildung unten*) grafisch angezeigt und bearbeitet werden. Über die grafische Oberfläche können Sie Schemas schnell und exakt Hilfe typischer GUI-Funktionen erstellen. Die Schema-Ansicht verfügt über zwei Bereiche: (i) einen oberen, in dem die Strukturbeziehungen zwischen Schemakomponenten erstellt werden und (ii) einem unteren, in dem Definitionen zu der im oberen Bereich ausgewählten Komponente bearbeitet werden können. Außerdem stehen drei Eingabehilfen zur Verfügung, die die Erstellung von Schemas erleichtern: "Komponenten", "Details" und "Facets".



Der obere Bereich: Schema-Design

Im oberen Bereich der Schema-Ansicht können Sie zwischen zwei Ansichten wechseln:

- In der Schema-Übersicht[177] werden alle globalen Komponenten des Schemas (wie z.B. globale Elemente und complexTypes) in Form einer einfachen tabellarischen Liste (*siehe Abbildung*[177]) angezeigt. Durch Klicken auf das Symbol für eine globale Komponente können Sie in die Content Model-Ansicht wechsen. Nicht alle globalen Komponenten können ein Content Model (z.B. simpleTypes) haben.
- In der Content Model-Ansicht[190] wird das Content Model der ausgewählten globalen Komponente angezeigt (*siehe Abbildung*[190]). Um zur Schema-Übersicht zurückzukehren, klicken Sie links oben im oberen Fenster auf das Symbol "Globale Elemente anzeigen".



Zur Content Model-Ansicht wechseln: Steht für globale Komponenten zur Verfügung, die ein Content Model haben. Öffnet das Content Model der globalen Komponente in der Content Model-Ansicht[190].



Globale Elemente anzeigen: Steht in der Content Model-Ansicht zur Verfügung. Dient zum Wechseln in die Schema-Übersicht[177].

Der untere Bereich: Attribute, Assertions und Identity Constraints

Der untere Bereich der Schema-Ansicht (*siehe Abbildung*[212]) enthält Register, über die Sie für die im oberen Bereich ausgewählte Komponente Attribute, Assertions und Identity Constraints[212] definieren können. Abgekürzt bezeichnen wir diesen Bereich als den AAIDC-Bereich.

- In XSD 1.0 hat der untere Bereich zwei Register: (i) Attribute[213] und (ii) Identity Constraints[220].
- Im XSD 1.1-Modus hat der untere Bereich drei Register: (i) Attribute[213], (ii) Assertions[217] und (iii) Identity Constraints[220].

Der AAIDC-Bereich wird in der Schema-Übersicht immer angezeigt und kann auch in der Content Model-Ansicht zu sehen sein. In der Content Model-Ansicht können alle Arten von Definitionen (Attribute, Assertions und IDCs) im Diagramm anstatt im AAIDC-Bereich angezeigt werden. Dazu müssen Sie die entsprechenden Schaltflächen in der Schema-Design-Symbolleiste aktivieren: (i) **Attribute direkt in Diagramm anzeigen**, (ii) **Assertions im Diagramm anzeigen** und (iii) **Identity Constraints in Diagramm anzeigen**. Alternativ dazu können Sie diese Einstellungen im Dialogfeld "Schema-Ansicht konfigurieren" (**Schema-Design | Ansicht konfigurieren**[1395]) anzeigen. Wenn alle Definitionstypen des AAIDC-Bereichs im Diagramm angezeigt werden, wird der untere Bereich in der Content Model-Ansicht nicht mehr eingeblendet.

Schemaeinstellungen

Das Dialogfeld "Schema-Einstellungen" (**Schema-Design | Schema-Einstellungen**[1386]) wird von der Schema-Ansicht aus aufgerufen. Hier können Sie die globalen Einstellungen für das aktive Schema vornehmen. Diese Einstellungen sind die Attribute des `xs:schema` Elements.

Gliederung dieses Abschnitts

Dieser Abschnitt wurde in die folgenden Unterabschnitte gegliedert

- XSD-Modus: XSD 1.0 oder 1.1[173]: Wählen Sie zwischen den beiden Bearbeitungsmodi.
- Schema-Übersicht[177]: Bearbeiten der Eigenschaften von globalen Komponenten
- Content Model-Ansicht[190]: Bearbeiten der Content Models einzelner globaler Komponenten
- Attribute, Assertions und Identity Constraints[212]: Definieren dieser Komponenteneigenschaften
- Eingabehilfen[229]: Verwendung der Eingabehilfen zum schnellen Definieren verschiedener Komponenteneigenschaften
- Smart Restrictions[244]: Grafisches Erstellen und Bearbeiten von Typen, die von einem Basistyp abgeleitet werden
- Verwendung von mit dem Präfix xml: versehenen Attributen[249]: Grafisches Hinzufügen der Attribute `base`, `id`, `lang` und `space` zu Schemakomponenten.
- Zurück und Vorwärts: Navigieren von einer Position zu einer anderen[251]: Hier wird eine Funktion der Schema-Ansicht erläutert, mit der Sie zwischen zuvor angezeigten Positionen navigieren können.

Herstellen einer Verbindung zu SchemaAgent

Sie können von XMLSpy aus eine Verbindung zu SchemaAgent herstellen, um Komponenten aus anderen Schemas auf der GUI darzustellen und diese Komponenten im Schema, das Sie gerade bearbeiten zu

© 2016-2022 Altova GmbH

verwenden. Eine Anleitung zum Arbeiten mit in XMLSpy finden Sie im Abschnitt Arbeiten mit SchemaAgent[470] .

Suchen in Schemas

Mit der Funktion "In Schemas suchen" können Sie eine intelligente Suche, d.h. eine Suche, die nach verschiedenen schemabezogenen Kriterien eingeschränkt ist, in Schemas durchführen. So kann eine Suche z.B. auf bestimmte Komponententypen eingeschränkt werden, um sie effizienter zu machen. Nähere Informationen zum Suchen in Schemas finden Sie im Abschnitt DTDs und XML-Schemas[482].

4.4.1 XSD-Modus: XSD 1.0 oder 1.1

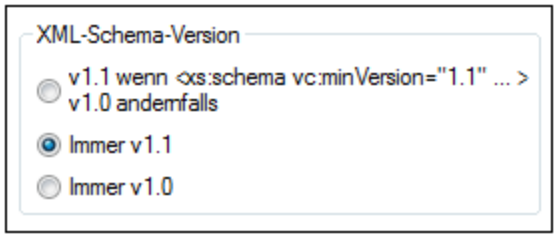
Sie können in der Schema-Ansicht auswählen, ob das XML-Schema (XSD) anhand der XML Schema 1.0 (XSD 1.0)- oder der XML Schema 1.1 (XSD 1.1)-Spezifikation bearbeitet und validiert werden soll. Der zum Bearbeiten einer Datei verwendete **XSD-Modus** basiert auf zwei Einstellungen: einer in der Applikation und der anderen im XSD-Dokument.

Auswahl des XSD-Modus

Es ist vom XSD-Modus abhängig, welche Bearbeitungs- und Validierungsfunktionen (XSD 1.0 oder 1.1) für das aktive Dokument in der Schema-Ansicht zur Verfügung stehen. Sie können entweder (i) eine Einstellung für die gesamte Applikation vornehmen. In diesem Fall werden alle XSD-Dokumente in der Schema-Ansicht im ausgewählten Modus bearbeitet. Oder (ii) Sie können die XSD-Versionsnummer im XSD-Dokument speichern, sodass die Applikation den XSD-Modus automatisch gemäß dieser Informationen wählt.

Applikationsweiter Modus

Die applikationsweite Einstellung wird im Dialogfeld "Optionen" (**Extras | Optionen**, *siehe Abbildung unten*) im Abschnitt "Datei" vorgenommen. Wenn Sie das Optionsfeld *Version 1.0* oder *Version 1.1* auswählen, so wird der ausgewählte Modus zum applikationsweiten Modus. Alle in der Schema-Ansicht geöffneten XML-Schema-Dokumente werden nun in diesem Modus bearbeitet. (Wenn Sie die Einstellung *v1.1 wenn <xs:schema vc:minVersion="1.1", v1.0 andernfalls* auswählen, hängt der Modus von den Informationen im Dokument ab und wird nicht applikationsweit verwendet. Nähere Informationen dazu finden Sie unter *dokumentspezifischer Modus* und den anderen Abschnitten weiter unten.)



Durch Auswahl der gewünschten Version im Dialogfeld "Optionen", Einstellung XML-Schema-Version (*Abbildung oben*) können Sie jederzeit zwischen den beiden applikationsweiten Modi (*Version 1.0* und *Version 1.1*) wechseln.

Anmerkung: Wenn es sich bei der aktuellen Einstellung um eine applikationsweite Einstellung handelt und Sie über die Schaltfläche **XSD 1.0** oder **XSD 1.1** in der Symbolleiste "Schema-Design" den Modus

wechseln, so ist der Moduswechsel temporär und der Modus wird beim Neuladen des Dokuments in den applikationsweiten Modus zurückgesetzt. Das Dokument wird bei jedem Ansichtswechsel und bei jedem Aktualisieren der Schema-Ansicht (mit dem Befehl **Datei | Neu laden**) neu geladen.

Dokumentspezifischer Modus

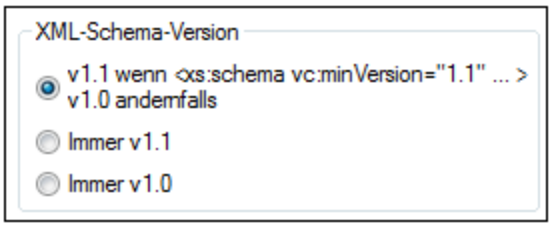
Sie können die XSD-Modusinformationen auch im XSD-Dokument selbst speichern. Dadurch wechselt die Schema-Ansicht automatisch in den XSD-Modus des Dokuments, wenn das Dokument geladen wird. Sie können die XSD-Modusinformationen durch Klicken auf die Schaltfläche **XSD 1.0** bzw. **XSD 1.1** in der Symbolleiste "Schema-Design" (*Abbildung unten*) zu einem XSD--Dokument hinzufügen. Dadurch wird der ausgewählte Modus im `vc:minVersion` Attribut des obersten `xs:schema` Elements gespeichert. (Der Wert von `vc:minVersion` kann in der Textansicht manuell hinzugefügt werden.)



Anmerkung: Das Attribut `vc:minVersion` muss sich, falls vorhanden, im Namespace `http://www.w3.org/2007/XMLSchema-versioning` befinden. In diesem Fall muss das XML-Schema-Dokument eine Namespace-Deklaration haben, die `vc:` Namespace-Präfix an diesen Namespace bindet. Wenn Sie die Symbolleisten-Schaltfläche **XSD 1.1** (*Abbildung oben*) verwenden, wird der Namespace automatisch hinzugefügt. Wenn Sie auf die Symbolleisten-Schaltfläche **XSD 1.0** klicken, wird diese Namespace-Deklaration entfernt, wenn sich kein anderer Node-Name im Dokument in diesem Namespace befindet.

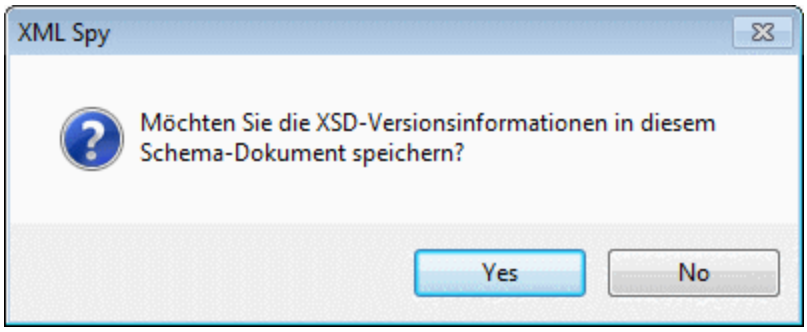
Um den dokumentspezifischen Modus zu aktivieren und den XSD-Modus eines Dokuments zu definieren, gehen Sie folgendermaßen vor:

1. *Aktivieren Sie den dokumentspezifischen Modus:* Setzen Sie die XML-Schema-Versionsoption im Dialogfeld "Optionen" (**Extras | Optionen**) im Abschnitt "Datei" auf *v1.1 wenn <xs:schema vc:minVersion="1.1", v1.0 andernfalls* (*siehe Abbildung unten*). Damit legen Sie fest, dass der XSD-Modus in der Schema-Ansicht von XMLSpy entsprechend dem `vc:minVersion` Attribut des `xs:schema` Elements gewählt wird.



2. *Definieren Sie die XSD-Version des Dokuments:* Klicken Sie in der Symbolleiste "Schema-Design" (*Abbildung oben*) auf die Schaltfläche **XSD 1.0** bzw. **XSD 1.1**. Daraufhin wird ein Bestätigungsdialogfeld angezeigt (*Abbildung unten*).

© 2016-2022 Altova GmbH



3. Wenn Sie auf **Ja** klicken, (i) wird der entsprechende Wert in das `vc:minVersion` Attribut des `xs:schema` Elements eingegeben. (ii) Wenn **XSD 1.1** ausgewählt wurde, wird der `XMLSchema-versioning` Namespace mit einer Bindung an das `vc:` Namespace-Präfix deklariert; wenn **XSD 1.0** ausgewählt wurde, wird die Namespace-Deklaration entfernt, wenn sich kein anderer Node im `XMLSchema-versioning` Namespace befindet. Das XML-Schema-Dokument enthält nun die XSD-Versionsnummer. Beim Speichern der Datei werden die XSD-Modusinformationen mit der Datei gespeichert. Wenn Sie die Datei erneut öffnen oder neu laden, wechselt die Schema-Ansicht automatisch in den XSD-Modus, der im Attribut `vc:minVersion` des `xs:schema` Elements des Dokuments gespeichert ist.

Anmerkung: Wenn der dokumentspezifische Modus ausgewählt ist und das XSD-Dokument kein `vc:minVersion`Attribut hat oder der Wert des `vc:minVersion` Attributs nicht `1.0` oder `1.1` ist, wird in der Schema-Ansicht automatisch der Modus XSD 1.0 verwendet.

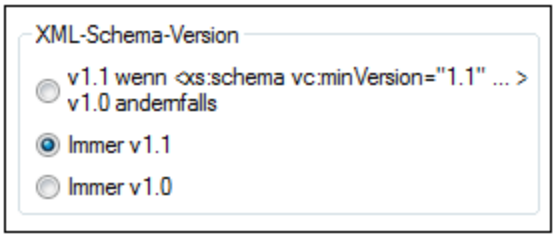
Anmerkung: Verwechseln Sie das `vc:minVersion` Attribut nicht mit dem `xsd:version` Attribut. Ersteres enthält die XSD-Versionsnummer, zweiteres die Versionsnummer des Dokuments.

XSD-Modus neuer Dokumente

Wenn Sie ein neues XSD-Dokument erstellen, werden Sie gefragt, ob Sie es als XSD 1.0- oder XSD 1.1-Dokument erstellen möchten. Bei Auswahl von XSD 1.1 wird ein neues Dokument mit dem Attribut `/xs:schema/@vc:minVersion="1.1"` erstellt und der `XMLSchema-versioning` Namespace wird mit einer Bindung an das `vc:` Namespace-Präfix deklariert. Bei Auswahl von XSD 1.0 werden weder das `vc:minVersion` Attribut noch die `XMLSchema-versioning` Namespace-Deklaration hinzugefügt. Welcher XSD-Modus in der Schema-Ansicht jedoch tatsächlich verwendet wird, hängt auch davon ab, welche XML-Schema-Version im Dialogfeld "Optionen" (**Extras | Optionen**) im Abschnitt "Datei" ausgewählt ist. Eine Beschreibung darüber, wie sich diese beiden Einstellungen beeinflussen, finden Sie im nächsten Abschnitt.

Der aktivierte XSD-Modus

Welcher XSD-Modus in der Schema-Ansicht aktiviert wird, hängt sowohl (i) vom Vorhandensein bzw. Fehlen - und, falls vorhanden, vom Wert - des `/xs:schema/@vc:minVersion` Attributs des XSD-Dokuments als auch (ii) von der XML-Schema-Version ab, die im Dialogfeld "Optionen" (**Extras | Optionen**, *siehe Abbildung unten*) im Abschnitt "Datei" ausgewählt wurde.



Es sind die folgenden Situationen möglich. *XML-Schema-Version* in der Tabelle unten bezieht sich auf die Auswahl im oben gezeigten Bereich XML-Schema-Version. Die `vc:minVersion` Werte in der unten stehenden Tabelle beziehen sich auf den Wert des `xs:schema/@vc:minVersion` Attributs im XML-Schema-Dokument.

XML-Schema-Version	`vc:minVersion` Attribut	XSD-Modus
Immer v 1.0	Fehlt oder ist mit einem beliebigen Wert vorhanden	1.0
Immer v 1.1	Fehlt oder ist mit einem beliebigen Wert vorhanden	1.1
Wert von @vc:minVersion	Das Attribut hat den Wert `1.1`	1.1
Wert von @vc:minVersion	Das Attribut fehlt oder ist vorhanden und hat einen anderen Wert als `1.1`	1.0

Anmerkung: In den in den ersten beiden Zeilen beschriebenen Situationen ist es möglich, dass ein XSD 1.1-Schema im XSD 1.0-Modus bzw. umgekehrt geöffnet wird. Die sich daraus ergebenden Unstimmigkeiten werden, wie weiter unten beschrieben, behandelt.

XSD-Modus-Funktionen

Die Benutzeroberfläche und die Bearbeitungsfunktionen der Schema-Ansicht ändern sich, je nachdem, welcher XSD-Modus (XSD 1.0 oder XSD 1.1) aktiviert ist.

Wenn der **XSD 1.0-Modus** ausgewählt ist:

- stehen die Bearbeitungsfunktionen für neue XML Schema 1.1-Komponenten und Eigenschaften nicht zur Verfügung. Wenn jedoch bereits XSD 1.1-Komponenten oder Eigenschaften im XSD-Dokument vorhanden sind, werden diese angezeigt und können gelöscht werden.
- Die Validierung wird anhand der XSD 1.0-Spezifikation durchgeführt. Wenn das Schema daher eine ausschließlich in XSD 1.1 vorhandene Komponente oder Eigenschaft enthält, wird ein Validierungsfehler ausgegeben.

Wenn der **XSD 1.1-Modus** aktiviert ist, stehen die Bearbeitungsfunktionen für alle XML-Schema 1.1-Eigenschaften zur Verfügung. Die Validierung erfolgt gemäß der XML-Schema 1.1-Spezifikation.

Behandlung von XSD 1.1-Funktionen im XSD 1.0-Modus

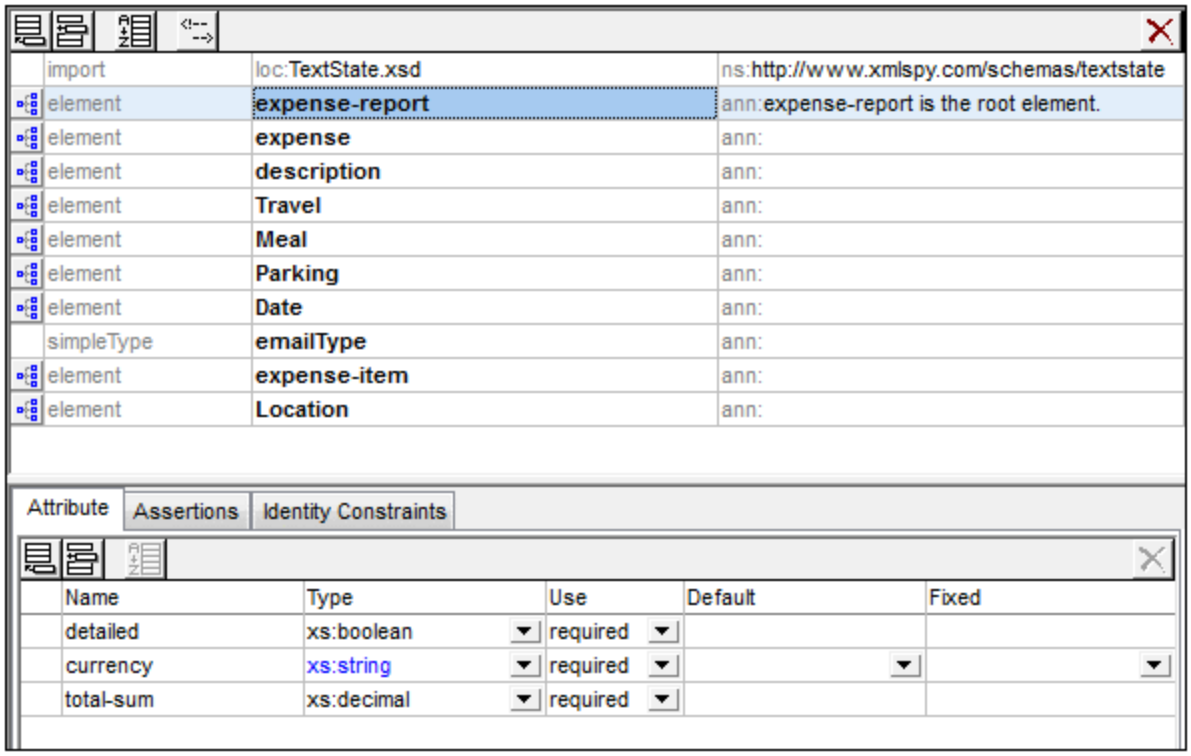
Im Folgenden wird beschrieben, wie eine XSD 1.1-Funktion, die in XSD 1.0 nicht vorhanden ist, im XSD 1.0-Modus angezeigt und behandelt wird.

© 2016-2022 Altova GmbH

- ***Assertions:*** Wenn es im ausgewählten simpleType mindestens eine Assertion gibt, enthält die Eingabehilfe "Facets" das Register "Assertions". Die Assertion kann jedoch nicht bearbeitet, sondern nur gelöscht werden.
- ***Asserts:*** Die Assertion wird, falls vorhanden, im Diagramm des complexType angezeigt. Im AAIDC-Bereich wird kein Register "Assertions" angezeigt. Es kann keine Assertion über das Kontextmenü hinzugefügt werden. Die Assertion kann nicht bearbeitet, sondern nur gelöscht werden.
- ***Attribute:*** Die neue Eigenschaft `inheritable` wird, falls vorhanden, angezeigt, kann aber nicht bearbeitet werden. Es kann nur der Leerwert ausgewählt werden (was bedeutet, dass die Eigenschaften entfernt werden).
- ***ComplexTypes:*** Die neue Eigenschaft `defaultAttributesApply` wird, falls vorhanden, in der Eingabehilfe "Details" angezeigt. Sie kann nicht bearbeitet werden. Es kann nur der Leerwert ausgewählt werden (was bedeutet, dass die Eigenschaften entfernt werden).
- ***Dokumentation:*** Neue XSD 1.1-spezifische Komponenten und Eigenschaften werden nicht in die Schema-Ansichtsdokumentation inkludiert.
- ***Facets:*** Unbekannte Facets verursachen Validierungsfehler und werden rot angezeigt.
- ***In Schemas suchen:*** Neue XSD 1.1-spezifische Komponenten und Eigenschaften werden ignoriert.
- ***Identity constraints (IDCs):*** Die Eigenschaft `isRef` wird bei Referenzen angezeigt und kann deaktiviert werden. Sie wird deaktiviert, sobald der Name des IDC geändert wird.
- ***Multiple substitution groups:*** Auswahlliste, zur Auswah einer einzigen Substitution Group (in XSD 1.0 sind nur einzelne Substitution Groups zulässig).
- ***Open content:*** Wird im Diagramm angezeigt, falls vorhanden. Kann nicht über das Kontextmenü hinzugefügt werden. Kann nicht bearbeitet, sondern nur gelöscht werden. Default Open Content wird in complexTypes nicht angezeigt.
- ***Override:*** Wird, falls vorhanden, in der Grid-Ansicht von globalen Elementen angezeigt. Kann nicht über das Menü hinzugefügt werden. Die Position kann nicht bearbeitet, sondern nur gelöscht werden. Überschreibende Komponenten (d.h. Subelemente von `xs:override`) werden ignoriert und nicht in der Eingabehilfe "Komponenten" angezeigt.
- ***Schema-Einstellungen:*** Die neuen Eigenschaften `defaultAttributes` und `xpathDefaultNamespace` werden, falls vorhanden, im Dialogfeld angezeigt. Sie können nicht bearbeitet werden. Es kann nur der Leerwert ausgewählt werden (was bedeutet, dass die Eigenschaften entfernt werden).
- ***SimpleTypes:*** Unbekannte Typen verursachen Validierungsfehler und werden rot angezeigt.
- ***Type Alternatives:*** Werden, falls vorhanden, im Diagramm angezeigt. Können nicht über das Kontextmenü hinzugefügt werden. Die Eigenschaften können nicht bearbeitet, sondern nur gelöscht werden.
- ***Wildcards:*** Neue Eigenschaften werden, falls vorhanden, angezeigt. Sie können nicht bearbeitet werden. Es kann nur der Leerwert ausgewählt werden (was bedeutet, dass die Eigenschaften entfernt werden).

4.4.2 Schema-Übersicht

In der Schema-Übersicht sehen Sie eine Liste aller globalen Komponenten des Schemas (`import` Elemente, globale Elemente, complexTypes usw.).



Sie können globale Komponenten einfügen, anhängen oder löschen und ihre Eigenschaften bearbeiten. Um Eigenschaften zu bearbeiten, wählen Sie die globale Komponente in der Schema-Übersichtsliste aus. Je nachdem, welche Art von globaler Komponente ausgewählt ist, können die Eigenschaften in der Eingabehilfe "Details" [233], "Facets" [235] und/oder im Attribute-/Identity Constraints-Bereich (AAIDC-Bereich) [212] bearbeitet werden.

Eine globale Komponente, die ein Content Model haben kann, hat in der Liste der globalen Komponenten links von der Komponente das Symbol **Zur Content Model-Ansicht wechseln**. Wenn Sie auf dieses Symbol klicken, wechseln Sie in die Content Model-Ansicht [190] dieser Komponente, wo das Inhaltsmodell (Content Model) dieser Komponente bearbeitet werden kann.

Zur Content Model-Ansicht wechseln: Steht für globale Komponente zur Verfügung, die ein Content Model haben. Öffnet das Content Model der globalen Komponente in der Content Model-Ansicht [190].

Globale Elemente anzeigen: Steht in der Content Model Ansicht zur Verfügung. Wechselt in die Schema-Übersicht [177].

In diesem Abschnittt werden zuerst die GUI-Mechanismen [178] der Schema-Übersicht beschrieben, bevor die verschiedenen globalen Komponenten [182] im Detail beschrieben werden.

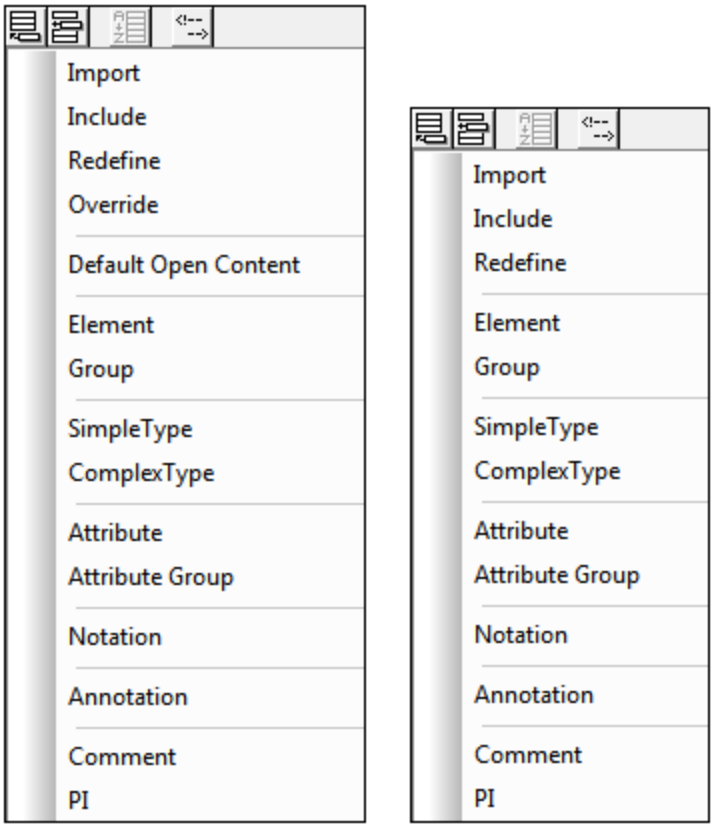
4.4.2.1 GUI-Mechanismen

Globale Komponenten werden als Children des obersten Elements `xs:schema` hinzugefügt. Um eine globale Komponente hinzuzufügen, klicken Sie auf das **Anhängen** oder **Einfügen**-Symbol in der linken oberen Ecke

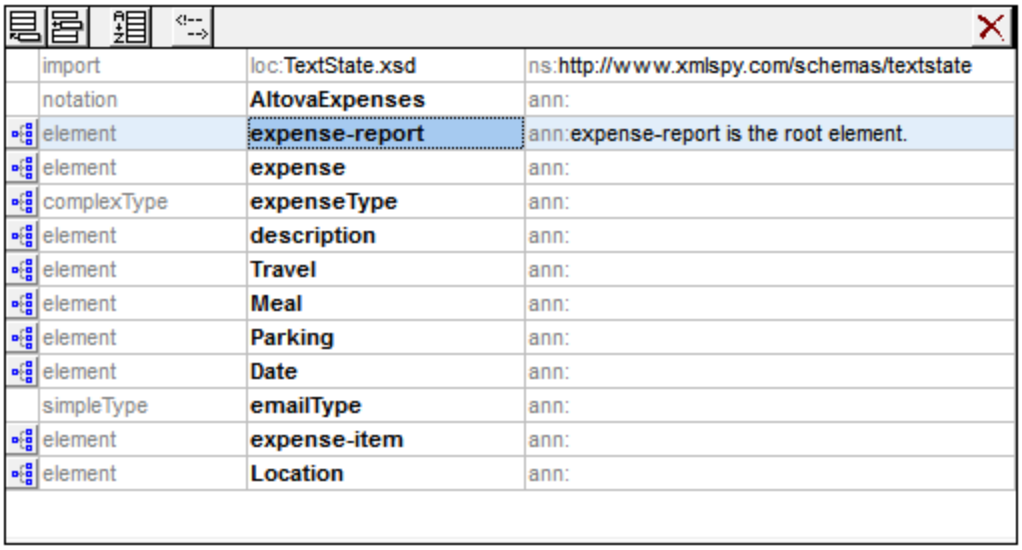
© 2016-2022 Altova GmbH

des oberen Fensters (*Liste der Symbole siehe unten*) und wählen Sie im Menü der globalen Komponenten (*Abbildungen unten*) die gewünschte globale Komponente aus.

In der Abbildung unten sehen Sie die globalen Komponenten, die hinzugefügt werden können. XSD 1.1-Modus auf der linken Seite und XSD 1.0-Modus auf der rechten Seite (*Override* und *Default Open Content* sind XSD 1.1-Features.)



Sie können beliebig viele globale Komponenten hinzufügen. Alle globalen Komponenten im Schema werden in der Schema-Übersicht (*Abbildung unten*) in Form einer tabellarischen Liste angezeigt.



	import	loc:TextState.xsd	ns:http://www.xmlspy.com/schemas/textstate
	notation	**AltovaExpenses**	ann:
	element	**expense-report**	ann:expense-report is the root element.
	element	**expense**	ann:
	complexType	**expenseType**	ann:
	element	**description**	ann:
	element	**Travel**	ann:
	element	**Meal**	ann:
	element	**Parking**	ann:
	element	**Date**	ann:
	simpleType	**emailType**	ann:
	element	**expense-item**	ann:
	element	**Location**	ann:

Bearbeitung in der Schema-Übersicht

Beachten Sie die folgenden Bearbeitungsfunktionen in der Schema-Übersicht:

- Sie können Komponenten in Liste in der Schema-Übersicht mittels Drag-and-Drop an eine andere Stelle verschieben.
- Sie können mit Hilfe der Pfeiltasten und der Tabulatortaste Ihrer Tastatur navigieren.
- Sie können globale Komponenten, Attribute, Assertions und Identity Constraints durch Ausschneiden/Kopieren-und-Einfügen von einem Diagramm an eine andere Stelle im Diagramm, in andere Diagramme und von einem Schema in ein anderes kopieren oder verschieben.
- Bei Rechtsklick auf eine Komponente wird ein Kontextmenü geöffnet, über das Sie die Annotationsdaten dieser Komponente ausschneiden, kopieren, einfügen, löschen oder bearbeiten können.
- Um eine neue Zeile in globalen Kommentaren und globalen Annotationen einzugeben, drücken Sie **Strg+Eingabetaste**. Um einen Tabulator einzugeben, drücken Sie **Strg+Tab**.

Schema-Übersicht und darin vorkommende Symbole

	*Globale Komponente anhängen:* Fügt am Ende der Liste der globalen Komponenten eine globale Komponente hinzu. Wenn die Komponente gemäß der Definition am Anfang des Dokuments stehen muss, wird sie am Anfang der Liste hinzugefügt.
	*Globale Komponente einfügen:* Fügt eine globale Komponente oberhalb von der ausgewählten Komponente hinzu. Wenn die Komponente gemäß der Definition am Anfang des Dokuments stehen muss, wird sie am Anfang der Liste hinzugefügt.
	*Sortieren:* Ruft das Dialogfeld "Komponenten sortieren" auf, in dem Sie vor der Sortierung (*Beschreibung siehe unten*) die Vorrangigkeit von Sortierkriterien (Name vor Art oder umgekehrt) definieren können.

© 2016-2022 Altova GmbH

	*Comments:* Ruft ein Menü auf, in dem Sie zwischen der einzeiligen und mehrzeiligen Anzeige von globalen Kommentaren wählen können. *Beschreibung siehe unten.*
	*Zur Content Model-Ansicht wechseln:* Steht für globale Komponenten zur Verfügung, die ein Content Model haben. Öffnet das Content Model der globalen Komponente in der Content Model-Ansicht 190.
	*Globale Elemente anzeigen:* Steht in der Content Model Ansicht zur Verfügung. Wechselt in die Schema-Übersicht 177.

Wechseln zwischen der Schema-Übersicht und der Content Model-Ansicht

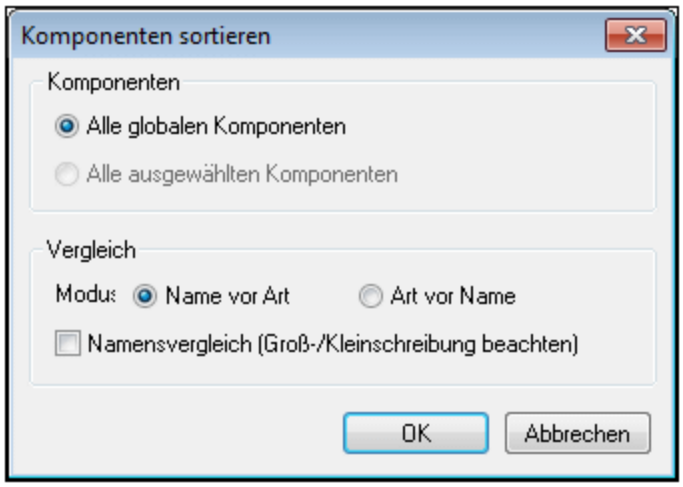
Einige globale Komponenten (wie z.B. complexTypes, Elementdeklarationen und Model Groups) haben links neben dem Komponentennamen (*siehe Liste der Symbole oben*) ein Symbol **Zur Content Model-Ansicht wechseln.** Damit wird angezeigt, dass diese globalen Komponenten ein Content Model haben können, in dem die Struktur und Inhalt der Komponente beschrieben werden.

Wenn Sie auf dieses Symbol klicken, wechseln Sie von der Schema-Übersicht in die Content Model-Ansicht 190 dieser globalen Komponente. Andere globale Komponenten (wie Annotationen, simpleTypes und Attributgruppen) haben kein Content Model und daher auch kein **Zur Content Model-Ansicht wechseln** Symbol. Um von der Content Model-Ansicht wieder in die Schema-Übersicht zu wechseln, klicken Sie auf das Symbol **Globale Elemente anzeigen** (*siehe Symbolliste oben*).

Sortieren globaler Komponenten

Durch Klicken auf die Schaltfläche **Sortieren** (*siehe Symbolliste oben*) in der Symbolleiste "Schema-Übersicht" können Sie globale Komponenten sortieren. Im Dialogfeld "Komponenten sortieren", das daraufhin angezeigt wird (*Abbildung unten*) können Sie auswählen, ob entweder alle sortierbaren globalen Komponenten oder die Gruppe der ausgewählten Komponenten sortiert werden soll. Mit **Klick+Umschalttaste** können Sie einen Bereich auswählen, mit **Klick+Strg** können Sie zusätzliche Komponenten zur Auswahl hinzufügen.

Anmerkung: Globale Komponenten, die am Anfang des Dokuments stehen müssen (wie z.B. `include` und `import`), sind nicht von der Sortierung betroffen. Sie bilden nicht Teil des sortierbaren Bereichs der globalen Komponenten.

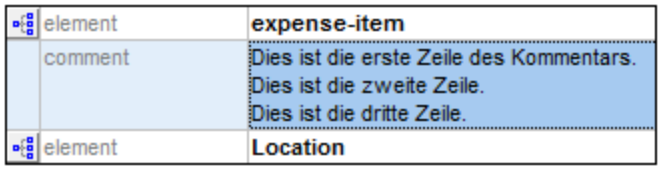


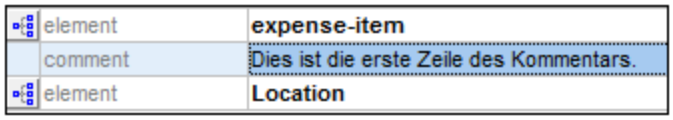
Nachdem Sie den Sortierbereich festgelegt haben, können Sie auswählen, ob die Komponenten alphabetisch (*Name vor Art*) oder zuerst nach ihrem Typ und dann nach dem Namen sortiert werden sollen.

Die Sortierreihenfolge wird im Text des Schemas implementiert.

Globale Kommentare: Zeilenanzeigemodus

Globale Komponenten können in einem mehrzeiligen (Standardeinstellung) oder in einem einzeiligen Textfeld angezeigt werden (*siehe Abbildungen unten*).





Um zwischen diesen beiden Anzeigemodi für Kommentare zu wechseln, klicken Sie auf das **Comments** Symbol am oberen Rand der Schema-Übersicht und wählen Sie die gewünschte Option aus. Wenn Sie im Text eines Kommentars eine neue Zeile erstellen möchten (und den Kommentar somit mehrzeilig machen wollen), drücken Sie **Strg+Eingabe**. Wenn die Kommentare einzeilig angezeigt werden, erscheint bei Platzierung des Cursors über einen mehrzeiligen Kommentar ein mehrzeiliges Textfeld, in dem alle Zeilen angezeigt werden.

4.4.2.2 Globale Komponenten

Globale Komponenten sind Komponenten, die als Children des obersten Elements `xs:schema` hinzugefügt werden, (während lokale Komponenten innerhalb von anderen Komponenten erstellt werden). Einige globale Komponenten, wie z.B. complexTypes, Elemente und Attribute können von anderen Komponenten im Schema referenziert werden.

Erstellen von globalen Komponenten in der Schema-Übersicht

Globale Komponenten werden normalerweise in der Schema-Übersicht[177] erstellt und bearbeitet. Sie werden in der Schema-Übersicht mit Hilfe der Symbole **Anhängen**[178] oder[178] **Einfügen**[178] hinzugefügt. Das Content Model einer globalen Komponente wird in der Content Model-Ansicht[190] dieser Komponente erstellt und bearbeitet (falls die globale Komponente ein Content Model haben kann, *siehe Tabelle unten)*. (Klicken Sie links vom Komponentennamen auf das Symbol **Zur Content Model-Ansicht wechseln,** um in die Content Model-Ansicht[190] zu wechseln.)

Einige globale Komponenten werden bei ihrer Erstellung in der Schema-Übersicht auch zur Eingabehilfe "Komponenten"[229] hinzugefügt. Wenn eine Komponente ein Content Model hat, wird dieses durch Doppelklick auf den Komponentennamen in der Eingabehilfe "Komponenten" zur Bearbeitung in der Content Model-Ansicht[190] geöffnet.

© 2016-2022 Altova GmbH

Wenn die globale Komponente eine Typdefinition hat (simpleType oder complexType), gelangen Sie bei Auswahl des Kontextmenübefehls **Gehe zu Typdefinition** der Komponente zur Typdefinition. In Fall von vordefinierten simpleTypes wird ein Meldungsfeld mit Informateionen über den simpleType angezeigt.

Anmerkung: Sie können einige globale Komponenten (Elemente, Attribute, simpleTypes, complexTypes und Model Groups) auch während der Bearbeitung in der Content Model-Ansicht erstellen. Klicken Sie dazu an einer beliebigen Stelle im Fenster auf die rechte Maustaste und wählen Sie den Befehl **Neues Global | < Art der globalen Komponente >**.

Anmerkung: Sie können ein lokales Element in der Content Model-Ansicht zu einem globalen Element - oder einem globalen complexType machen, wenn das Element als Child ein Element oder Attribut hat. Wählen Sie das lokale Element aus, rechtsklicken Sie in das Fenster und wählen Sie den Befehl **Als global definieren | Element** oder **Als global definieren | Complex type**.

Globale Komponente	Position im Schema	Content Model
`include`	Anfang	Nein
`import`	Anfang	Nein
`redefine`	Anfang	Nein
`override` [1.1]	Anfang	Nein
`defaultOpenContent` [1.1]	Nach Includes, Imports, Redefines Overrides; vor jedem anderen Element	Ja
`element`	An beliebiger Stelle nach `defOpenCont`	Ja
`group`	An beliebiger Stelle nach `defOpenCont`	Ja
`simpleType`	An beliebiger Stelle nach `defOpenCont`	Nein
`complexType`	An beliebiger Stelle nach `defOpenCont`	Ja
`attribute`	An beliebiger Stelle nach `defOpenCont`	Nein
`attributeGroup`	An beliebiger Stelle nach `defOpenCont`	Nein
`notation`	An beliebiger Stelle nach `defOpenCont`	Nein
`annotation`	An beliebiger Stelle nach `defOpenCont`	Nein
Comment	An beliebiger Stelle	Nein
Processing Instruction	An beliebiger Stelle	Nein

Im Folgenden finden Sie die wichtigsten Punkte zur Bearbeitung dieser Komponenten in der Schema-Ansicht.

Includes, Imports, Redefines und Overrides

Mit Hilfe dieser vier globalen Komponenten können Sie andere Schemadokumente innerhalb des aktuellen Schemadokuments wiederverwenden.

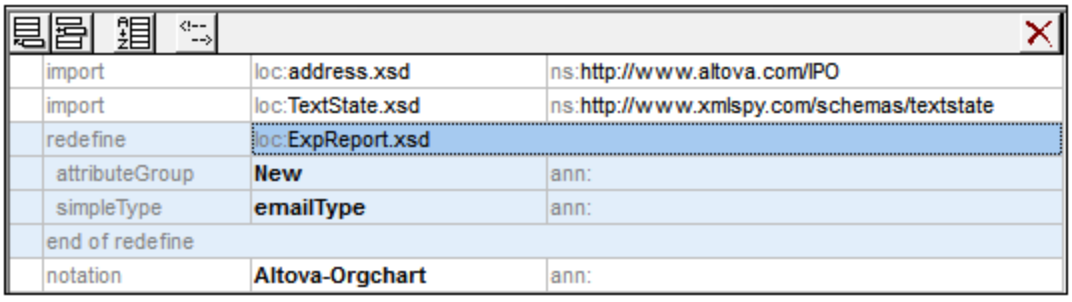
- Mittels "Include" werden Dokumente wiederverwendet, die denselben Ziel-Namespace wie das aktuelle Dokument haben.
- Mittels "Import" werden Dokumente wiederverwendet, die einen anderen Ziel-Namespace als das aktuelle Dokument haben.
- "Redefine"/"Override" ist insofern ein Include-Typ, als er denselben Ziel-Namespace wie das aktuelle Dokument hat, jedoch werden damit Teile der inkludierten Schemas geändert. "Redefine" ist ein XSD 1.0-Feature und wird in XSD 1.1 nicht mehr verwendet. "Override" ist ein 1.1-Feature, das flexibler ist und in XSD 1.1 anstelle von "Redefine" tritt.

Alle vier Komponenten haben ein `schemaLocation` Attribut, das auf das Schema verweist, das wiederverwendet werden soll. Wenn Sie in der Schema-Ansicht auf das `loc` Feld dieser Komponenten doppelklicken, können Sie zur gewünschten Datei, die wiederverwendet werden soll, navigieren, und ihren Pfad relativ zum aktuellen Dokument definieren. Die `import` Komponente hat zusätzlich dazu ein `namespace` Attribut, das den Ziel-Namespace des importierten Schemas enthält.

Wenn ein Schema im aktuellen Schemadokument (mittels include, import, redefine oder override) wiederverwendet wird, werden seine globalen Komponenten, Namespaces und Identity Constraints in der Eingabehilfe "Komponenten" 229 des aktuellen Dokuments angezeigt.

Redefines

In einer `redefine` Komponente können Sie complexTypes, simpleTypes, Model Groups und Attribute Groups ändern. Die umzudefinierende Komponente befindet sich im dem im Feld `loc` definierten Schema der `redefine` Komponente (in der Abbildung unten befinden sich die umzudefinierenden Komponenten im Schema `ExpReport.xsd`). Nachdem eine `redefine` Komponente hinzugefügt wurde, müssen Sie die umzudefinierende Komponente an einer Stelle zwischen der Zeile `redefine` und `end of redefine` in der Liste der globalen Komponenten hinzufügen (*siehe Abbildung unten, in der die Komponenten* `New` *und* `emailType` *umdefiniert werden*). Diese beiden Komponenten stammen aus dem Schema `ExpReport.xsd` und werden für das aktuelle Schema umdefiniert.



Um eine Komponente umzudefinieren, gehen Sie folgendermaßen vor:

1. Wählen Sie die Zeile `end of redefine`.
2. Klicken Sie links oben in der Schema-Übersicht auf das **Einfügen**-Symbol.
3. Wählen Sie die Art der zu definierenden Komponente aus (complex type, simple type, model group oder attribute group). Die Komponente wird innerhalb der redefine-Komponente hinzugefügt.

© 2016-2022 Altova GmbH

4. Geben Sie ihr denselben Namen wie die umzudefinierende Komponente. Die Komponente erhält dadurch alle Eigenschaften der Komponente aus dem wiederverwendeten Schema.
5. Definieren Sie die Komponente um, indem Sie sie auswählen und ihre Eigenschaften in den Eingabehilfen "Details" und "Facets" ändern oder indem Sie ihr Content Model in der Content Model-Ansicht ändern (falls die Komponente ein Content Model hat).

Anmerkung: Sie können die umzudefinierenden Komponenten unter Umständen auch folgendermaßen einfügen: entweder von einer anderen Stelle in der Liste der globalen Komponenten oder mittels Drag-and-Drop oder Kopieren/Einfügen aus der Eingabehilfe "Komponenten".

Umdefinierte Komponenten können von anderen Komponenten im Schema aus referenziert werden.

Overrides

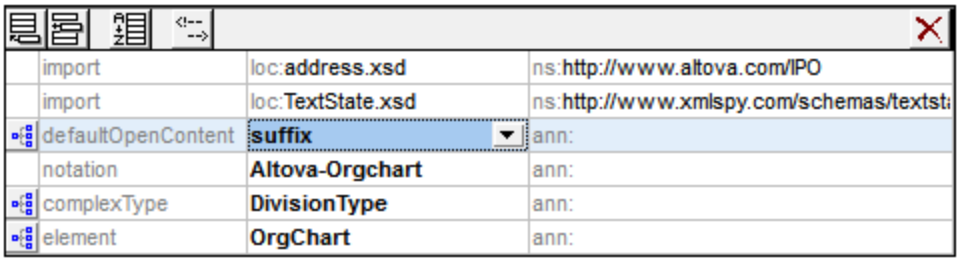
In einem `override` Element können Sie die folgenden Komponenten definieren: complexTypes, simpleTypes, globale Elemente, globale Attribute, Model Groups, Attribute Groups und notations. Wenn eine dieser Komponenten in einem `override` Element definiert wird, so ersetzt diese Komponente im überschriebenen Schema alle Komponenten derselben Art, die denselben Namen wie die überschreibende Komponente haben. Das überschriebene Schema wird im `loc` Feld des Override-Elements definiert.

Override-Elemente unterscheiden sich insofern von Redefine-Elemente (*siehe oben),* als es sich hierbei um Komponenten handelt, die von Grund auf neu definiert werden und nicht auf einer wiederverwendeten Komponente basieren. Override-Komponenten werden in der Schema-Ansicht auf ähnliche Weise wie Redefine-Komponenten hinzugefügt. Fügen Sie die überschreibende Komponente oberhalb der Zeile `end of override` hinzu und definieren Sie anschließend ihre Eigenschaften. *Siehe Abschnitt 'Redefine' weiter oben.* Der Hauptunterschied zwischen Override und Redefine ist, dass eine zu einem Override-Element hinzugefügte Komponente nicht auf einer Komponente aus dem wiederverwendeten Schema basiert.

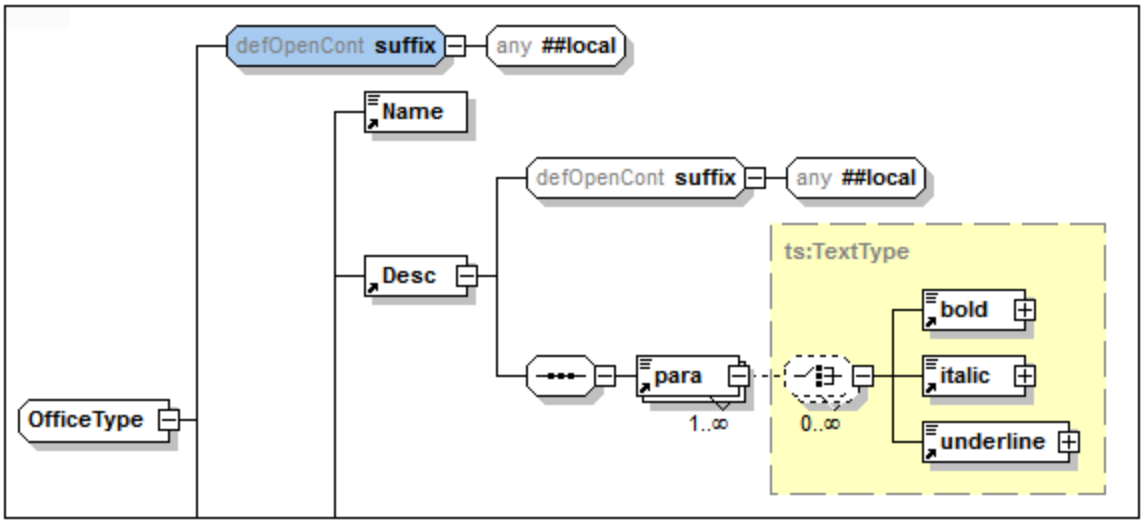
Default Open Content

Das Element `defaultOpenContent` ist neu in XSD 1.1 und definiert, dass ein oder mehrere nicht definierte Elemente zu jedem complexType mit mixed oder element-only Inhalt hinzugefügt werden können. Es ist dem Element `openContent` 209 (ebenfalls neu in XSD 1.1) ähnlich, mit dem Unterschied, dass das Element `defaultOpenContent` im Gegensatz zum Element `openContent`, das auf einen einzigen complexType im Schema angewendet wird, auf alle complexTypes im Schema angewendet wird.

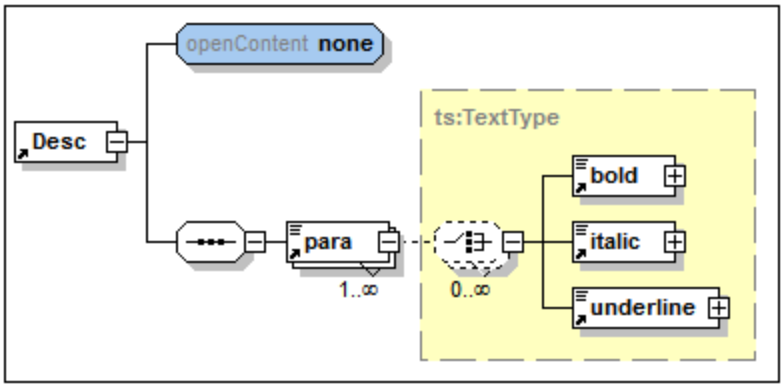
Das Element `defaultOpenContent` kommt nur einmal im Dokument vor (*siehe Abbildung unten*): nach Includes, Imports, Redefines und Overrides und vor der Definition von Komponenten. Es hat ein `mode` Attribut, das entweder den Wert `interleave` oder `suffix` erhalten kann. Die Standardeinstellung ist `interleave`.



Das Element `defaultOpenContent` hat ein Content Model, das Sie in der Content Model-Ansicht bearbeiten können. Nachdem es deklariert wurde, gilt das Element `defaultOpenContent` für alle complexTypes im Schema. In der Abbildung unten sehen Sie, dass `defaultOpenContent` automatisch auf die complexTypes `OfficeType` und `Desc` angewendet wurde.



Um das `defaultOpenContent` Element bei Anwendung auf einen bestimmten complexType außer Kraft zu setzen, fügen Sie ein `openContent` Child-Element zu diesem complexType hinzu. Zum `Desc` Element mit dem `defaultOpenContent` Element (*siehe Abbildung oben*) wurde in der Abbildung unten ein `openContent` Element hinzugefügt, das das `defaultOpenContent` Element außer Kraft setzt.



Globale Elemente (element)

In der Schema-Übersicht können Sie ein globales Element erstellen. Wenn das globale Element ein Content Model haben soll, wird dieses in der Content Model-Ansicht des globalen Elements definiert. Wenn das Element in einer dieser Ansichten ausgewählt ist, können Sie dafür auf den entsprechenden Registern des AAIDC-Bereichs[212] Attribute[213], Assertions[217] und Identity Constraints[220] definieren. Facets und andere Eigenschaften können in den Eingabehilfen Facets[235] und Details[233] für dieses Element definiert werden. Globale Elemente können anschließend von complexTypes referenziert werden.

Model Groups (group)

In der Schema-Übersicht können Sie benannte Model Groups erstellen, die in complexTypes referenziert werden können. Mit Hilfe einer benannten Model Group (Element `xs:group`) können Sie ein wiederverwendbares Content Model vordefinieren. Es kann eine von drei Arten von Child Model Groups enthalten: eine `sequence` Group, eine `choice` Group oder eine `all` Group.

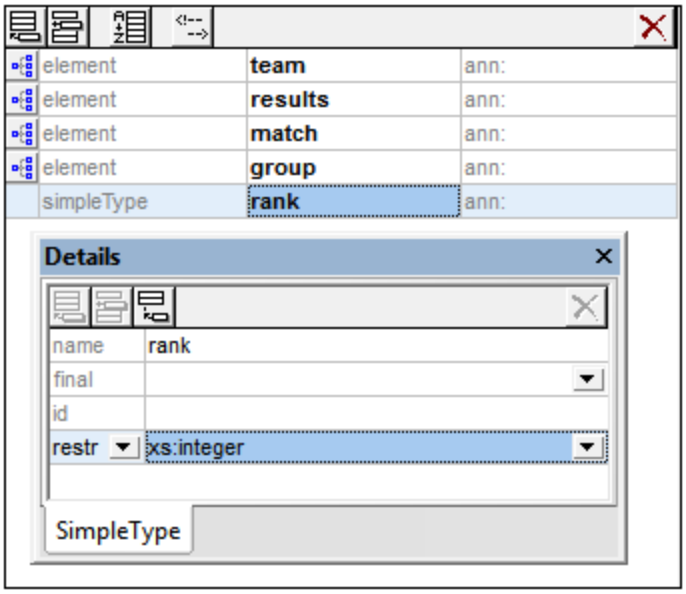
© 2016-2022 Altova GmbH

Durch Hinzufügen einer Group-Komponente, Benennen der Komponente und Definieren ihres Content Model in der Content Model-Ansicht können Sie in der Schema-Übersicht eine benannte Model Group erstellen. Die benannte Model Group kann anschließend zum Content Model eines complexType hinzugefügt werden.

Benannte simpleTypes (simpleType)

Sie können in der Schema-Übersicht benannte simpleTypes (*siehe Abbildung unten*) erstellen, die anschließend in Element- und Attributdeklarationen referenziert werden können.

In der Eingabehilfe "Details" wird der Inhalt des simpleType (`restriction`, `list`, `union`) und der entsprechende Typ, nämlich base type, item type bzw. member type, definiert. In der Abbildung unten ist der base type (Basistyp) der Restriction des simpleType `xs:integer`. Nähere Informationen dazu finden Sie im Abschnitt Eingabehilfe "Details" 233. Um einen simpleType mittels Facets einzuschränken, verwenden Sie die Optionen in der Eingabehilfe "Facets" 235.



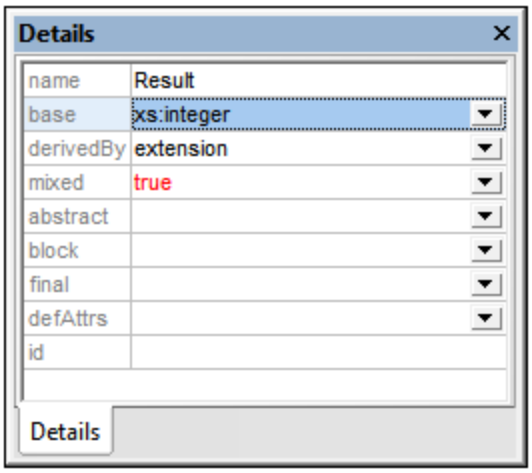
Anmerkung: Anonyme Typen können zu einem Element oder Attribut mit simple Content entweder in der Schema-Übersicht 177 oder in der Content Model-Ansicht 190 deklariert werden. Wenn Sie die Eigenschaft `derivedBy` (in der Eingabehilfe "Details") auf `restriction`, `list` oder `union` setzen, erstellen Sie innerhalb dieser Element- oder Attributdeklaration einen anonymen simpleType. In der Eingabehilfe "Facets" können Sie Restriction Facets und in der Eingabehilfe "Details" andere Eigenschaften definieren.

Benannte complexTypes (complexType)

In der Schema-Übersicht können Sie benannte complexTypes erstellen, die anschließend in Elementdeklarationen referenziert werden können. Wenn der benannte complexType in einer der Ansichten ausgewählt ist, können Sie seine Attribute 213 und Assertions 217 auf den entsprechenden Registern des AAIDC-Bereichs 212 definieren.

Ein complexType kann vier Arten von Inhalt haben (*siehe Liste unten*). Die verschiedenen Arten von Inhalt können in der Eingabehilfe "Details", wie unten beschrieben definiert werden. Falls der complexType ein Content Model haben kann, können Sie dieses bei Bedarf in der Content Model-Ansicht 190 definieren.

- *Simple content:* Definieren Sie den Basistyp (base type) des simple content (*siehe Abbildung unten*). Das Attribut `mixed` (für mixed Content) muss den Wert `false` (Standardwert) haben; daher wird `true` in der Abbildung unten rot angezeigt. Es ist kein Content Model gestattet.



- *Element-only content:* Erstellen Sie im Content Model-Diagramm Child-Elemente. Es gibt keinen Basistyp.
- *Mixed content:* Das Attribut `mixed` muss auf `true` gesetzt werden. Überall innerhalb des Elements können in seinen Child Element Nodes Zeichen vorkommen. Die Zeichen haben keinen Datentyp, daher kann es auch keinen Basistyp geben (*siehe Abbildung oben*). Im Content Model-Diagramm können Child-Elemente erstellt werden.
- *Empty content:* Das Element enthält weder Zeichen noch Child-Elemente. Es darf keinn Basistyp definiert sein und `mixed` muss auf false gesetzt sein. Die Daten in empty-content Elementen sind normalerweise in Attributen gespeichert.

Anmerkung: Für alle vier Arten von Content können (im AAIDC-Bereich)[212] Attribute und Assertions definiert werden.

Anmerkung: Um innerhalb eines Elements anonyme complexTypes zu erstellen, wird in der Content Model-Ansicht[190] ein Content Model für dieses Element erstellt.

Globale Attribute und Attributgruppen (attribute, attributeGroup)

Globale Attribute und Attributgruppen werden in der Schema-Übersicht hinzugefügt.

- Die Eigenschaften eines **globalen Attributs** werden in der Eingabehilfe "Details" des Attributs definiert.
- Nachdem Sie eine **globale Attributgruppe** erstellt haben, können Sie auf folgende Weise Attribute zur Gruppe hinzufügen: (i) Wählen Sie die globale Attributgruppe in der Liste der globalen Komponenten aus; (ii) fügen Sie im Abschnitt "Attribute" des AAIDC-Bereichs[212] Attribute hinzu und (iii) definieren Sie die Eigenschaften der einzelnen Attribute in der Eingabehilfe "Details" des ausgewählten Attributs.

Nachdem die globalen Attribute und Attributgruppen erstellt wurden, können sie in den Deklarationen von Elementen und complexTypes referenziert werden.

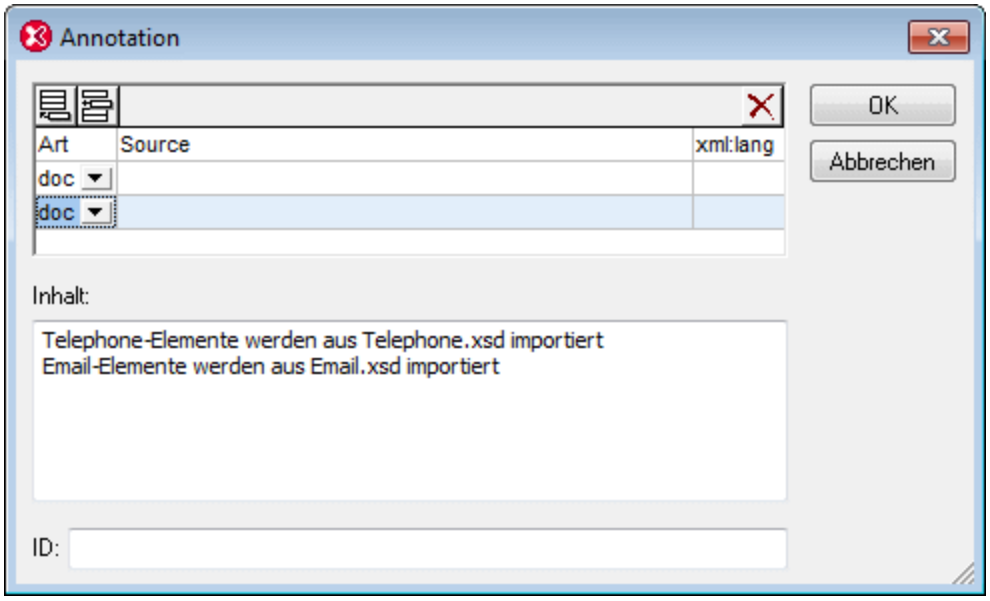
© 2016-2022 Altova GmbH

Notations (notation)

Notations sind immer global; es gibt keine lokalen Notations. Die Eigenschaften einer Notation werden in der Eingabehilfe "Details" der Notation definiert. Der Name der Notation kann direkt in der Liste der globalen Komponenten definiert werden. Alle Notations im Schema werden der Übersichtlichkeit halber in der Eingabehilfe "Komponenten" angezeigt.

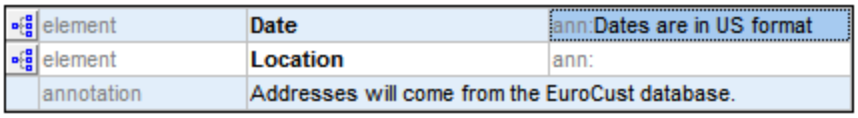
Globale Annotationen

Globale Annotationen sind globale Komponenten und nicht dasselbe wie die für einige globalenKomponenten verfügbaren optionalen Annotationen. Sie können eine globale Annotation in ihrem Textfeld in der Schema-Übersicht oder im Dialogfeld "Annotation" bearbeiten (*Abbildung unten*). Dieses Dialogfeld wird durch Rechtsklick auf die globale Annotationskomponente und Auswahl des Befehls **Alle Annotationsdaten** aufgerufen.



Jedes Annotation-Element kann ein `id` Attribut und mehrere `documentation` und/oder `appinfo` Child-Elemente haben. `documentation` oder `appinfo` Elemente können durch Klicken auf die Schaltflächen **Anhängen** oder **Einfügen** links oben im Dialogfeld und Auswahl des Eintrags `doc` oder `app` aus der entsprechenden Auswahlliste hinzugefügt werden. Wählen Sie ein `doc` oder `app` Element im oberen Bereich des Dialogfelds aus und geben Sie seinen Inhalt im Bereich *Inhalt* ein. Wenn Sie im Inhalt eine neue Zeile erstellen möchten (und den Inhalt dadurch zu mehrzeiligem Inhalt machen möchten), drücken Sie die **Eingabetaste**. In der Abbildung oben wurde das `documentation` Element ausgewählt. Sie sehen, dass es Inhalt in zwei Zeilen enthält. Für jedes `documentation` oder `appinfo` Element können Sie auch ein optionales `source` und `xml:lang` Attribut eingeben.

In der Schema-Übersicht wird nur das erste `documentation` oder `appinfo` Element der globalen Annotation angezeigt und kann direkt in der Liste der globalen Komponenten bearbeitet werden. Wenn dieser Inhalt mehrzeilig ist, werden bei Positionierung des Cursors über dieses Element alle Zeilen in einem mehrzeiligen Popup-Feld angezeigt. Um den Inhalt anderer `documentation` und/oder `appinfo` Elemente anzuzeigen oder zu bearbeiten, rufen Sie das Dialogfeld "Annotation" für diese globale Annotation auf.



Anmerkung: Die optionalen Annotationen, die für einige globale Komponenten zur Verfügung stehen, können ebenfalls auf genau die gleiche Weise wie oben beschrieben über das Dialogfeld "Annotation" bearbeitet werden.

Kommentare und Processing Instructions

Kommentare und Processing Instructions (Verarbeitungsanweisungen) können an jeder beliebigen Stelle in der Liste der globalen Komponenten in der Schema-Übersicht, nicht aber in der Content Model-Ansicht, hinzugefügt werden. Wenn es innerhalb von simpleTypes oder complexTypes einen odere mehrere Kommentare oder Processing Instructions gibt, werden diese gesammelt und an das Ende des einschließenden Objekts verschoben. In solchen Fällen wird daher empfohlen, Annotationen anstelle von Kommentaren zu verwenden.

4.4.3 Content Model-Ansicht

Ein Content Model dient zur Beschreibung der Struktur und des Inhalts einer Komponente. Die folgenden Komponenten können ein Content Model haben:

- ComplexTypes
- Elemente
- Model Groups
- Default Open Content

In der Liste der globalen Komponenten in der Schema-Übersicht[177] sind diese Komponenten durch das Symbol **Zur Content Model-Ansicht wechseln** links vom Komponentennamen gekennzeichnet.

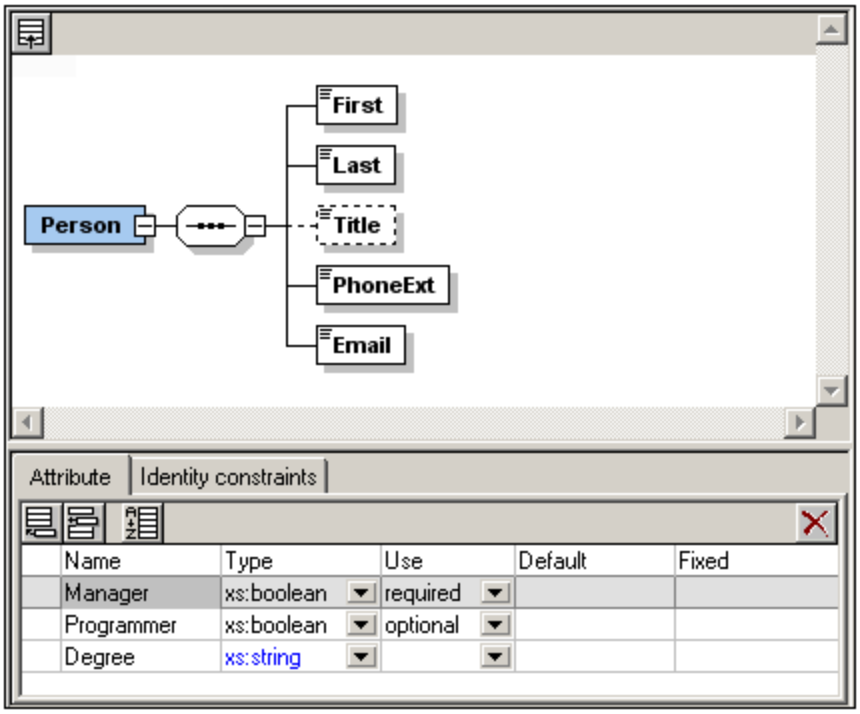
	*Zur Content Model-Ansicht wechseln:* Steht für globale Komponenten zur Verfügung, die ein Content Model haben. Öffnet das Content Model der globalen Komponente in der Content Model-Ansicht[190].
	*Globale Elemente anzeigen:* Steht in der Content Model-Ansicht zur Verfügung. Dient zum Wechseln in die Schema-Übersicht[177].

Wenn Sie auf das Symbol **Zur Content Model-Ansicht wechseln** klicken, wird die Content Model-Ansicht für diese globale Komponente (*siehe Abbildung unten*) angezeigt. Alternativ dazu (i) wählen Sie eine Komponente in der Schema-Übersicht[177] aus und anschließend die Menüoption **Schema-Design | Diagramm anzeigen,** oder (ii) doppelklicken Sie in der Eingabehilfe "Komponenten"[229] auf den Namen einer Komponente. Beachten Sie, dass immer nur ein Content Model in der Schemastruktur geöffnet sein kann. Wenn ein Content Model geöffnet ist, können Sie zum Content Model einer Komponente innerhalb des aktuellen Modells navigieren, indem Sie die **Strg**-Taste gedrückt halten und auf die gewünschte Komponente doppelklicken.

© 2016-2022 Altova GmbH

Allgemeine Beschreibung der Content Model-Ansicht

Das Content Model wird in der Content Model-Ansicht als Baumstruktur dargestellt (*siehe Abbildung unten*). Sie können das Aussehen der Baumstruktur im Dialogfeld "Schema-Ansicht konfigurieren" (Menübefehl **Schema Design | Ansicht konfigurieren**[1395]) anpassen.



Beachten Sie bitte die folgenden Punkte:

- Objekte in der Content Model-Ansicht können einem von zwei Typen angehören: Kompositoren und Komponenten. Zusätzlich dazu können in den Kästchen zu den Komponenten Attribute, Assertions, Identity Constraints und Open Content angezeigt werden.
- Jede Ebene in der Baumstruktur ist durch einen Kompositor mit den benachbarten Ebenen verbunden. Das Content Model kann um beliebig viele Ebenen erweitert werden.
- Über das Kontextmenü eines Objekts (Aufruf durch Rechtsklick auf das Objekt) kann ein Objekt relativ zu diesem Objekt hinzugefügt werden.
- Die Komponenten im Content Model können lokale Komponenten sein oder globale Komponenten referenzieren.
- Sie können Objekte mittels Drag-and-Drop verschieben.
- Sie können Objekte mit Hilfe der Tastenkombinationen **Strg + C** und **Strg + V** kopieren bzw. einfügen
- Sie können die Eigenschaften eines Objekts in der Details-Eingabehilfe und im AAIDC-Bereich[212] bearbeiten.
- Die Attribute, Assertions und Identity Constraints einer Komponente werden am unteren Rand der Content Model-Ansicht, im AAIDC-Bereich[212] angezeigt. Attribute und Identity Constraints können auch im Content Model-Diagramm anstatt im AAIDC-Bereich[212] angezeigt werden. Diese Ansichtsoption lässt sich im Dialogfeld Schema-Ansicht konfigurieren[1395] einstellen. Alternativ dazu können Sie auch die drei **Im Diagramm anzeigen** Schaltflächen der Schema-Design-Symbolleiste (*Abbildung unten*) verwenden.



- Sie können gleichrangige Komponenten sortieren, indem Sie die Komponenten, die Sie sortieren möchten, mit der rechten Maustaste anklicken und den Befehl **Komponenten sortieren** aus dem Kontextmenü wählen. Sie können die Priorität zweier Sortierkriterien wählen: (i) lokaler Name und (ii) Komponentenart.

Diese Funktionen werden in den Unterkapiteln dieses Abschnitts und im Tutorial näher erläutert.

Um zur Schema-Übersicht[177] zurückzukehren, klicken Sie auf das Symbol **Globale Elemente anzeigen** oder wählen Sie die Menüoption **Schema-Design | Globale Elemente anzeigen**.

4.4.3.1 Content Model-Objekte

Die im Diagramm in der Content Model-Ansicht gezeigten Objekte lassen sich in drei große Gruppen einteilen:

- Kompositoren[192]: sequence, choice, all
- Komponenten[193]: Element, complexType, Model Group, Wildcard
- Diverses[196]: Attribut, Attribute Group, Assertion, Constraint, Open Content

Die grafischen Darstellungen der einzelnen Objekte sind weiter unten im Detail beschrieben.

Kompositoren

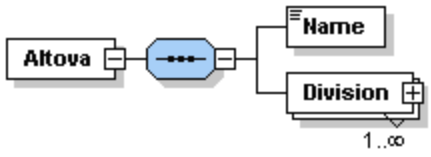
Ein **Kompositor** definiert die Reihenfolge, in der Child-Elemente angeordnet sind. Es gibt drei Kompositoren: `sequence`, `choice` und `all`.

So fügen Sie einen Kompositor ein:

1. Rechtsklicken Sie auf das Element, zu dem Sie Child-Elemente hinzufügen möchten.
2. Wählen Sie den Befehl **Child hinzufügen | Sequence** (oder **Choice** oder **All**).

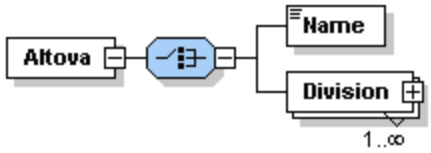
Daraufhin wird ein Kompositor hinzugefügt, der folgendermaßen aussieht:

- **Sequence**

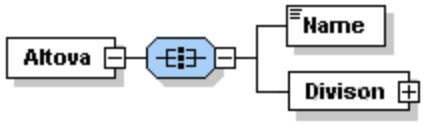


- **Choice**

© 2016-2022 Altova GmbH



- **All**



Um den Kompositor zu ändern, rechtsklicken Sie darauf und wählen Sie **Modell ändern | Sequence** (oder **Choice** oder **All**). Nachdem Sie den Kompositor hinzugefügt haben, müssen Sie (ein) Child-Element(e) oder eine Model Group hinzufügen.

Komponenten

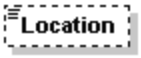
Unten finden Sie eine Liste von Komponenten, die in Content Models verwendet werden. Die grafische Darstellung der einzelnen Komponenten gibt Ihnen genaue Informationen über die Art der Komponente und ihre strukturellen Eigenschaften.

- Obligatorisches Einzelelement

  Details: Das Rechteck zeigt an, dass es sich um ein Element handelt, die durchgezogene Umrandung bedeutet, dass das Element "required" also obligatorisch ist. Das Fehlen eines Zahlenbereichs weist darauf hin, dass es ein Einzelelement ist (d.h. `minOcc=1` und `maxOcc=1`). Der Name des Elements ist `Country`. Die blaue Farbe zeigt an, dass das Element gerade ausgewählt ist (eine Komponente wird durch Anklicken ausgewählt). Wenn eine Komponente nicht ausgewählt ist, ist sie weiß.

- Optionales Einzelelement

  Details: Das Rechteck bedeutet, dass es sich um ein Element handelt und die gestrichelte Umrandung, dass das Element optional ist. Der fehlende Zahlenbereich gibt an, das es sich um ein Einzelelement handelt (d.h. `minOcc=0` und `maxOcc=1`. Der Element-Name ist `Location`.
  Anmerkung: Im Kontextmenü können Sie ein obligatorisches Element über die Menüoption **Optional** in ein optionales umwandeln.

- Obligatorisches mehrmals vorkommendes Element

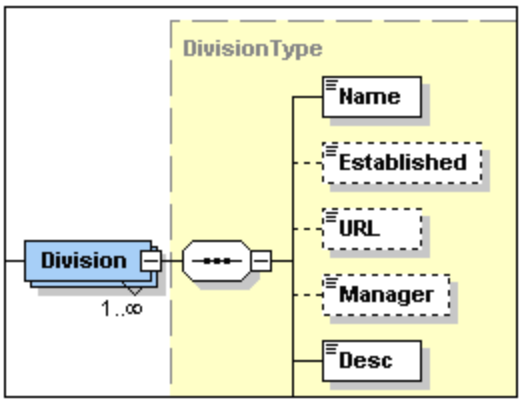


Details: Das Rechteck bedeutet, dass es sich um ein Element handelt und die durchgezogene Umrandung, dass das Element obligatorisch ist. Der Zahlenbereich 1..5 bedeutet `minOcc=1` und `maxOcc=5`. Der Element-Name ist `Alias`.

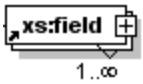
- Obligatorisches mehrfach vorkommendes Element, das Child-Elemente enthält



Details: Das Rechteck bedeutet, dass es sich um ein Element handelt und die durchgezogene Umrandung, dass das Element obligatorisch ist. Der Zahlenbereich 1..unendlich bedeutet, dass `minOcc=1` und `maxOcc=unbounded`. Das Plus-Zeichen bedeutet complex content (d.h. es hat mindestens ein Child-Element oder -Attribut). Der Element-Name ist `Division`.
Anmerkung: Über die Kontext-Menüoption **Unbounded** können Sie `maxOcc` in `unbounded` ändern. Wenn Sie auf das +-Zeichen des Elements klicken, wird die Baumstruktur erweitert und Sie sehen die Child-Elemente.



- Element, das ein globales Element referenziert



Details: Pfeil in der linken unteren Ecke bedeutet, dass das Element ein globales Element referenziert. Das Rechteck bedeutet, dass es sich um ein Element handelt und die durchgezogene Umrandung, das das Element obligatorisch ist. Der Zahlenbereich 1..unendlich bedeutet `minOcc=1` und `maxOcc=unbounded`. Das Plus-Zeichen bedeutet complex content (d.h. mindestens ein Child-Element oder -Attribut). Der Element-Name ist `xs:field`.
Anmerkung: Ein globales Element kann von simple- und complex type-Definitionen aus referenziert werden, sodass Sie eine globale Deklaration an mehreren Stellen in Ihrem Schema wiederverwenden können. Es gibt zwei Möglichkeiten, um eine Referenz auf ein globales Element zu erstellen: (i) durch Eingabe eines Namens für das lokale Element, der mit dem des globalen Elements identisch ist; und (ii) durch Rechtsklick auf das lokale Element und Auswahl der Kontextmenüoption **Reference**. Um die Definition eines globalen Elements anzusehen, halten Sie die **Strg**-Taste gedrückt und doppelklicken

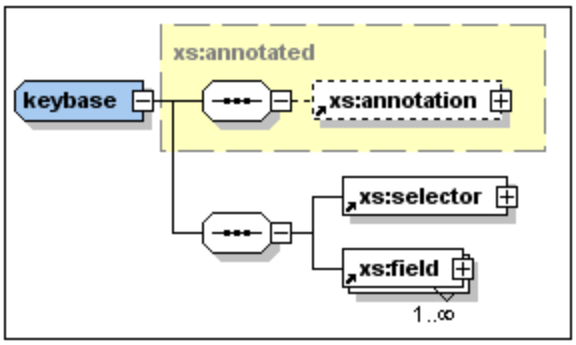
 © 2016-2022 Altova GmbH

Sie auf das Element. Alternativ dazu können Sie auch rechtsklicken und den Befehl **Gehe zu Definition** auswählen. Wenn Sie ein Element referenzieren, das nicht existiert, wird der Elementname rot angezeigt. Dies ist eine Warnung, dass es keine referenzierbare Definition gibt.

- ComplexType



Details: Ein unregelmäßiges Sechseck mit einem Plus-Zeichen (siehe oben) kennzeichnet einen complexType. Der hier dargestellte complexType hat den Namen `keybase`. Dieses Symbol (das unregelmäßige Sechseck mit einem Plus-Zeichen) bedeutet, dass es sich um einen globalen complexType handelt. Ein globaler complexType wird in der Schema-Übersicht deklariert. Sein Content Model wird normalerweise in der Content Model-Ansicht definiert. Ein globaler complexType kann für folgende Zwecke verwendet werden: (i) als Datentyp eines Elements oder (ii) als Base Type eines anderen complexType, indem Sie ihn in der Details-Eingabehilfe (entweder in der Content Model-Ansicht oder in der Schema-Übersicht) dem Element bzw. complexType zuweisen.



Der oben dargestellte complexType `keybase` wurde in der Schema-Übersicht mit einem Base Type `xs:annotated` deklariert. Der Base Type wird als Rechteck mit einer grau gestrichelten Umrandung und gelbem Hintergrund dargestellt. Anschließend wurden in der Content Model-Ansicht die Child-Elemente `xs:selector` und `xs:field` erstellt. (Beachten Sie die kleinen Pfeile in der linken unteren Ecke der Rechtecke mit der Beschriftung `xs:selector` und `xs:field`. Diese Pfeile bedeuten, dass beide Elemente globale Elemente dieses Namens referenzieren.)

Ein lokaler complexType wird direkt in der Content Model-Ansicht durch Erstellung eines Child-Elements oder -Attributs für ein Element definiert. Es gibt kein separates Symbol für lokale complexTypes.

Anmerkung: Der Base Type eines Content Models wird als Rechteck mit grau gestrichelter Umrandung und gelber Hintergrundfarbe dargestellt. Um das Content Model des Base Type aufzurufen, doppelklicken Sie auf seinen Namen.

- Model Group



Details: Das unregelmäßige Achteck mit einem Plus-Zeichen (siehe Abbildung oben) kennzeichnet eine Model Group. Eine Model Group gestattet Ihnen, Element-Deklarationen zu definieren und wiederzuverwenden.
Anmerkung: Wenn die Model Group (in der Schema-Übersicht) deklariert wird, erhält sie einen Namen. Anschließend wird (in der Content Model-Ansicht) der Content Mode durch Zuweisung eines Child-Kompositors, der die Element-Deklarationen enthält, definiert. Wenn die Model Group verwendet wird, wird sie (in der Content Model-Ansicht) innerhalb des Content Models einer anderen Komponente als Child eingefügt oder angehängt.

- Wildcards



Details: Das unregelmäßige Achteck mit `any` auf der linken Seite kennzeichnet eine Wildcard.
Anmerkung: Wildcards werden als Platzhalter für Elemente verwendet, die im Schema nicht definiert sind oder aus anderen Namespaces stammen. `##other` = Elemente können zu jedem Namespace gehören, bei dem es sich nicht um den im Schema definierten Target Namespace handelt; `##any` = Elemente können zu jedem Namespace gehören; `##targetNamespace` = Elemente müssen zum Target Namespace gehören, der im Schema definiert wurde; `##local` = Elemente können nicht zu jedem beliebigen Namespace gehören; *`anyURI`* = Elemente gehören zu dem von Ihnen angegebenen Namespace.

Diverse Objekte

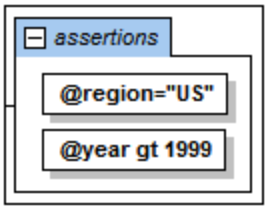
Dazu gehören Attribute, Attribute Groups, Assertions, Identity Constraints und Open Content.

- Attribute, Attribute Groups



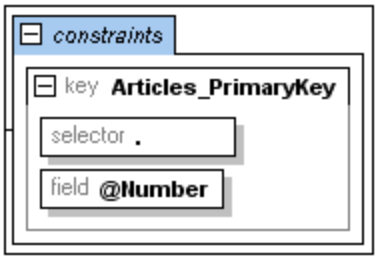
Details: Attribute werden durch das kursiv geschriebene Wort *'attributes'* in einem erweiterbaren Rechteck dargestellt. Jedes Attribut wird in einem Rechteck (i) mit einer strichlierten Umrandung dargestellt, wenn das Attribut optional ist und (ii) mit einer durchgezogenen Umrandung, wenn das Attribut obligatorisch ist. Attribute Groups und Attribute Wildcards sind ebenfalls im *'attributes'*-Rechteck enthalten.
Anmerkung: Attribute können im Diagramm und in der Details-Eingabehilfe bearbeitet werden. Sie können Attribute im Diagramm der Content Model-Ansicht oder im AAIDC-Bereich 212 unterhalb der Content Model-Ansicht anzeigen. Durch Klicken auf das Symbol "Attribute anzeigen" können Sie zwischen den beiden Ansichten hin- und herwechseln. Um die Reihenfolge von Attributen eines Elements zu ändern, ziehen Sie das Attribut aus dem Kästchen heraus und lassen Sie die Maustaste los, wenn an der gewünschten Stelle ein Pfeil erscheint.
- Assertions

© 2016-2022 Altova GmbH



Details: Werden in einem erweiterbaren Rechteck durch das kursiv geschriebene Wort *'assertions'* gekennzeichnet. Jede Assertion wird innerhalb des Kästchens "Assertions" in einem Rechteck angezeigt.
Anmerkung: Assertions können im Diagramm und in der Eingabehilfe "Details" bearbeitet werden. Sie können in der Content Model-Ansicht oder im AAIDC-Bereich[212] unterhalb der Content Model-Ansicht angezeigt werden. Durch Klicken auf das Symbol "Assertions anzeigen" können Sie zwischen diesen beiden Ansichten wechseln. Um die Reihenfolge der Assertions eines Elements zu ändern, ziehen Sie die Assertion mit der Maus an eine Stelle und lassen Sie die Maustaste los, wenn an der gewünschten Position ein Pfeil erscheint.

- Identity Constraints

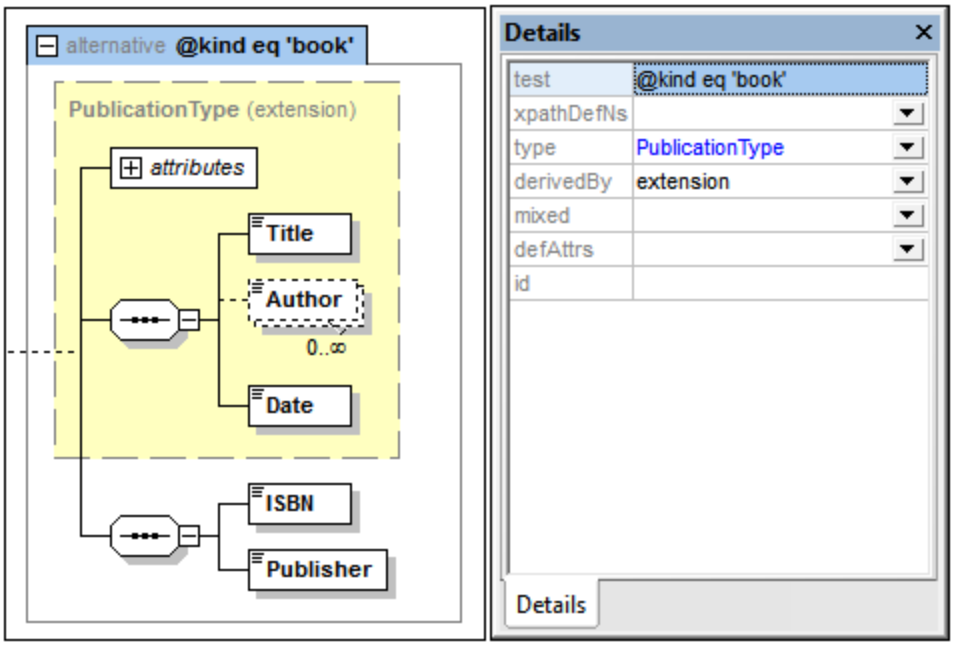


Details: Identity Constraints werden durch das kursiv gedruckte Wort '*constraints*' in einem erweiterbaren Rechteck dargestellt.
Anmerkung: Die im Content Model einer Komponente aufgelisteten Identity Constraints zeigen Constraints, die mit Hilfe der Elemente `key` und `keyref` definiert wurden und das Element `unique` haben. Identity Constraints, die mit Hilfe des `ID`-Datentyps definiert wurden, werden nicht im Content Model-Diagramm, sondern in der Details-Eingabehilfe angezeigt. Identity Constraints können in der Content Model-Ansicht oder auf dem Register "Identity Constraints" der Schema-Übersicht angezeigt und bearbeitet werden. In der Content Model-Ansicht können Sie den Constraints-Kasten durch Klicken auf das Symbol "Contraints anzeigen" ein- und ausblenden.

- Conditional Type Assignment



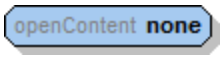
Details: Das Element `alternative` ist ein Rechteck, das den XPath-Ausdruck enthält, der ausgewertet werden soll (*siehe Abbildung oben*). Die Art des `alternative` Elements ist in der Eingabehilfe "Details" definiert. Wenn es sich um einen complexType handelt, wird dies im erweiterten Rechteck des `alternative` Elements angezeigt und kann dort weiter bearbeitet werdent (*siehe Abbildung unten*). SimpleType-Elemente werden im Diagramm nicht angezeigt, sondern können in der Eingabehilfe "Details" auf dem Register "Simple Type" definiert werden.



Anmerkung: Das Element `alternative` ist neu in XSD 1.1. Wenn der XPath-Ausdruck auf true ausgewertet wird, so wird der vom `alternative` Element definierte Typ verwendet. Es wird das erste `alternative` Element aus den gleichrangigen `alternative` Elementen, das auf true ausgewertet wird, ausgewählt. Daher spielt die Reihenfolge der alternativ-Elemente eine wichtige Reihenfolge. Die Reihenfolge kann durch Ziehen der `alternative` Elementkästchen an die gewünschte Position geändert werden. Eine ausführliche Beschreibung dazu finden Sie im Abschnitt Conditional Type Assignment 205.

- Default Open Content, Open Content





Details: Die Elemente `defaultOpenContent` und `openContent` sind in der Content Model-Ansicht mit `openContent` und `defOpenContent` beschriftet. Wildcard-Elementinhalt wird durch ein `any` Kästchen gekennzeichnet (*siehe Abbildung oben*).
Anmerkung: Die Elemente `defaultOpenContent` und `openContent` sind neu in XSD 1.1. Default Open Content ist eine globale Komponente und wird in der Schema-Übersicht 177 erstellt. In der Content Model-Ansicht einer Komponente können Sie das Element "Default Open Content" durch ein Element "Open Content" nur für diese Komponente ersetzen, das den Default Open Content des Schemas außer Kraft setzt. Fügen Sie das Open Content-Element einfach als Child der Komponente hinzu. Daraufhin wird das "Default Open Content" Kästchen durch ein "Open Content"-Kästchen ersetzt. In der Content Model-Ansicht können Sie das `mode` Attribut des Open Content und den Namespace seines Wildcard-Elements sowohl im Diagramm als auch in der Eingabehilfe "Details" bearbeiten. Sie können das Default Open Content-Element (für das gesamte Schema) auch in der Content Model-Ansicht über das entsprechende Kästchen für einen beliebigen complexType ändern.

© 2016-2022 Altova GmbH

Anmerkung:

- Vordefinierte Details, die Sie im Dialogfeld "Schema-Ansicht konfigurieren"[1395] definiert haben, können durch Klicken auf die Symbolleisten-Schaltfläche "Vordefinierte Details hinzufügen" ein- und ausgeblendet werden.
- Durch Auswahl des entsprechenden Symbols zur Anzeige von Attributen, Assertions und Identity Constraints können Sie auswählen, ob diese im Diagramm des Content Model selbst oder im AAIDC-Bereich[212] (unterhalb der Content Model-Ansicht) angezeigt werden sollen.
- In der Content Model-Ansicht können Sie zur Content Model-Ansicht einer beliebigen globalen Komponente im aktuellen Content Model springen, indem Sie die **Strg**-Taste gedrückt halten und auf die gewünschte Komponente doppelklicken.
- Das Kontextmenü von Komponenten enthält Befehle, um (i) zur Definition einer Komponente und (ii) zur Typdefinition einer Komponente, falls vorhanden, zu springen.

4.4.3.2 Bearbeiten in der Content Model-Ansicht

Die Anleitung zur Bearbeitung in der Content Model-Ansicht ist in die folgenden Abschnitte gegliedert:

- Konfigurieren der Content Model-Ansicht[199]
- Attribute, Assertions und Identity Constraints[199]
- Symbole der Content Model-Ansicht[200]
- Kontextmenü-Operationen[200]
- Tastaturkürzel und Drag-and-Drop[203]
- Komponenteneigenschaften[204]
- Annotationen[204]
- Kommentare und Processing Instructions[205]
- Dokumentieren des Content Model[205]

Konfigurieren der Content Model-Ansicht

Sie können die Content Model-Ansicht für das gesamte Schema im Dialogfeld "Schema-Ansicht konfigurieren" (**Schema-Design | Ansicht konfigurieren**) konfigurieren. Nähere Informationen über die Konfigurationsoptionen finden Sie weiter hinten, unter Menübefehle unter Ansicht konfigurieren[1395]. Beachten Sie, dass die Einstellungen, die Sie hier definieren, auf das gesamte Schema sowie auf die Ausgabe der Schema-Dokumentation und die Druckausgabe angewendet werden.

Attribute, Assertions und Identity Constraints

Die Attribute, Assertions und Identity Constraints einer Komponente können in einem Bereich unterhalb der Content Model-Ansicht, nämlich dem AAIDC-Bereich[212], oder in Kästchen in der Content Model-Ansicht selbst, also im Diagramm, angezeigt werden. Diese zweite Ansichtsoption kann im Dialogfeld Schema-Ansicht konfigurieren[1395] eingestellt werden. Alternativ dazu können Sie auch die drei **Im Diagramm anzeigen** Symbolleisten-Schaltflächen (*Abbildung unten, siehe auch Liste der Symbole unten*) verwenden.



Eine Anleitung zum Einfügen und Bearbeiten von Attributen, Assertions und Identity Constraints finden Sie im Abschnitt Attribute, Assertions und Identity Constraints 212.

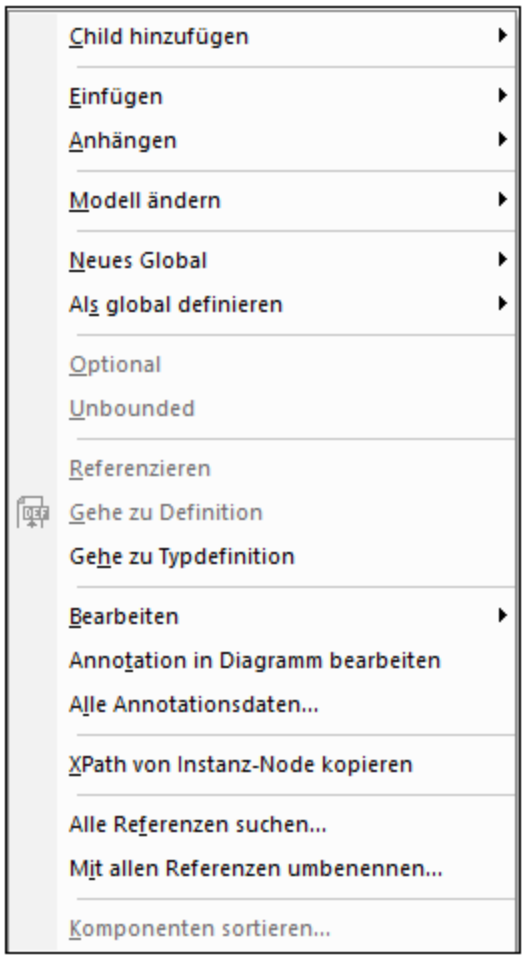
Symbole der Content Model-Ansicht

	*Globale Elemente anzeigen:* Steht in der Content Model Ansicht zur Verfügung. Wechselt in die Schema-Übersicht 177.
	*Vordefinierte Details hinzufügen:* In der Schema-Design-Symbolleiste und aktiviert in der Content Model-Ansicht. Blendet die vordefinierten Details in Komponenten ein und aus.
	*Attribute im Diagramm anzeigen:* In der Schema-Design-Symbolleiste und aktiviert in der Content Model-Ansicht. Wechselt zwischen der Anzeige der Attribute im Diagramm (aktiviert) und auf dem Register "Attribute".
	*Assertions im Diagramm anzeigen:* In der Schema-Design-Symbolleiste und aktiviert in der Content Model-Ansicht. Wechselt zwischen der Anzeige der Assertions im Diagramm (aktiviert) und auf dem Register "Assertions".
	*Constraints im Diagramm anzeigen:* In der Schema-Design-Symbolleiste und aktiviert in der Content Model-Ansicht. Wechselt zwischen der Anzeige der Identity Constraints im Diagramm (aktiviert) und auf dem Register "Identity Constraints".
	*Identity Constraints visualisieren:* In der Schema-Design-Symbolleiste und aktiviert in der Content Model-Ansicht. Blendet Identity Constraints-Informationen ein und aus.

Kontextmenü-Operationen

Über das Kontextmenü, das aufgerufen wird, wenn Sie mit der rechten Maustaste in die Content Model-Ansicht klicken, (*Abbildung unten*), werden verschiedene Bearbeitungsoperationen in der Content Model-Ansicht ausgeführt. Es sind nur Befehle für Operationen aktiviert, die an dieser Stelle im Content Model ausgeführt werden können. Die Operationen werden relativ zu dem Objekt ausgeführt, auf das Sie mit der rechten Maustaste geklickt haben. Wenn z.B. ein Child hinzugefügt wird, wird es relativ zum angeklickten Objekt hinzugefügt.

© 2016-2022 Altova GmbH



Im Folgenden finden Sie eine Liste der im Kontextmenü verfügbaren Operationen.

- *Child-Kompositoren und -Komponenten hinzufügen:* Mit dem Befehl **Child hinzufügen** wird ein Untermenü aufgerufen, in dem Sie den gewünschten Kompositor / die gewünschte Komponente auswählen können.

- *Kompositoren und Komponenten einfügen/anhängen:* Fügt den Kompositor oder die Komponente auf derselben hierarchischen Ebene ein, auf der sich das ausgewählte Objekt befindet. Mit **Einfügen** wird das Objekt vor dem ausgewählten Objekt eingefügt, mit **Anhängen** wird es nach dem letzten gleichrangigen Element des ausgewählten Objekts hinzugefügt.

- *Kompositor ändern:* Rechtsklicken Sie auf einen Kompositor und wählen Sie **Modell ändern | <neuer Kompositor>**.

- *Globale Komponenten erstellen:* (i) Der Befehl **Neues Global** kann durch einen Klick an eine beliebige Stelle in der Content Model-Ansicht aufgerufen werden. Daraufhin wird ein Untermenü aufgerufen, aus dem Sie die gewünschte neue globale Komponente auswählen können. (ii) Wenn ein Objekt als globale Komponente erstellt werden kann, ist der Befehl **Als global definieren** im Kontextmenü aktiv. Bei Auswahl dieses Befehls wird das Objekt als globale Komponente in der Content Model-Ansicht erstellt und das Objekt selbst enthält eine Referenz auf die neu erstellte globale Komponente.

- *Occurrence-Definition ändern:* Mit Hilfe der Ein/Aus-Befehle **Optional** und **Unbounded** können Sie die gewünschte Occurrence-Einstellung definieren: (i) *optional* = 0 oder 1; (ii) *optional* + *unbounded* = 0 bis unendlich; (iii) *unbounded* = 1 bis unendlich; (iv) *nicht optional* + *nicht unbounded* = 1. (Anmerkung: *optional* definiert das `minOccurs`-Attribut der Komponente, *unbounded* definiert das `maxOccurs`-Attribut.)

- *Zwischen lokalen und globalen Definitionen wechseln:* Wenn es ein globales Element gibt, das denselben Namen wie ein lokales Element hat, können Sie mit Hilfe des Ein/Aus-Befehls **Referenzieren** zwischen dem Referenzieren der globalen Definition (Befehl ein) und der Verwendung der lokalen Definition (Befehl aus) wechseln.

- *Zu einem anderen Content Model springen:* Klicken Sie mit der rechten Maustaste auf das Content Model, zu dem Sie wechseln möchten, und wählen Sie den Befehl **Gehe zu Definition**. (Der Befehl ist nur bei den Komponenten aktiv, die ein Content Model haben können.) Alternativ dazu können Sie **Strg** drücken und auf die Komponente doppelklicken.

- *Zur Typdefinition der Komponente gehen:* Wenn eine Komponente eine Typdefinition hat (simple oder complexType), gelangen Sie bei Aufruf des Befehls **Gehe zu Typdefinition** zur Typdefinition. Bei simpleTypes wird ein Meldungsfeld mit Informationen über den simpleType angezeigt.

- *Vordefinierte Details bearbeiten:* Wenn laut Einstellung (über die Schaltfläche **Vordefinierte Details hinzufügen** in der Symbolleiste "Schema-Design) vordefinierte Details im Diagramm angezeigt werden, so wird mit dem Befehl **Bearbeiten** ein Untermenü mit den vordefinierten Details angezeigt. Wählen Sie das gewünschte vordefinierte Detail aus und bearbeiten Sie seinen Wert im Diagramm.

- *Kompositor-/Komponenten-Annotationen erstellen und bearbeiten:* Mit dem Befehl **Annotation bearbeiten** wird unterhalb des Kompositors/der Komponente ein Bereich für Annotationen angelegt (*siehe Abbildung unten*). Hier können Sie die Annotation eingeben und bearbeiten. Wenn die Annotation bereits vorhanden ist, wird der Annotationstext bei Aufruf des Befehls für die Bearbeitung markiert. Eine schnellere Methode für die Bearbeitung ist ein Doppelklick auf bestehenden Annotationstext.

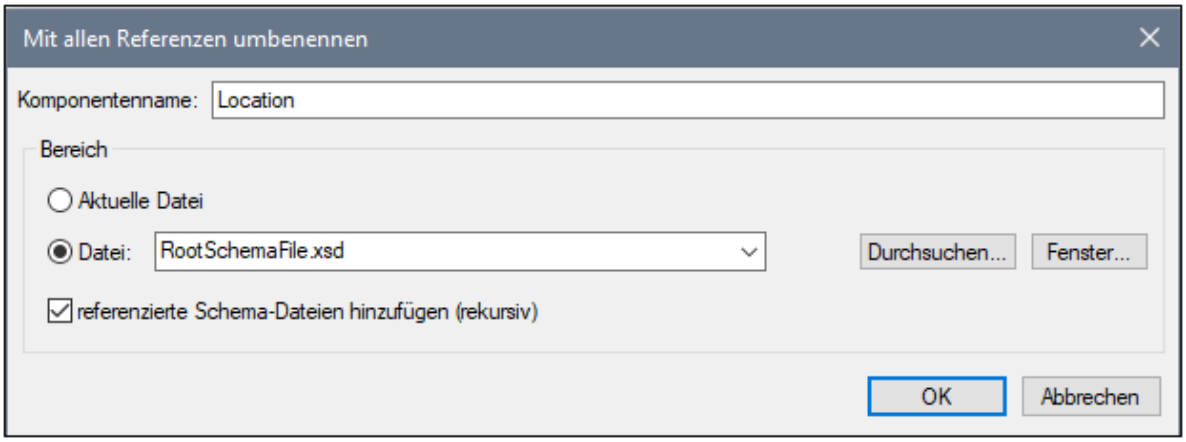
  

  Im XML-Schema-Dokument wird die Annotation innerhalb des Elements `annotation/documentation` des Kompositors/der Komponente erstellt. *Siehe auch Abschnitt weiter unten zur Dokumentation.*

- *XPath von Instanz-Node kopieren:* Der Befehl **XPath von Instanz-Node kopieren** ist für innerhalb eines globalen Elements oder globalen complexType definierte Elemente und Attribute aktiv. Der Befehl kopiert einen XPath-Ausdruck zum Aufrufen des ausgewählten Nodes in die Zwischenablage. Dieser XPath-Ausdruck beginnt bei der globalen Komponente, deren Content Model gerade in der Content Model-Ansicht angezeigt wird.

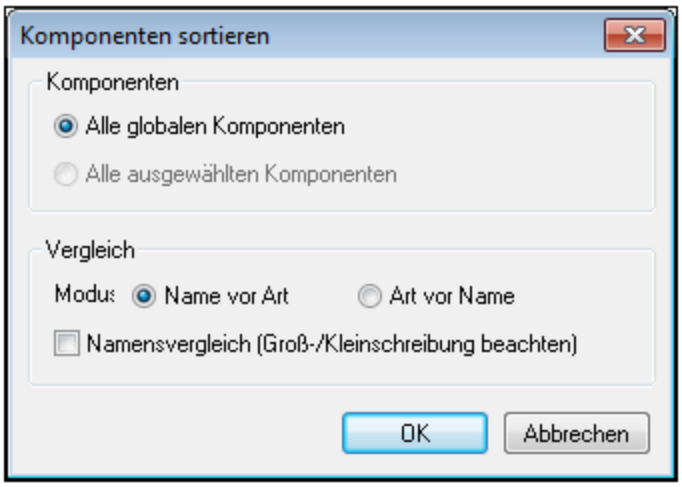
- *Komponente suchen und umbenennen:* Die Befehle **Alle Referenzen suchen** und **Mit allen Referenzen umbenennen** sind für globale Elemente aktiviert. Diese Befehle suchen alle Instanzen der ausgewählten Komponente bzw. benennen diese im aktiven Dokument und optional in allen mit dem aktiven Dokument in Zusammenhang stehenden Schema-Dateien um.

 © 2016-2022 Altova GmbH



In der Abbildung oben wird der Name der mit Rechtsklick markierten Komponente und aller ihrer Referenzen innerhalb des Suchbereichs durch den Namen `Email` ersetzt. Nähere Informationen dazu finden Sie unter Suchen und Umbenennen von globalen Komponenten 495.

- *Deklarationen und Referenzen sortieren:* Mit dem Befehl **Sortieren** können alle ausgewählten Komponenten oder die gleichrangigen Elemente der ausgewählten Komponente sortiert werden. Nehmen Sie die Sortiereinstellungen im Dialogfeld "Komponenten sortieren" (*Abbildung unten*) vor und klicken Sie auf **OK**.



Um mehrere Komponenten auszuwählen, halten Sie beim Anklicken Sie **Umschalt**- oder **Strg**-Taste gedrückt. Sie können die Komponenten zuerst nach Namen und dann nach Art sortieren oder umgekehrt.

Anmerkung: Sie können eine Komponente auswählen und sie kopieren, ausschneiden, löschen oder mit der Maus ziehen. In einigen Fällen, wie z.B. bei Attributen einer `complexType`-Einschränkung ist dies eventuell nicht zulässig.

Tastaturkürzel und Drag-and-Drop

Mit Hilfe der Tastaturkürzel **Strg+c** und **Strg+v** können Sie Elemente in der Content Model-Ansicht kopieren und einfügen. Kopierte Objekte werden als Child-Objekte des ausgewählten Objekts eingefügt. Wo dies aus Gründen der Struktur nicht möglich ist, wird eine entsprechenden Meldung angezeigt.

Außerdem können Sie (i) Objekte mit Drag-and-Drop an eine andere Stelle im Diagramm verschieben (ii) einige Komponenten wie z.B. Attribute mit Drag-and-Drop aus der Eingabehilfe "Komponenten" in das Diagramm ziehen.

Komponenteneigenschaften

Wenn die Content Model-Ansicht so konfiguriert ist, dass die Komponenten mit vordefinierten Details im Komponentenkästchen angezeigt werden, können Sie diese Informationen direkt im Diagramm bearbeiten. Die vordefinierten Details können durch Klicken auf die Symbolleisten-Schaltfläche **Vordefinierte Details hinzufügen** (*siehe Liste der Symbole oben*) ein- und ausgeblendet werden.

Alternativ dazu können Sie die Eigenschaften einer Komponente in der Eingabehilfe "Details" 233 bearbeiten. Die Änderungen werden in den Platzhalterfeldern angezeigt, falls eingestellt wurde, dass diese angezeigt werden.

Annotationen

Die XML-Schema-Annotationen befinden sich im Element `annotation`. Es gibt zwei Annotationsarten, von denen jede in einem anderen Child-Element von `annotation` enthalten ist:

- `documentation` Child: Enthält Informationen, die bei der Bearbeitung des Schemas nützlich sein könnten.
- `appinfo` Child: Hier kann ein Skript oder Informationen eingefügt werden, die von einer Processing Instruction gelesen werden kann

Unten finden Sie den Text eines annotation-Elements, das beide Arten von Child-Elementen enthält.

```
<xs:element name="session_date" type="xs:dateTime" nillable="true">
   <xs:annotation>
      <xs:documentation>Date and time when interview was held</xs:documentation>
      <xs:appinfo source="http://www.altova.com/datehandlers/interviews">separator =
:</xs:appinfo>
   </xs:annotation>
</xs:element>
```

In der Content Model-Ansicht können Sie folgendermaßen Annotationen für einzelne Kompositoren und Komponenten erstellen.

1. Rechtsklicken Sie auf den Kompositor oder die Komponente.



2. Wählen Sie die Kontextmenüoption **Alle Annotationsdaten**. Damit wird das Dialogfeld "Annotation" (*siehe Abbildung unten)* aufgerufen. Wenn für dieses Element eine Annotation (entweder `documentation` oder `appinfo`) vorhanden ist, sehen Sie dafür im Dialogfeld eine entsprechende Zeile.

 © 2016-2022 Altova GmbH



3. Um ein `appinfo`-Element zu erstellen, klicken Sie auf das Symbol "Anhängen" oder "Einfügen" in der linken oberen Ecke, um eine neue Zeile anzuhängen bzw. einzufügen.
4. Wählen Sie im Feld "*Art*" der neuen Zeile die Option `app` aus der Dropdown-Liste aus.
5. Geben Sie im Bereich "Inhalt" des Dialogfelds das Skript oder die Informationen ein, die von der Applikation verarbeitet werden sollen.
6. Optional dazu können Sie im Feld "*Source*" eine Quell-URI eingeben, unter der die verarbeitende Applikationen weitere Informationen abrufen kann.

Kommentare und Processing Instructions

Wenn XML-Schema-Dokumente in XMLSpy geladen oder Ansichten gewechselt werden, werden Kommentare und Processing Instructions innherhalb von simpleTypes und complexTypes gesammelt und an das Ende des einschließenden Objekts verschoben. In solchen Fällen wird daher empfohlen, Annotationen anstelle von Kommentaren zu verwenden.

Dokumentieren des Content Model

Sie können zu Ihrem Schema eine detaillierte Dokumentation[1389] in den Formaten HTML und MS Word erstellen. Zu jeder globalen Komponente wird detaillierte Dokumentation generiert und die Liste der globalen Komponenten wird auf einer Inhaltsverzeichnisseite angezeigt, über die Sie Verknüpfungen zu den Content Models der einzelnen Komponenten anlegen können. Außerdem werden miteinander in Zusammenhang stehende Elemente (wie z.B. Child-Elemente oder complexTypes) durch Hyperlinks referenziert, so dass Sie von einem Element zum anderen navigieren können. Um Schema-Dokumentation zu generieren, wählen Sie den Menübefehl **Schema-Design | Dokumentation generieren.**

4.4.3.3 Conditional Type Assignment

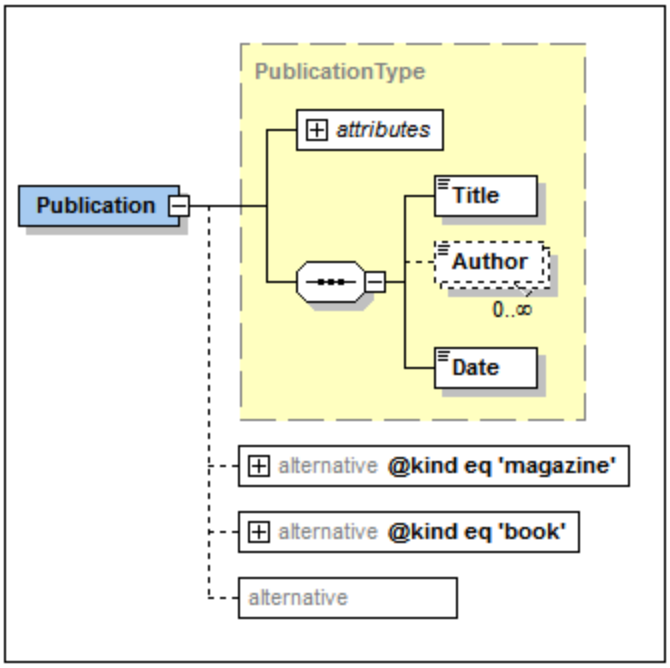
Ein Conditional Type Assignment ist ein XSD 1.1 Feature, mit Hilfe dessen der Typ eines Elements anhand des Inhalts des XML-Dokuments, genauer gesagt, anhand des Werts der Attribute des Elements oder anhand

des Vorhandenseins oder Fehlens von Attributen ermittelt wird. Angenommen, das XML-Dokument hat das folgende Element:

```
<publication kind="magazine">
   ...
</publication>
```

Im Schema kann definiert werden, dass der Typ des Elements `publication` abhängig vom Wert des Attributwerts `@kind` des Instanzelements ein unterschiedlicher sein kann. Im Schema wird dies mit Hilfe des Elements `alternative` definiert, welches neu in XSD 1.1 ist. Es werden mehrere Typen definiert, von denen sich jeder in einem `alternative` Element befindet.

In der Abbildung unten ist das Element `Publication` mit drei `alternative` Child-Elementen deklariert, von denen zwei ein `test` Attribut (`@kind eq 'magazine'` und `@kind eq 'book'`) haben. Das dritte `alternative` Element hat kein `test` Attribut, sondern eine simpleType-Zuweisung `xs:error` (diese Zuweisung erfolgt in der Eingabehilfe "Details" und wird im Diagramm nicht angezeigt). Wenn dieser Typ ausgelöst wird, wird ein XML-Validierungsfehler zurückgegeben.



Unten sehen Sie das XML-Fragment für die obigen Deklarationen:

```
<xs:complexType name="PublicationType">
   <xs:sequence>
      <xs:element name="Title" type="xs:string"/>
      <xs:element name="Author" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="Date" type="xs:gYear"/>
   </xs:sequence>
   <xs:attribute name="kind" type="xs:string"/>
</xs:complexType>

<xs:complexType name="MagazineType">
   <xs:complexContent>
      <xs:restriction base="PublicationType">
         <xs:sequence>
```

 © 2016-2022 Altova GmbH

```
            <xs:element name="Title" type="xs:string"/>
            <xs:element name="Date" type="xs:gYear"/>
         </xs:sequence>
      </xs:restriction>
   </xs:complexContent>
</xs:complexType>

<xs:element name="Publication" type="PublicationType">
   <xs:alternative test="@kind eq 'magazine'" type="MagazineType"/>
   <xs:alternative test="@kind eq 'book'">
      <xs:complexType>
         <xs:complexContent>
            <xs:extension base="PublicationType">
               <xs:sequence>
                  <xs:element name="ISBN" type="xs:string"/>
                  <xs:element name="Publisher" type="xs:string"/>
               </xs:sequence>
            </xs:extension>
         </xs:complexContent>
      </xs:complexType>
   </xs:alternative>
   <xs:alternative type="xs:error"/>
</xs:element>
```

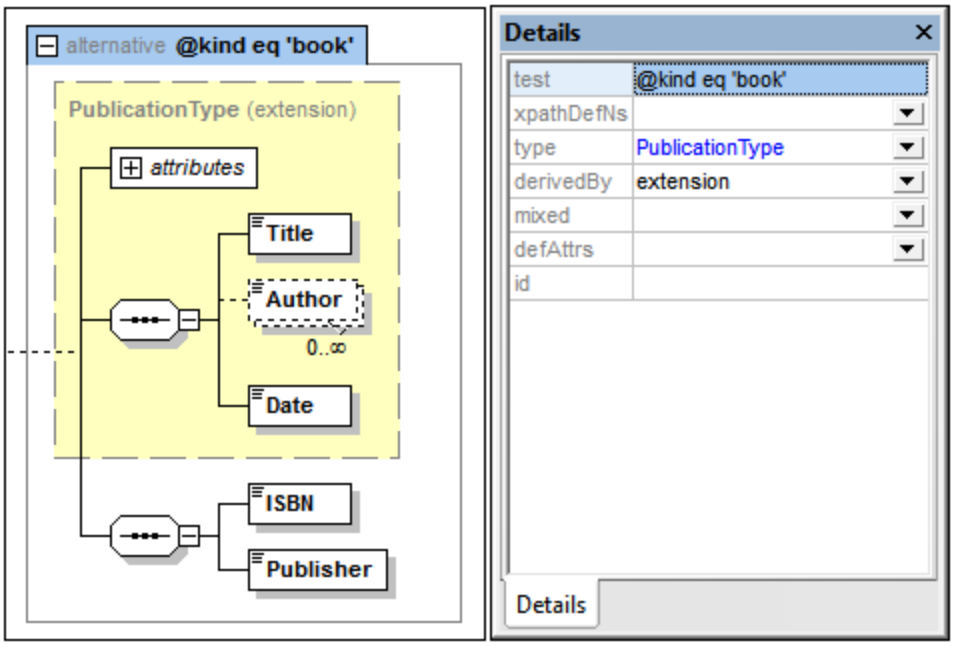
Beachten Sie die folgenden Punkte:

- Aus den gleichrangigen `alternative` Elementen wird das erste `alternative` Element, das auf true ausgewertet wird, ausgewählt. Daher ist es wichtig, welche Reihenfolge die alternative Elemente haben. Sie können deren Reihenfolge in der Content Model-Ansicht durch Ziehen der `alternative` Elementkästchen an die gewünschte Stelle ändern.
- Beachten Sie, dass das `Publication` Element einen Typ hat (`PublicationType`). Dieser Typ wird als Standardtyp verwendet, wenn keines der `alternative` Elemente verwendet wird. In unserem Beispiel oben wird jedoch das `alternative` Element vom Typ `xs:error` verwendet, wenn beide vorherigen `alternative` Elementbedingungen `false` zurückgeben.
- Wenn keine `alternative` Elementbedingung erfüllt wird und das Element keinen Standardtyp hat, so wird dem Element der Typ `anyType` zugewiesen. In diesem Fall kann das Element jeden wohlgeformten XML-Inhalt haben.
- Das Element `alternative` und der simpleType `xs:error` sind neu in XSD 1.1.

Bearbeitung in der Content Model-Ansicht

Über das Kontextmenü des Elements (*siehe Content Model in der Abbildung oben*) können Sie einen alternative-Typ als Child zu einer Elementdeklaration hinzufügen.

Der Typ des `alternative` Elements wird in der Eingabehilfe "Details" definiert. Wenn der Typ ein complexType ist, wird er im erweiterten Rechteck des `alternative` Elements angezeigt und kann dort weiter bearbeitet werden (*siehe Abbildung unten*). SimpleTypes werden im Diagramm nicht angezeigt, können aber auf dem Register "Simple Type) der Eingabehilfe "Details" definiert werden.



Anmerkung: Über das Attribut `xpathDefaultNamespace` können Sie in der Eingabehilfe "Details" einen Namespace für den XPath-Ausdruck definieren. Nähere Informationen zu XPath-Standard-Namespaces finden Sie im Abschnitt unten.

Verwendung von xpathDefaultNamespace

Ein im XML-Schema-Dokument deklarierter Standard-Namespace ist der Standard-Namespace des XML-Schema-Dokuments. Er gilt für Elementnamen ohne Präfix im Schema-Dokument, nicht aber für Elementnamen ohne Präfix in XPath-Ausdrücken im Schema-Dokument.

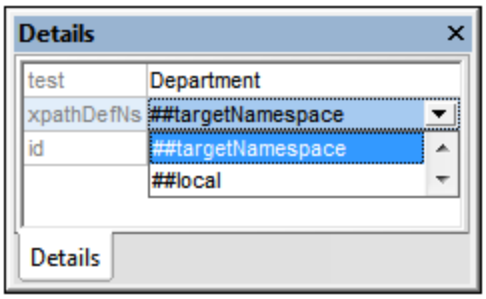
Mit dem Attribut `xpathDefaultNamespace` (einem XSD 1.1-Feature) wird der Namespace definiert, dem Elementnamen ohne Präfix im XPath-Ausdruck angehören.

Der Geltungsbereich von XPath-Standard-Namespaces erstreckt sich auf die XML-Schema-Elemente, für die diese Namespaces deklariert wurden. Das Attribut `xpathDefaultNamespace` kann in den folgenden XML Schema 1.1-Elementen vorkommen:

- `xs:schema`
- `xs:assert` and `xs:assertion`
- `xs:alternative`
- `xs:selector` and `xs:field` (in Identity Constraints)

Das `xpathDefaultNamespace` Attribut für `xs:schema` wird im Dialogfeld "Schema-Einstellungen" (**Schema-Design | Schema-Einstellungen**) im XSD 1.1-Modus definiert. Für die anderen oben aufgelisteten Elemente wird es in der jeweiligen Details-Eingabehilfe (*Beispiel in der Abbildung unten*) definiert.

© 2016-2022 Altova GmbH



Durch Deklarieren des XPath-Standard-Namespace für `xs:schema` wird der XPath-Standard-Namespace für den Geltungsbereich des gesamten Schemas deklariert. Sie können diese Deklaration für Elemente, in denen das Attribut `xpathDefaultNamespace` zulässig ist (*siehe Liste oben*), außer Kraft setzen.

Anstelle eines tatsächlichen Namespace kann das Attribut `xpathDefaultNamespace` eines von drei Schlüsselwörtern enthalten:

- `##targetNamespace`: Der XPath-Standard-Namespace ist mit dem Target Namespace des Schemas identisch.
- `##defaultNamespace`: Der XPath-Standard-Namespace ist mit dem Standard-Namespace des Schemas identisch.
- `##local`: Es gibt keinen XPath-Standard-Namespace.

Wenn im Dokument kein XPath-Standard-Namespace deklariert ist, haben Elemente ohne Präfix in XPath-Ausdrücken keinen Namespace.

Anmerkung: Die XPath-Standard-Namespace-Deklaration gilt nicht für Attribute.

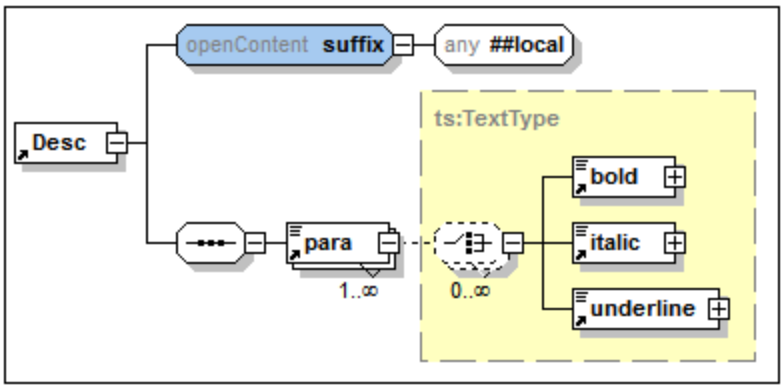
4.4.3.4 Offene Content Models

Offene Content Models sind neu in XSD 1.1. Sie werden für complexTypes deklariert. Mit Hilfe von offenen Content Models kann jedes beliebige Element (d.h. ein im Content Model des complexType nicht definiertes Element) beliebig oft entweder (i) zwischen im Content Model definierten Elementen oder (ii) nach im Content Model definierten Elementen vorkommen.

Das Element `openContent` ist ein Child des complexType und steht einmal vor dem Content Model des complexType (*siehe Abbildung unten*).

Modus

Das Element `openContent` hat ein obligatorisches `mode` Attribut, das die Werte `interleave`, `suffix` oder `none` (*siehe Abbildung unten*) haben kann. Der Standardwert ist `interleave`.



Diese Werte haben die folgende Bedeutung:

- Wenn als Modus `mode="interleave"` oder `mode="suffix"` eingestellt ist, so muss Wildcard Elementinhalt (`xs:any`) ohne eine Einschränkung durch minimum oder maximum occurrence vorhanden sein. Daraus folgt, dass beliebig viele nicht definierte Elemente (Wildcards) zulässig sind.
- Wenn der Modus `interleave` ist, können vor oder nach einzelnen definierten Elementen im Content Model beliebig viele nicht definierte Elemente stehen. Sie stehen zwischen definierten Elementen.
- Wenn der Modus `suffix` ist, können hinter dem letzten definierten Element des Content Model beliebig viele nicht definierte Elemente stehen.
- Wenn der Modus `none` ist, dürfen keine nicht definierten Elemente (`xs:any` Child) vorkommen; das Content Model ist nicht offen. Mit Hilfe des Werts `none` wird das Element `defaultOpenContent` 182, das für das gesamte Schema gilt, außer Kraft gesetzt.

Offener Content wird in der Content Model-Ansicht (über den Kontextmenübefehl **Child hinzufügen**) als Child des complexType hinzugefügt. Definieren Sie den Modus entweder durch Doppelklick in das `openContent` Kästchen im Diagramm (*siehe Abbildung oben*) und Auswahl eines Werts (`interleave`, `suffix` oder `none`) oder durch Auswahl eines Werts in der Eingabehilfe "Details".

Wildcard (xs:any) Eigenschaften

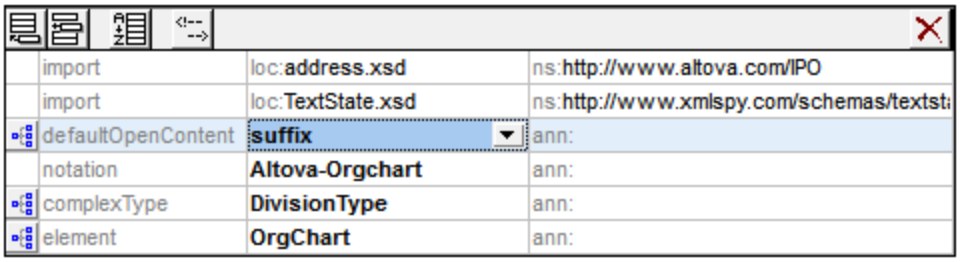
Die Wildcard-Eigenschaften werden in der Details-Eingabehilfe der Wildcard definiert. Wählen Sie die Wildcard im Diagramm aus und geben Sie die Eigenschaftswerte in die Eingabehilfe "Details" ein.

Default Open Content

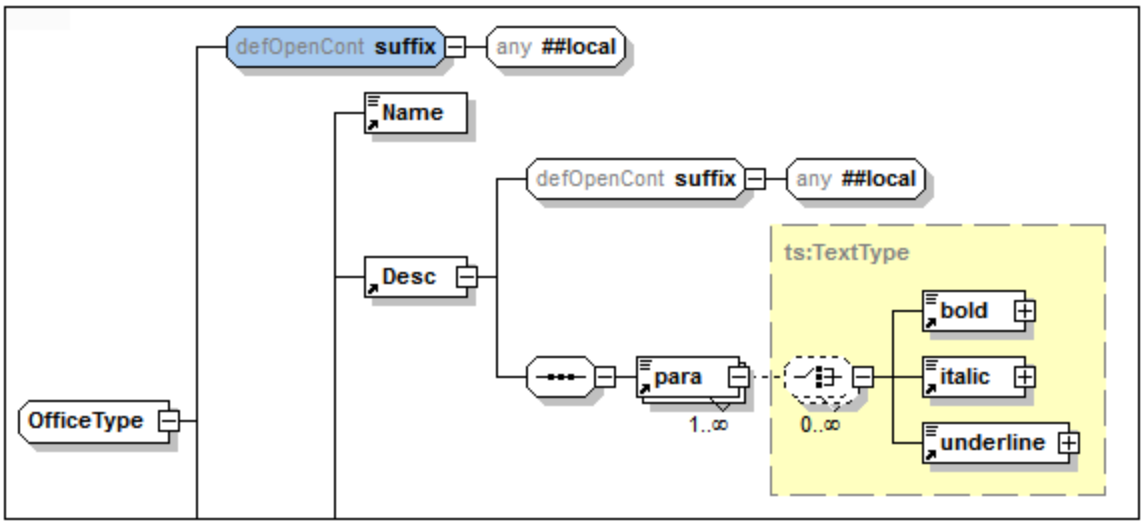
Das Element `defaultOpenContent` ist neu in XSD 1.1 und definiert, dass ein oder mehrere nicht definierte Elemente zu jedem beliebigen complexTyp mit "mixed" oder "element-only" Content hinzugefügt werden können. Es ähnelt dem Element `openContent` (welches ebenfalls neu in XSD 1.1 ist), mit dem wichtigen Unterschied, dass das Element `openContent` für einen einzigen complexType gilt, während das Element `defaultOpenContent` auf alle complexTypes im Schema angewendet wird.

Das Element `defaultOpenContent` ist eine globale Komponente 182 und kommt nur einmal im Dokument vor (*siehe Abbildung unten*) und zwar nach Includes, Imports, Redefines und Overrides und vor der Definition von Komponenten. Es hat ein `mode` Attribut, das entweder den Wert `interleave` oder `suffix` haben kann. Der Standardwert ist `interleave`.

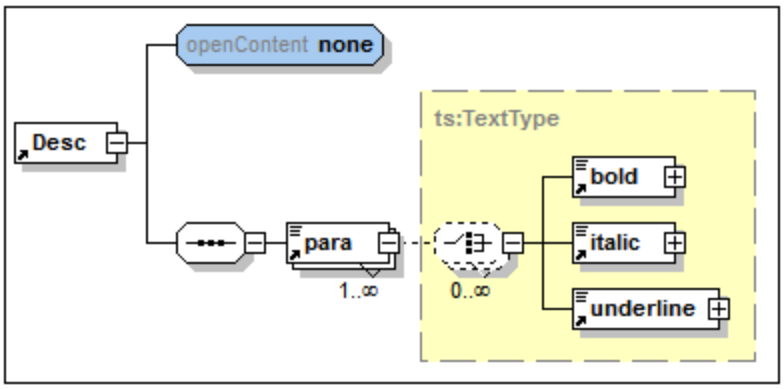
© 2016-2022 Altova GmbH



Das Element `defaultOpenContent` hat ein Content Model, das Sie in der Content Model-Ansicht auf dieselbe Art wie das openContent-Element bearbeiten können (*siehe oben*). Sobald das Element `defaultOpenContent` deklariert worden ist, wird es automatisch auf alle complexTypes im Schema angewendet und in ihren Content Models angezeigt. In der Abbildung unten sehen Sie, dass das Element `defaultOpenContent` automatisch auf die complexTypes `OfficeType` und `Desc` angewendet wurde.

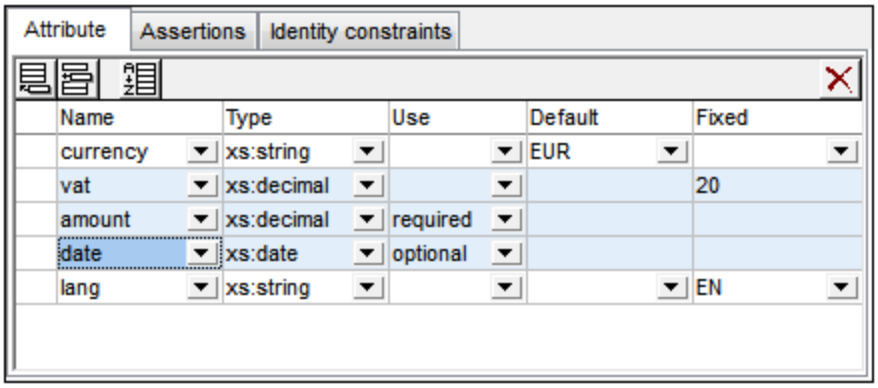


Um das Element `defaultOpenContent` für einen bestimmten complexType außer Kraft zu setzen, fügen Sie ein `openContent` Child-Element zu diesem complexType hinzu. In der Abbildung unten wurde zum Element `Desc` mit dem Element `defaultOpenContent` (*siehe Abbildung oben*) ein `openContent` Element hinzugefügt, das das Element `defaultOpenContent` außer Kraft setzt.



4.4.4 Attribute, Assertions und Identity Constraints

Der AAIDC-Bereich (Attribute/Assertions/Identity Constraints) (*Abbildung unten*) befindet sich in der Schema-Übersicht und der Content Model-Ansicht unterhalb des Hauptbereichs. Der Bereich und seine Register sind fix, doch in der Content Model-Ansicht kann jedes Register einzeln ein- und ausgeblendet werden, sodass die Komponenten des Registers im Diagramm in der Content Model-Ansicht anstatt im AAIDC-Bereich angezeigt und bearbeitet werden können. Wenn alle drei Register im Diagramm angezeigt werden, wird der AAIDC-Bereich nicht mehr angezeigt.



Sie können entweder über das Dialogfeld Schema-Ansicht konfigurieren[1395] (**Schema-Design | Ansicht konfigurieren | Register "Element"**) oder durch Klicken auf die entsprechende Schaltfläche in der Symbolleiste "Schema-Design" (*siehe unten*) zwischen dem AIDC-Bereich und dem Diagramm wechseln.

	Attribute im Diagramm anzeigen: Ist in der Content Model-Ansicht aktiv. Wechselt zwischen der Anzeige der Attribute im Diagramm (aktiviert) und auf dem Register "Attribute".
	Assertions im Diagramm anzeigen: Ist in der Content Model-Ansicht aktiv. Wechselt zwischen der Anzeige der Assertions im Diagramm (aktiviert) und auf dem Register "Assertions".
	Constraints im Diagramm anzeigen: Ist in der Content Model-Ansicht aktiv. Wechselt zwischen der Anzeige der Identity Constraints im Diagramm (aktiviert) und auf dem Register "Identity Constraints".

Verwendung der Register

Je nachdem was für eine Komponente im oberen Bereich der Schema-Übersicht oder der Content Model-Ansicht ausgewählt ist, wird das entsprechende Register im AAIDC-Bereich aktiv. Da Sie z.B. ein Attribut zu einem complexType hinzufügen können, ist das Register "Attribute" aktiv, wenn im Hauptfenster ein complexType ausgewählt ist. (Ein Register gilt dann als aktiv, wenn die Befehle dazu aktiv sind).

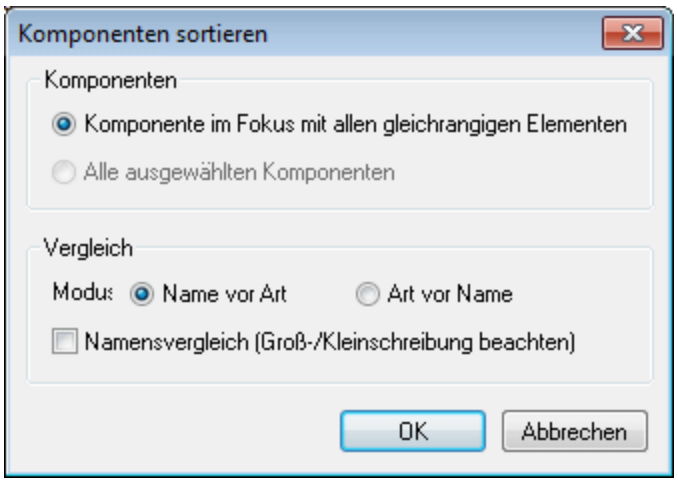
Eine Anleitung zur Verwendung der beiden Modi finden Sie in den Unterabschnitten dieses Abschnitts.

- Attribute, Attributgruppen, Attribut-Wildcards[213]
- Assertions[217]
- Identity Constraints[220]

© 2016-2022 Altova GmbH

Sortieren von Attributen und Identity Constraints

Durch Klicken auf die Schaltfläche **Sortieren** in der Symbolleiste des jeweiligen Registers können Sie die Attribute und Identity Constraints sortieren. Im Dialogfeld "Komponenten sortieren", das daraufhin angezeigt wird (*Abbildung unten*) können Sie auswählen, ob entweder die einzelne ausgewählte Komponente und ihre gleichrangigen Komponenten oder die Gruppe der ausgewählten Komponenten sortiert werden soll. So wurden z.B. in der Abbildung oben drei Attribute ausgewählt (*blau markiert*). Mit Hilfe von **Umschalt** + Klick können Sie einen Bereich auswählen, mit **Strg** + Klick können Sie zusätzliche Komponenten zur Auswahl hinzufügen.



Nach Auswahl der Komponentengruppe können Sie auswählen, ob diese zuerst alphabetisch nach Namen und dann nach Art (*Name vor Art*) oder umgekehrt (*Art vor Name*) sortiert werden sollen. Die Sortierreihenfolge wird sofort im Text des Schema-Dokuments implementiert; dies ist nicht nur eine Maske der Benutzeroberfläche.

4.4.4.1 Attribute, Attributgruppen, Attribut-Wildcards

Auf dem Register "Attribute" des AAIDC (Attribute/Assertions/Identity Constraints)-Bereichs (*Abbildung unten*) können Sie:

- Attribute zum ausgewählten complexType lokal deklarieren 214
- Attributgruppen für die Verwendung im ausgewählten complexType referenzieren 215
- Attribut-Wildcars im ausgewählten complexType definieren 215

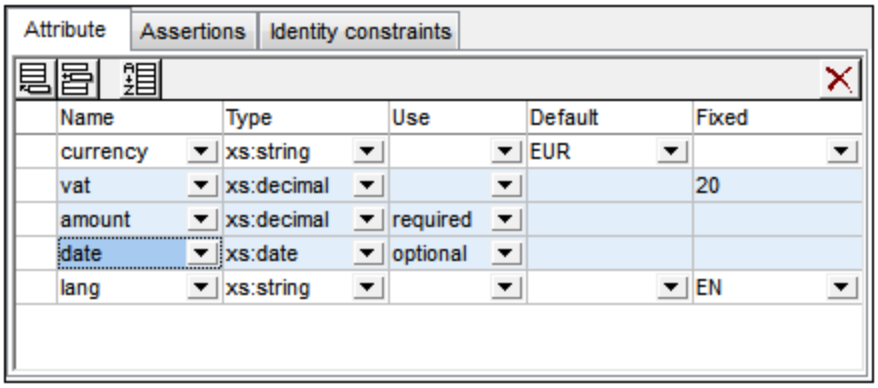
Anmerkung: Wenn Sie die Ansicht so konfiguriert haben, dass die Attribute im Diagramm und nicht im AAIDC-Bereich angezeigt werden (**Schema-Design | Ansicht konfigurieren**), können Sie die Eigenschaften der Attribute, der Attributgruppenreferenzen und Attribut-Wildcards im Diagramm und in der Eingabehilfe "Details" bearbeiten.



Attribute im Diagramm anzeigen: Ist in der Content Model-Ansicht aktiv. Wechselt zwischen der Anzeige der Attribute im Diagramm (Schaltfläche aktiv) und auf dem Register "Attribute".

Attribute

Auf dem Register "Attribute" des AAIDC (Attribute/Assertions/Identity Constraints)-Bereichs (*Abbildung unten*) können Sie lokale Attribute von Elementen und complexTypes sowie die Attribute, aus denen die Attributgruppen bestehen, deklarieren.



Gehen Sie folgendermaßen vor, um Attribute zu erstellen:

1. Wählen Sie in der Schema-Übersicht den complexType bzw. die Attributgruppe aus, für die Sie das Attribut erstellen möchten.
2. Klicken Sie auf dem Register "Attribute" auf das Symbol **Anhängen** oder **Einfügen** links oben und wählen Sie die Option **Attribut** aus..
3. Geben Sie in der Zeile, die für das Attribut erstellt wurde, die Einzelheiten zum Attribut ein (Name, Typ, use (Verwendung), default (Standardwert) oder fixed (fixer Wert)). Die Eigenschaft `name` ist obligatorisch und der Standardwert von `use` ist `optional`. Der Datentyp und die Werteigenschaften "default/fixed" sind optional.

Anmerkung: Attribute können zu Attributgruppen nur in der Schema-Übersicht[171], zu complexTypes aber sowohl in der Schema-Übersicht[171] als auch der Content Model-Ansicht[171] hinzugefügt werden.

Anmerkung: Wenn ein Attribut eine Typdefinition hat, gelangen Sie bei Aufruf des Befehls **Gehe zu Typdefinition** zur Typdefinition. Bei simpleTypes wird ein Meldungsfeld mit Informationen über den simpleType angezeigt.

Standardwerte und fixe Werte

Ein Standardwert oder fixer Wert wird in einer Attributdeklaration definiert und auf das Instanzdokument angewendet, wenn dieses Attribut im Instanzdokument fehlt. Es kann entweder ein Standardwert oder ein fixer Wert definiert werden, nicht aber beides (*siehe Abbildung unten*). Der Standardwert bzw. fixe Wert muss entsprechend dem Datentyp des Attributs gültig sein. Wenn `use` auf `required` gesetzt ist, ist weder ein Standardwert noch ein fixer Wert zulässig.

Beachten Sie bitte Folgendes:

- *Standardwerte:* Ein Standardwert wird nur dann eingefügt, wenn das Attribut fehlt. Wenn das Attribut vorhanden ist und einen gültigen Wert hat, wird der Standardwert nicht eingefügt. Wenn der Wert im Instanzdokument ungültig ist, wird ein Fehler ausgegeben.

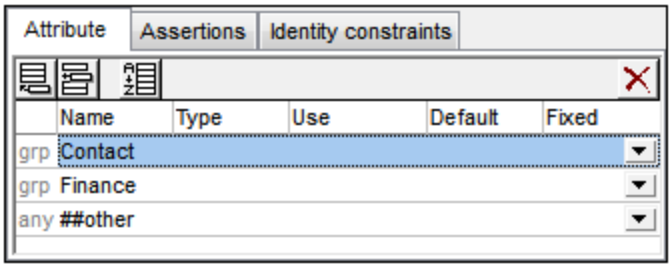
© 2016-2022 Altova GmbH

- *Fixe Werte:* Ein fixer Wert wird nicht nur dann angewendet, wenn das Attribut fehlt, sondern auch, wenn der Wert im Instanzdokument nicht mit dem in der Attributdeklaration definierten fixen Wert übereinstimmt.

Anmerkung: Standardwerte und fixe Werte können sowohl für lokale als auch globale Attribute definiert werden. Bei lokalen Attributen können sie sowohl auf dem Register "Attribute" des AAIDC-Bereichs (*Abbildung oben*) als auch in der Eingabehilfe "Details" definiert werden. Bei globalen Attributen können sie in der Eingabehilfe "Details" definiert werden.

Attributgruppenreferenzen

Wenn eine globale Attributgruppe deklariert wurde, können Sie in der Definition eines complexType eine Referenz auf diese Attributgruppe hinzufügen. Wählen Sie dazu die complexType-Komponente in der Schema-Übersicht oder der Content Model-Ansicht aus und klicken Sie anschließend auf das Symbol **Anhängen** oder **Einfügen** links oben auf dem Register "Attribute" des AAIDC-Bereichs und wählen Sie **Attribute Group** aus. Geben Sie in der Attributgruppenzeile, die erstellt wurde, den Namen der zu referenzierenden Attributgruppe ein (*siehe Abbildung unten, in der Sie zwei Attributgruppenreferenzen sehen*). Sie können mehrere Attributgruppen hinzufügen.

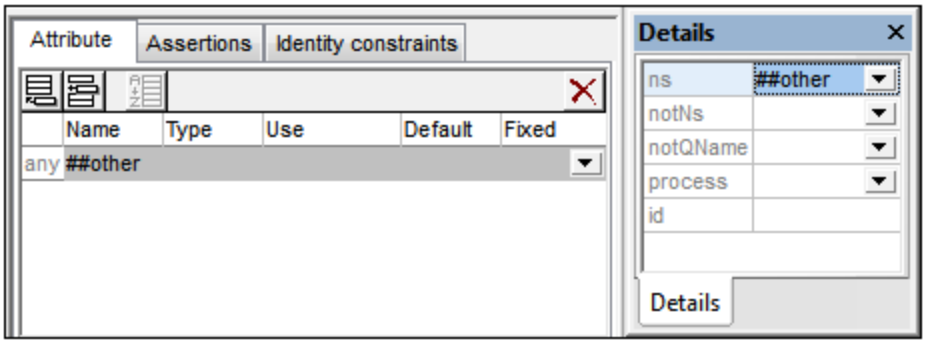


Wenn die Attributgruppe auf dem Register "Attribute" ausgewählt ist, können Sie ihre Eigenschaften auch in der Eingabehilfe "Details" bearbeiten.

Attribut-Wildcards: anyAttribute

Eine Attribut-Wildcard kann zu einem complexType hinzugefügt werden, damit in einem Element jedes beliebige Attribut verwendet werden kann. Eine Attribut-Wildcard wird mit Hilfe eines einzigen `anyAttribute` Elements definiert. Dieses Element ermöglicht die Verwendung beliebig vieler Attribute aus dem definierten Namespace im Element des Instanzdokuments.

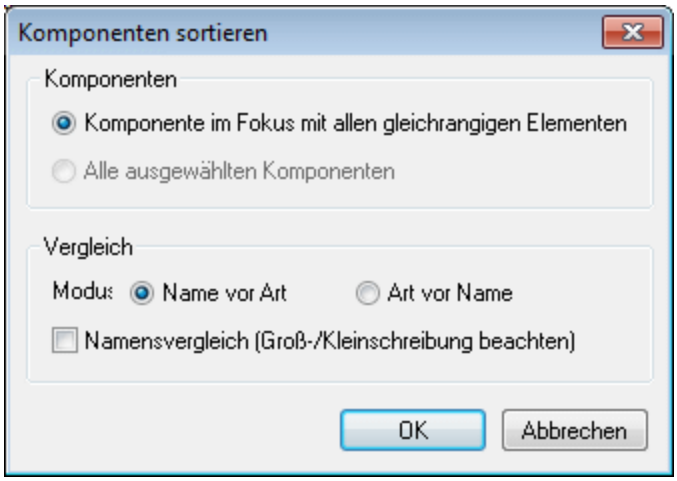
Um eine Attribut-Wildcard hinzuzufügen, wählen Sie die complexType-Komponente in der Schema-Übersicht oder der Content Model-Ansicht aus, klicken Sie auf das Symbol **Anhängen** oder **Einfügen** links oben auf dem Register "Attribute" des AAIDC-Bereichs und wählen Sie den Eintrag **Any Attribute** aus. Daraufhin wird eine Zeile für die Attribut-Wildcard `anyAttribute` erstellt (*siehe Abbildung unten*).



Auf dem Register "Attribute" können Sie die `namespace` Eigenschaft der Attribut-Wildcard `anyAttribute` definieren. Wenn die Attribut-Wildcard auf dem Register "Attribute" ausgewählt ist, können Sie in der Eingabehilfe "Details" (*siehe Abbildung oben*) zusätzliche Eigenschaften definieren. Beachten Sie, dass die Eigenschaften `notNamespace` und `notQName` XSD 1.1 Features[173] sind und im XSD 1.0 Modus[173] nicht verfügbar sind.

Sortieren von Attributen und Attributgruppen

Sie können die Attribute und Attributgruppen auf dem Register "Attribute" sortieren. Klicken Sie dazu auf das **Sortieren**-Symbol in der Symbolleiste des Registers. Im Dialogfeld "Komponenten sortieren" (*Abbildung unten*), das daraufhin angezeigt wird, können Sie auswählen, ob entweder die ausgewählte Komponente und alle ihre gleichrangigen Elemente oder die gesamte Gruppe der ausgewählten Komponenten sortiert werden soll. Mit Hilfe von **Umschalt**+Klick können Sie einen Bereich auswählen, mit Hilfe von **Strg**+Klick können Sie zusätzliche Komponenten zur Auswahl hinzufügen.



Nachdem Sie den Bereich ausgewählt haben, können Sie wählen, ob der gesamte Bereich der Attribute und Attributgruppen alphabetisch sortiert werden solll (*Name vor Art*) oder ob zuerst die Attribute alphabetisch sortiert werden sollen und danach erst die Attributgruppen.

Die Sortierreihenfolge wird sofort im Text des Schema-Dokuments implementiert; dies ist nicht nur eine Maske der Benutzeroberfläche.

Anmerkung: Attribut-Wildcards werden im Sortierbereich nicht berücksichtigt, da sie immer am Ende einer complexType-Deklaration stehen müssen und nur eine Attribut-Wildcard je complexType-Deklaration zulässig ist.

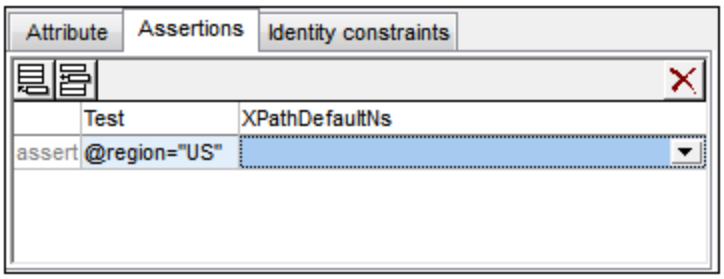
 © 2016-2022 Altova GmbH

4.4.4.2 Assertions

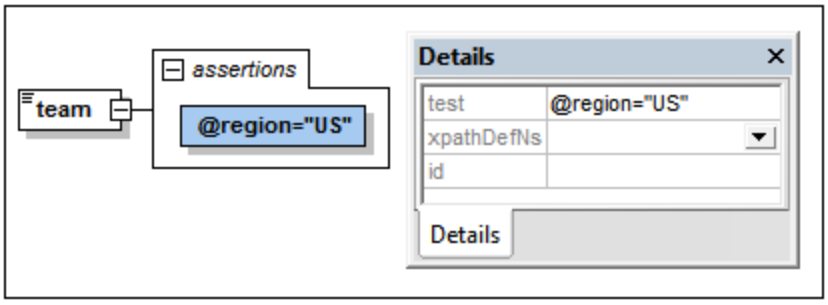
Die in diesem Abschnitt beschriebenen Assertions (Behauptungen) sind **Assertions zu complexTypes**. Eine solche Assertion wird in einem `xs:assert` Element definiert und dient als Gültigkeits-Constraint für den complexType. (Bei der anderen Art von Assertion handelt es sich um eine Assertion zu einem simpleType. Sie wird in einem `xs:assertion` Element definiert und in der Eingabehilfe "Facets" [237] eines simpleType erstellt und bearbeitet.)

ComplexType Assertions können in der Schema-Ansicht über folgende GUI-Bereiche erstellt und bearbeitet werden:

- *In der Schema-Übersicht:* im AAIDC-Bereich (Attribute/Assertions/Identity Constraints) auf dem Register "Assertions" (*Abbildung unten*).



- *In der Content Model-Ansicht:* Assertions können auf dem Register "Assertions" (Abbildung oben) oder im Diagramm (*Abbildung unten*) bearbeitet werden. Um die Anzeige im Diagramm zu wählen, klicken Sie in der Symbolleiste "Schema-Design" auf die Schaltfläche **Assertions im Diagram anzeigen.** Wählen Sie im Diagramm das Kästchen "Assertion" des complexType oder complex-content Elements aus und geben Sie anschließend die Definition der Assertion in die Eingabehilfe "Details" ein oder bearbeiten Sie diese.



Assertions im Diagramm anzeigen: Ist in der Content Model-Ansicht aktiv. Dient zum Umschalten zwischen der Anzeige von Assertions im Diagramm (Schaltfläche aktiviert) und auf dem Register "Assertions".

Anmerkung: Assertions sind ein XSD 1.1-Feature. Daher stehen die Funktionen zur Bearbeitung von Assertions nur im XSD 1.1-Modus [173] zur Verfügung.

Geltungsbereich der Assertion

Der zur Definition des Assertion Constraint verwendete XPath-Ausdruck muss sich im Geltungsbereich des complexType, zu dem er definiert ist, befinden. Wenn der XPath-Ausdruck also auf einen bestimmten Node zugreifen soll, muss die Assertion für einen Vorfahren dieses Node definiert werden.

Hinzufügen und Löschen von Assertions

Ein complexType kann mehrere Assertions haben. Damit das Element im Instanzdokument gültig ist, muss das Ergebnis der XPath-Auswertung für jede Assertion `true` sein. Um eine Assertion zu einem complexType hinzuzufügen, gehen Sie folgendermaßen vor:

- *In der Schema-Übersicht:* Wählen Sie den complexType aus. Klicken Sie anschließend auf dem Register "Assertions" des AAIDC-Bereichs (*siehe Abbildung oben*) auf das **Anhängen-** oder **Einfügen-**Symbol links oben auf dem Register. Sie können mehrere Assertions hinzufügen. Um eine Assertion zu löschen, wählen Sie diese aus und klicken Sie auf das **Löschen**-Symbol rechts oben auf dem Register.
- *In der Content Model-Ansicht (siehe Abbildung oben):* Klicken Sie mit der rechten Maustaste auf den complexType und wählen Sie den Befehl **Child hinzufügen | Assertion**. Alternativ dazu können Sie auch mit der rechten Maustaste auf eine vorhandene Assertion im Diagramm klicken und den Befehl **Anhängen | Assertion** oder **Einfügen | Assertion** wählen. Sie können mehrere Assertions zu einem complexType hinzufügen. Um eine Assertion zu löschen, wählen Sie diese aus und betätigen Sie die **Entf**-Taste.

Definieren des XPath-Ausdrucks der Assertion

Der XPath-Ausdruck einer complexType Assertion definiert den Validierungs-Constraint, der im Instanzdokument am complexType-Element angewendet werden soll. So wurde die Assertion in der Abbildung oben z.B. für das complex-type Element `team` definiert. Der XPath-Ausdruck der Assertion lautet: `@region="US"`. Im XML-Schema-Dokument wird die Assertion folgendermaßen definiert:

```
<xs:assert test='@region="US"'/>
```

Die Assertion legt fest, dass das Element `team` im Instanzdokument ein Attribut `region` mit dem Wert `US` haben muss. Andernfalls ist das Dokument ungültig.

Beachten Sie die folgenden Punkte:

- XPath-Ausdrücke müssen in XPath 2.0 geschrieben werden.
- Die im XPath-Ausdruck ausgewerteten Nodes müssen sich im Geltungsbereich der Assertion befinden (*siehe oben*).
- Wenn das Ergebnis eines Ausdruck nicht der Boolesche Wert `true/false` ist, so wird der zurückgegebene Wert in einen Booleschen Wert umgewandelt. Eine nicht leere Sequenz wird in `true` umgewandelt, während eine leere Sequenz in `false` konvertiert wird.
- Wenn der Ausdruck Syntaxfehler enthält, wird er rot angezeigt. Kontextfehler werden nicht markiert. Wenn der XPath-Ausdruck z.B. ein Attribut testet und dieses Attribut im Schema nicht definiert ist, so wird kein Fehler angezeigt.

Die Assertion-Meldung

Oft ist es nützlich, wenn z.B. mit der Definition einer Assertion auch eine Erläuterung dazu verfügbar ist, damit eine entsprechende Meldung angezeigt werden kann, wenn die Assertion bei der Validierung des XML-

© 2016-2022 Altova GmbH

Instanzdokuments nicht erfüllt wird. Da in der XML Schema-Spezifikation keine derartige Meldung vorgesehen ist, können Sie in XMLSpy im Altova xml-schema-extensions Namespace `http://www.altova.com/de/xml-schema-extensions` (oder in jedem anderen Namespace) zusammen mit der Definition der Assertion eine Meldung definieren, die bei der Validierung des XML-Instanzdokuments verwendet werden soll. Zum Beispiel:

```
<xs:assert test="count(//MyNode) ge 1" altova:message="Es muss mindestens ein MyNode-Element vorhanden sein"/> oder
<xs:assertion test="count(//MyNode) ge 1" altova:message="Es muss mindestens ein MyNode-Element vorhanden sein"/>
```

Wenn die in der Assertion definierte Einschränkung nicht erfüllt wird, zeigt der Validierungsprozessor von XMLSpy zusammen mit der Validierungsfehlermeldung die mit der Assertion verknüpfte Erklärung als Meldung an. Der Validierungsprozessor gibt den Wert eines `assert/@message` Attributs oder eines `assertion/@message` Attributs unabhängig davon aus, in welchem Namespace sich das `message` Attribut befindet. Sie können in der Schema-Ansicht jedoch nur `message` Attribute bearbeiten, die sich im Altova xml-schema-extension Namespace befinden. Um `message` Attribute in anderen Namespaces zu bearbeiten, verwenden Sie die Textansicht.

Nähere Informationen dazu finden Sie unter Assertion-Meldungen 241.

Verwendung von xpathDefaultNamespace

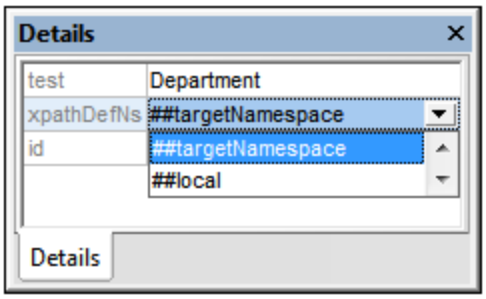
Ein im XML-Schema-Dokument deklarierter Standard-Namespace ist der Standard-Namespace des XML-Schema-Dokuments. Er gilt für Elementnamen ohne Präfix im Schema-Dokument, nicht aber für Elementnamen ohne Präfix in XPath-Ausdrücken im Schema-Dokument.

Mit dem Attribut `xpathDefaultNamespace` (einem XSD 1.1-Feature) wird der Namespace definiert, dem Elementnamen ohne Präfix im XPath-Ausdruck angehören.

Der Geltungsbereich von XPath-Standard-Namespaces erstreckt sich auf die XML-Schema-Elemente, für die diese Namespaces deklariert wurden. Das Attribut `xpathDefaultNamespace` kann in den folgenden XML Schema 1.1-Elementen vorkommen:

- `xs:schema`
- `xs:assert` and `xs:assertion`
- `xs:alternative`
- `xs:selector` and `xs:field` (in Identity Constraints)

Das `xpathDefaultNamespace` Attribut für `xs:schema` wird im Dialogfeld "Schema-Einstellungen" (**Schema-Design | Schema-Einstellungen**) im XSD 1.1-Modus definiert. Für die anderen oben aufgelisteten Elemente wird es in der jeweiligen Details-Eingabehilfe (*Beispiel in der Abbildung unten*) definiert.



Durch Deklarieren des XPath-Standard-Namespace für `xs:schema` wird der XPath-Standard-Namespace für den Geltungsbereich des gesamten Schemas deklariert. Sie können diese Deklaration für Elemente, in denen das Attribut `xpathDefaultNamespace` zulässig ist (*siehe Liste oben*), außer Kraft setzen.

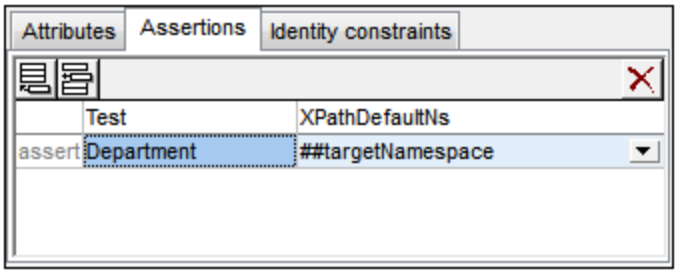
Anstelle eines tatsächlichen Namespace kann das Attribut `xpathDefaultNamespace` eines von drei Schlüsselwörtern enthalten:

- `##targetNamespace`: Der XPath-Standard-Namespace ist mit dem Target Namespace des Schemas identisch.
- `##defaultNamespace`: Der XPath-Standard-Namespace ist mit dem Standard-Namespace des Schemas identisch.
- `##local`: Es gibt keinen XPath-Standard-Namespace.

Wenn im Dokument kein XPath-Standard-Namespace deklariert ist, haben Elemente ohne Präfix in XPath-Ausdrücken keinen Namespace.

Anmerkung: Die XPath-Standard-Namespace-Deklaration gilt nicht für Attribute.

Sie können den XPath-Standard-Namespace für XPath-Ausdrücke in Assertions auch in der Definition der Assertion festlegen. Wählen Sie im AAIDC-Bereich (Abbildung unten) auf dem Register "Assertions" das gewünschte Schlüsselwort aus der Dropdown-Liste des Felds *XPathDefaultNS* aus.



Der ausgewählte Namespace befindet sich im Geltungsbereich der Assertion.

4.4.4.3 Identity Constraints

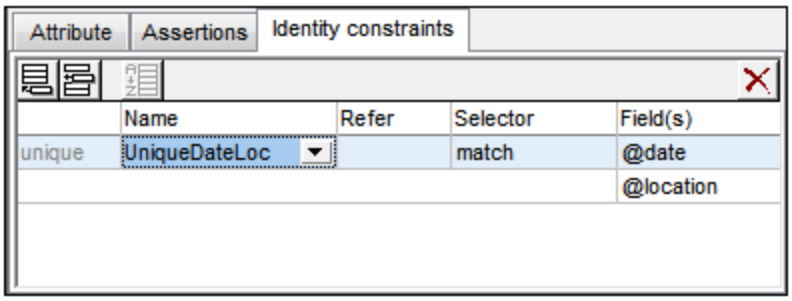
Identity Constraints (IDCs) können für globale oder lokale Element-Deklarationen definiert werden. Sie definieren die Eindeutigkeit von Nodes und ermöglichen eine korrekte Referenzierung zwischen eindeutigen Nodes.

 © 2016-2022 Altova GmbH

Methoden für die Deklaration

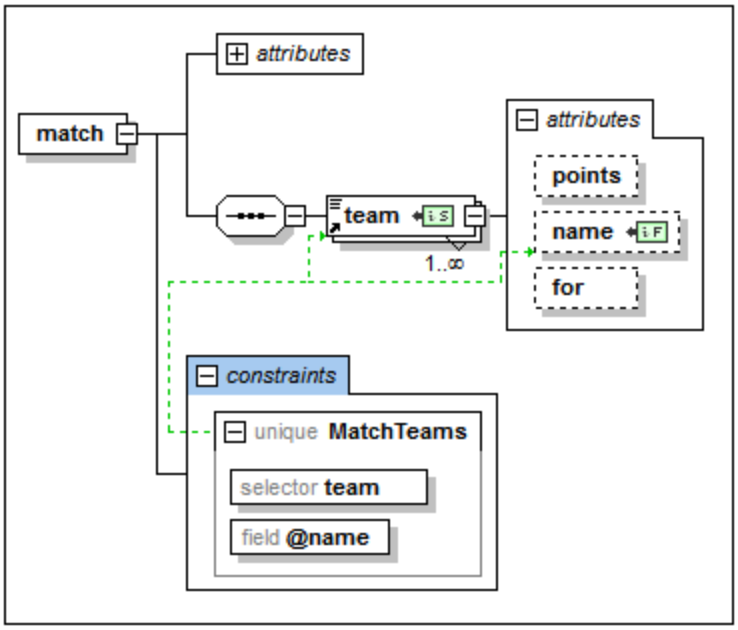
Zur Definition eines IDC (`unique`, `key`, `keyref`) stehen die folgenden Methoden zur Verfügung:

- In der Schema-Übersicht[177] können IDCs für globale Elemente deklariert werden. Wählen Sie ein globales Element aus und definieren Sie die IDCs auf dem Register "Identity Constraints" des AAIDC-Bereichs (Attribute/Assertions/Identity-Constraints) (*Abbildung unten*).



  Über das **Einfügen-** oder **Anhängen-**Symbol des Registers "Identity Constraints" können Sie einen IDC (`unique`, `key`, `keyref`) hinzufügen. Mit Hilfe dieser Symbole können Sie auch ein Feld (`field`) zum ausgewählten IDC hinzufügen. Über die Schaltfläche **Löschen** können Sie das ausgewählte Feld oder den ausgewählten IDC löschen.

- In der Content Model-Ansicht[190] eines globalen Elements können IDCs für das globale Elemente oder einen lokalen Nachfahren davon definiert werden. In dieser Ansicht können IDCs entweder auf dem Register "Identity Constraints" (*Abbildung oben*) oder im *Constraints*-Kästchen eines Elements im Diagramm bearbeitet werden (*Abbildung unten*, in dem das Element `match` einen "unique"-Constraint hat, der einen Selektor `team` hat). Zweiteres kann im Dialogfeld "Schema-Ansicht konfigurieren" (**Schema-Design | Ansicht konfigurieren**) ausgewählt werden. Alternativ dazu können Sie in der Schema-Design-Symbolleiste auf die Schaltfläche **Constraints im Diagramm anzeigen** klicken, um die IDCs ein- bzw. auszublenden. Im Diagramm sehen Sie eine grafische Darstellung der IDCs, die mit Drag-and-Drop bearbeitet werden können.



Um einen IDC (`unique`, `key`, `keyref`) im Diagramm hinzuzufügen, wenn der Diagramm-Modus für IDCs aktiviert ist, klicken Sie mit der rechten Maustaste auf das gewünschte Element und wählen Sie im Kontextmenü den Befehl **Child hinzufügen | [IDC]**. Das Element `field` ist im Kontextmenü nur dann verfügbar, wenn im Diagramm ein IDC ausgewählt ist. Drücken Sie die **Entf**-Taste, um das ausgewählt Feld (`field`) bzw. den ausgewählten IDC zu löschen.

Der XPath-Ausdruck kann auf drei verschiedene Arten in die Kästchen `selector` und `field` eingegeben werden: (i) durch manuelle Eingabe, (ii) durch Auswahl des gewünschten Node aus einer Dropdown-Liste, die automatisch angezeigt wird, wenn Sie in das `selector`- oder `field`-Kästchen klicken oder (iii) durch Ziehen des Target Node in das `selector`- oder `field`-Kästchen und Loslassen der Maustaste, wenn das Kästchen markiert erscheint; der XPath-Ausdruck wird automatisch erstellt.

Anmerkung: Zusätzlich dazu steht in der Eingabehilfe "Komponenten" auf dem Register "Identity Constraints" eine Übersicht über alle Identity Constraints[229] im Schema zur Verfügung.

Identity Constraint-Schaltflächen

Constraints im Diagramm anzeigen: Ist in der Content Model-Ansicht aktiviert. Damit wechseln Sie zwischen der Anzeige der IDCs im Diagramm (Schaltfläche aktiviert) und der Anzeige der IDCs auf dem Register "Identity Constraints".

Identity Constraints visualisieren: Ist in der Content Model-Ansicht aktiviert. Damit können Sie die IDC-Informationen ein- und ausblenden.

Selector Node, Field Node: Mit diesen beiden in Node-Kästchen des Diagramms angezeigten Symbolen wird der (in IDCs) vom XPath-Ausdruck für `selector` bzw. `field` ausgewählte Node gekennzeichnet.

© 2016-2022 Altova GmbH

Visualisieren von IDCs

Wenn die Schaltfläche "Identity Constraints visualisieren" aktiviert ist, werden die IDC-Informationen im Diagramm angezeigt. Wenn die Visualisierung aktiviert ist, werden Nodes, die von `selector`- und `field`-XPath-Ausdrücken ausgewählt weden, durch Symbole in den entsprechenden Kästchen gekennzeichnet (*siehe Schaltflächen-Abschnitt oben*) und das IDC-Kästchen durch grüne Linien mit seinen selector- und field-Kästchen verbunden (*siehe Abbildung oben*).

Mit der Schaltfläche "ID Constraints visualisieren" wird auch die IDC-Validierungsfunktion in der Schema-Ansicht aktiviert. Wenn ein XPath-Ausdruck einen Fehler aufweist oder ein IDC fehlerhaft ist, werden Fehler durch roten und Warnungen durch orangen Text gekennzeichnet. Wenn Sie das XML-Schema-Dokument validieren, werden die Fehler- und Warnmeldungen im Fenster "Meldungen" angezeigt.

Sie können die Validierung auch deaktivieren, indem Sie die Schaltfläche "ID Constraints visualisieren" deaktivieren.

XML-Code

Die IDC-Beispiele weiter unten in diesem Abschnitt basieren auf dem folgenden gültigen Instanzdokument.

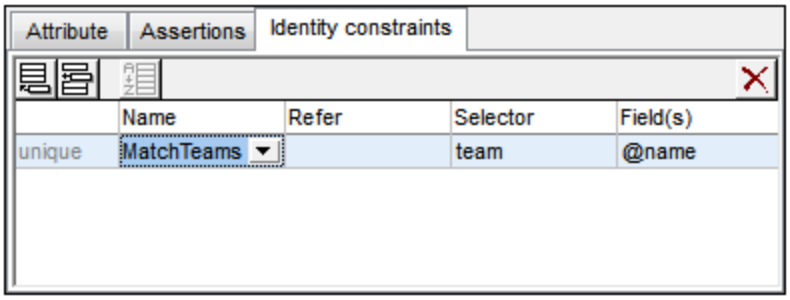
```
<results xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:noNamespaceSchemaLocation="Scores.xsd">
   <!----- Groups ----->
   <group id="A">
          <team name="Brazil"/>
          <team name="Germany"/>
          <team name="Italy"/>
          <team name="Holland"/>
   </group>
   <group id="B">
          <team name="Argentina"/>
          <team name="France"/>
          <team name="England"/>
          <team name="Spain"/>
   </group>
   <!----- Matches ----->
   <match group="A" date="2012-06-12" location="Munich">
          <team name="Brazil" for="2" points="3"/>
          <team name="Germany" for="1" points="0"/>
   </match>
   <match group="A" date="2012-06-12" location="Frankfurt">
          <team name="Italy" for="2" points="1"/>
          <team name="Holland" for="2" points="1"/>
   </match>
   <match group="B" date="2012-06-13" location="Munich">
          <team name="Argentina" for="2" points="3"/>
          <team name="France" for="0" points="0"/>
   </match>
   <match group="B" date="2012-06-13" location="Berlin">
          <team name="England" for="0" points="1"/>
          <team name="Spain" for="0" points="1"/>
```

```
    </match>
</results>
```

Unique Constraints (unique)

Ein Unique Constraint gibt an, dass der Wert eines Elements oder Attributs (oder einer Gruppe von Elementen und/oder Attributen) innerhalb eines definierten Geltungsbereichs eindeutig sein muss. Im oben gezeigten XML-Fragment soll gewährleistet werden, dass es sich bei den beiden Mannschaften, die in einem Match gegeneinander antreten, nicht um dieselbe Mannschaft handelt. Daher werden die Werte des Nodes `team/@name` im Geltungsbereich jedes `match` Elements als unique (eindeutig) definiert. Dies geschieht folgendermaßen:

1. Wählen Sie in der Schema-Übersicht das Element `match` aus. Das Element `match` ist somit der Geltungsbereich der Identity Constraint-Definition.
2. Klicken Sie auf dem Register "Identity Constraints" in der linken oberen Ecke des Registers auf das Symbol **Anhängen** oder **Einfügen** und wählen Sie im Menü, das daraufhin angezeigt wird, den Eintrag **Unique**. Daraufin wird eine Zeile für den Unique Constraint eingefügt (*siehe Abbildung unten*).



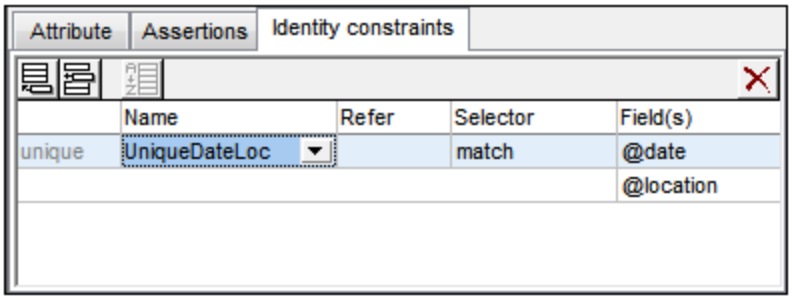
3. Geben Sie dem Identity Constraint einen Namen. (In der Abbildung oben ist der Name `MatchTeams`.)
4. Geben Sie in der Spalte *Selector* einen XPath-Ausdruck zur Auswahl des Elements `team` ein. Beachten Sie, dass das Element `match` der Kontext-Node ist. Das Element `team` ist nun der Selektor des IDC, also der Node, auf den der Unique Constraint angewendet wird.
5. Geben Sie in der Spalte *Field* diden Node `@name` ein, der eindeutig sein muss. Der Wert dieses Node ist der Wert, der eindeutig sein muss.

Mit dem oben beschriebenen Unique Constraint wird definiert, dass jedes `team`-Element im Geltungsbereich jedes `match`-Elements einen eindeutigen `@name`-Attributwert haben muss.

Sie können die Eindeutigkeit anhand zusätzlicher Felder überprüfen. So kann z.B. ein Unique Constraint für das Element `results` definiert werden, um sicherzustellen, dass alle Spiele eine eindeutige Kombination von Datum und Ort aufweisen: An ein und demselben Ort darf am selben Datum nicht mehr als ein Spiel stattfinden. Beim Unique Constraint muss die Kombination der Werte von `@date` und `@location` für jedes `match` Element (den Selektor) im Geltungsbereich des `results`-Elements eindeutig sein.

Definieren Sie den Unique Constraint zum Element `results` auf ähnliche Weise wie oben beschrieben. Der Selektor ist `match`, die Felder (fields) sind `@date` und `@location` (*siehe Abbildung unten*). Um das zweite Feld hinzuzufügen, klicken Sie auf das **Anhängen**-Symbol und anschließend auf **Field**.

© 2016-2022 Altova GmbH

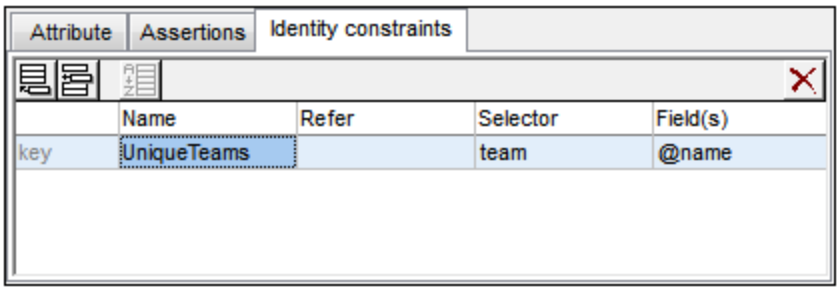


Anmerkung: Die Spalte *Refer* auf dem Register "Identity Constraints" ist nur bei `keyref`-Constraints, nicht aber bei `unique`- oder `key`-Constraints aktiv.

Key Constraints (key)

Mit einem Key Constraint wird festgelegt, (i) dass der Wert eines Elements oder Attributs (oder einer Gruppe von Elementen und/oder Attributen) innerhalb eines bestimmten Geltungsbereichs eindeutig sein muss und (ii) dass diese Feld-Elemente und/oder -Attribute im XML-Instandokument vorhanden sein müssen; optionale Elemente oder Attribute sollten daher nicht als Felder eines Key Constraint ausgewählt werden. Was den oben beschriebenen Punkt (i) betrifft, ist ein Key Constraint daher mit einem Unique Constraint identisch. Er legt jedoch einen zusätzlichen Constraint fest: seine Feld-Elemente/Attribute müssen im XML-Dokument vorhanden sein.

In der Abbildung unten sehen Sie einen Key Constraint für ein `match`-Element, der dem ersten oben beschriebenen Unique Constraint ähnelt.



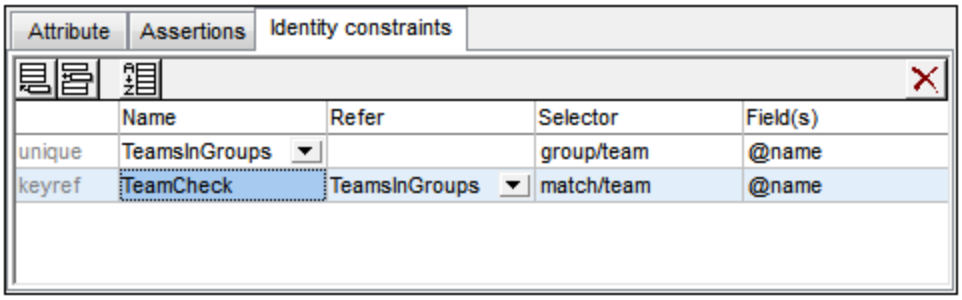
Mit diesem Key Constraint wird festgelegt, dass jedes `team`-Element im Geltungsbereich jedes `match`-Element einen eindeutigen `@name` Attributwert haben muss. Zusätzlich dazu wird festgelegt, dass das `@name`-Attribut in jedem `match/team`-Element vorhanden sein muss.

Anmerkung: Die Spalte *Refer* auf dem Register "Identity Constraints" ist nur bei `keyref`-Constraints, nicht aber bei `unique`- oder `key`-Constraints aktiv.

Key Reference Constraint (keyref)

Ein Key Reference Constraint vergleicht eine Gruppe von Werten in einem Instanzdokument mit einer anderen. In unserem XML-Fragment (*siehe oben*) können wir z.B. mit Hilfe eines Key Reference Constraint überprüfen, ob die spielenden Mannschaften sich unter den in den jeweiligen Gruppen aufgelisteten Mannschaften befinden. Falls dies nicht der Fall ist, ist das XML-Dokument ungültig.

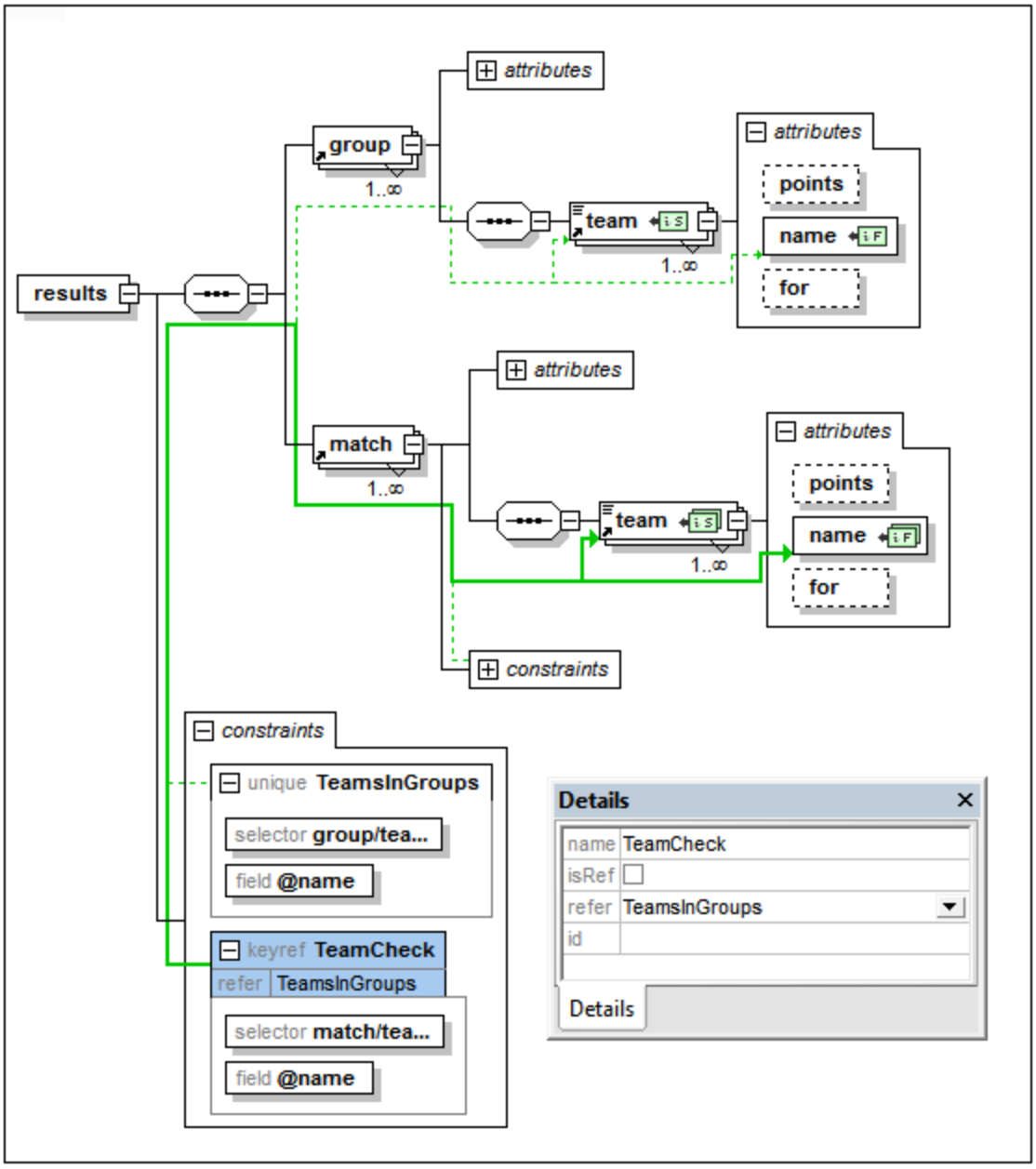
Zuerst erstellen wir eine Unique Konstante oder einen Key Constraint. In der Abbildung unten sehen Sie einen für das Element `results` erstellten Unique Constraint (`unique`) `TeamsInGroups`First. Dieser Constraint verlangt, dass jedes `team` in `group` ein eindeutiges `@name` Attribut hat.



Als nächstes erstellen wie den Key Reference Constraint (`keyref`) `TeamCheck`, der das Child `team` von `match` auswählt und überprüft, ob sein `@name`-Attributwert in den von `T` Attribute `eamsInGroups`, das es (in der Spalte *Refer*) referenziert, zurückgegebenen Werten vorhanden ist.

In der Abbildung unten sehen Sie die grafische Anzeige dieses Key Reference Constraint (blau markiert) zusammen mit der Eingabehilfe "Details" (in der Sie den referenzierten IDC ebenfalls auswählen können). Die Beziehungen des ausgewählten IDC werden mit einer durchgezogenen grünen Linie gezeigt, während nicht ausgewählte IDCs mit einer gepunkteten grünen Linie angezeigt werden. Außerdem wird für jeden Identity Constraint der vom XPath-Ausdruck für `selector` und `field` ausgewählte Node durch die Symbole bzw. markiert. Wenn ein Node reduziert ist, endet die Beziehungslinie zu diesem mit einem Auslassungszeichen.

© 2016-2022 Altova GmbH



Verwendung von xpathDefaultNamespace

Ein im XML-Schema-Dokument deklarierter Standard-Namespace ist der Standard-Namespace des XML-Schema-Dokuments. Er gilt für Elementnamen ohne Präfix im Schema-Dokument, nicht aber für Elementnamen ohne Präfix in XPath-Ausdrücken im Schema-Dokument.

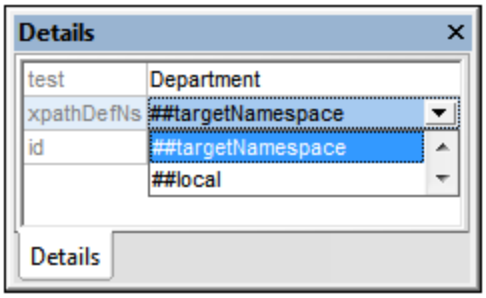
Mit dem Attribut `xpathDefaultNamespace` (einem XSD 1.1-Feature) wird der Namespace definiert, dem Elementnamen ohne Präfix im XPath-Ausdruck angehören.

Der Geltungsbereich von XPath-Standard-Namespaces erstreckt sich auf die XML-Schema-Elemente, für die diese Namespaces deklariert wurden. Das Attribut `xpathDefaultNamespace` kann in den folgenden XML

Schema 1.1-Elementen vorkommen:

- `xs:schema`
- `xs:assert` and `xs:assertion`
- `xs:alternative`
- `xs:selector` and `xs:field` (in Identity Constraints)

Das `xpathDefaultNamespace` Attribut für `xs:schema` wird im Dialogfeld "Schema-Einstellungen" (**Schema-Design | Schema-Einstellungen**) im XSD 1.1-Modus definiert. Für die anderen oben aufgelisteten Elemente wird es in der jeweiligen Details-Eingabehilfe (*Beispiel in der Abbildung unten*) definiert.



Durch Deklarieren des XPath-Standard-Namespace für `xs:schema` wird der XPath-Standard-Namespace für den Geltungsbereich des gesamten Schemas deklariert. Sie können diese Deklaration für Elemente, in denen das Attribut `xpathDefaultNamespace` zulässig ist (*siehe Liste oben*), außer Kraft setzen.

Anstelle eines tatsächlichen Namespace kann das Attribut `xpathDefaultNamespace` eines von drei Schlüsselwörtern enthalten:

- `##targetNamespace`: Der XPath-Standard-Namespace ist mit dem Target Namespace des Schemas identisch.
- `##defaultNamespace`: Der XPath-Standard-Namespace ist mit dem Standard-Namespace des Schemas identisch.
- `##local`: Es gibt keinen XPath-Standard-Namespace.

Wenn im Dokument kein XPath-Standard-Namespace deklariert ist, haben Elemente ohne Präfix in XPath-Ausdrücken keinen Namespace.

Anmerkung: Die XPath-Standard-Namespace-Deklaration gilt nicht für Attribute.

IDs von Identity Constraints

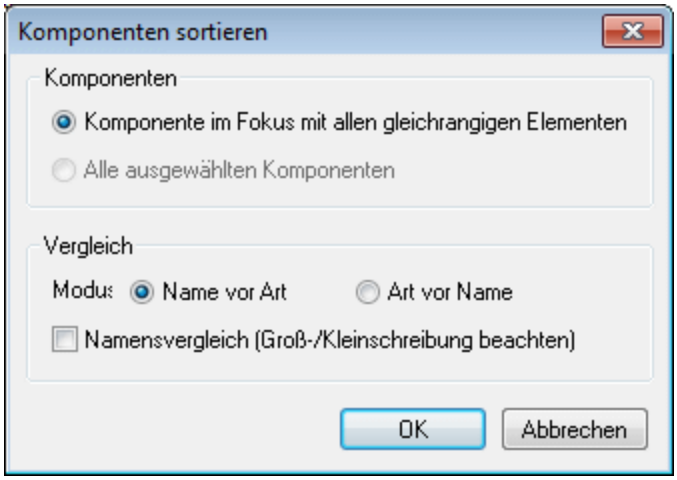
Eine ID kann einem Identity Constraint, seinem Selektor und/oder Feldern zugewiesen werden. Wählen Sie dazu die gewünschte Komponente aus und geben Sie die ID in der Eingabehilfe "Details" in die Zeile `id` ein.

Sortieren von Identity Constraints

Sie können die IDCs auf dem Register "Identity Constraints" durch Klicken auf die Schaltfläche **Sortieren** in der Symbolleiste des Registers sortieren. Im daraufhin angezeigten Dialogfeld "Komponenten sortieren" (*Abbildung unten)* können Sie auswählen, ob entweder die ausgewählte einzelne Komponente und ihre gleichrangigen Komponenten oder eine Gruppe von ausgewählten Komponenten sortiert werden soll. Mit

© 2016-2022 Altova GmbH

Hilfe von **Umschalt** + Klick können Sie einen Bereich auswählen, mit **Strg** + Klick können Sie zusätzliche Komponenten zur Auswahl hinzufügen.



Nach Auswahl des Bereichs können Sie festlegen, ob der gesamte Bereich alphabetisch (*Name vor Art*) oder alphabetisch nach Art (d.h. zuerst nach dem Unique Constraint, dann nach Key Constraints und dann nach Key References sortiert werden soll).

Die Sortierreihenfolge wird im Text des Schemas implementiert.

4.4.5 Eingabehilfen in der Schema-Ansicht

Es gibt drei Arten von Eingabehilfen in der Schema-Ansicht. In den Unterabschnitten dieses Abschnitts sind diese Eingabehilfen näher beschrieben.

- Eingabehilfe "Komponenten" 229
- Eingabehilfe "Details" 233
- Eingabehilfe "Facets" 235

Die Eingabehilfen sind in der Schema-Übersicht und der Content Model-Ansicht dieselben. Sie können damit Definitionen von Schemakomponenten grafisch hinzufügen und bearbeiten. Normalerweise können Sie Komponenten mit der Maus aus einer Eingabehilfe ziehen oder eine Komponente im Design auswählen und dann Eigenschaften dafür in einer Eingaberhilfe definieren.

4.4.5.1 Komponenten

Die Komponenten-Eingabehilfe in der Schema-Ansicht (*siehe Abbildungen unten*) hat drei Aufgaben:

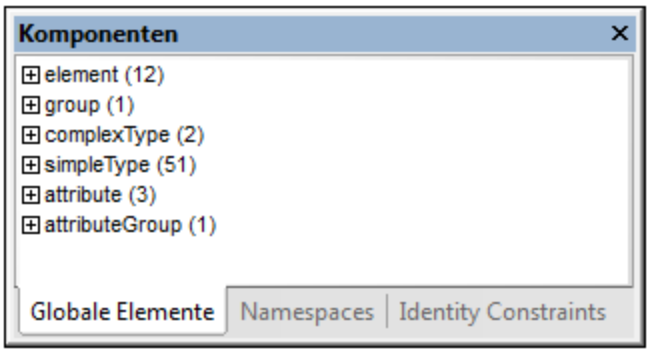
- Globale Komponenten in einem Baumdiagramm nach Komponententyp und Namespace (*siehe Abbildung unten)* zu strukturieren. Dadurch steht Ihnen eine strukturierte Ansicht aller globalen Komponenten und aller globalen Komponenten nach Namespace zur Verfügung.

- Ihnen das Navigieren zur Content Model-Ansicht einer globalen Komponenten zu ermöglichen und diese anzuzeigen - falls die Komponente ein Content Model hat. Wenn eine Komponente kein Content Model hat, wird die Komponente in der Schema-Ansicht markiert. Auch aus anderen Schemas inkludierte oder importierte globale Komponenten werden in der Komponenten-Eingabehilfe angezeigt.
- Ihnen eine Übersicht über die im Schemadokument definierten Identity Constraints zu verschaffen. Eine Beschreibung von Identity Constraints finden Sie im Abschnitt Identity Constraints 220.

Anmerkung: Ob die vordefinierten Datentypen von XSD 1.0 oder 1.1 angezeigt werden, hängt davon ab, welcher XSD-Modus (XSD 1.0 oder 1.1) ausgewählt ist.

Das Register "Globale Elemente"

Auf dem Register "Globale Elemente (*siehe Abbildung unten*) sind globale Komponenten nach Komponententyp in einer Baumstruktur gruppiert. Die Anzahl der einzelnen im Schema vorhandenen Komponententypen wird neben dem jeweiligen Komponententyp angezeigt.



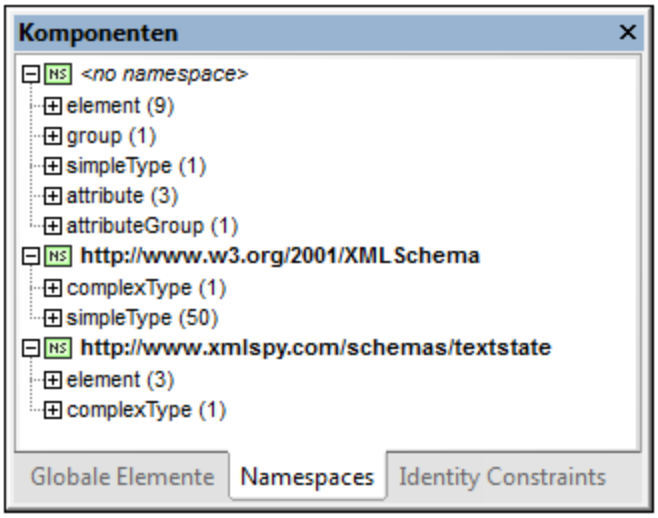
In der Baumstrukturanzeige werden globale Komponenten in die folgenden sieben Gruppen eingeordnet. Beachten Sie, dass ein Komponententyp nur dann in einer Baumstruktur angezeigt wird, wenn mindestens eine Komponente dieses Typs im Schema existiert.

- Element-Deklarationen (Elemente)
- Model Groups (Groups)
- ComplexTypes
- SimpleTypes
- Attribut-Deklarationen (Attribute)
- Attribut-Gruppen
- Notations

Das Register "Namespaces'

Auf dem Register "Namespaces" (*siehe Abbildung unten*) sind Komponenten zuerst nach Namespace und dann nach Komponententyp geordnet.

© 2016-2022 Altova GmbH

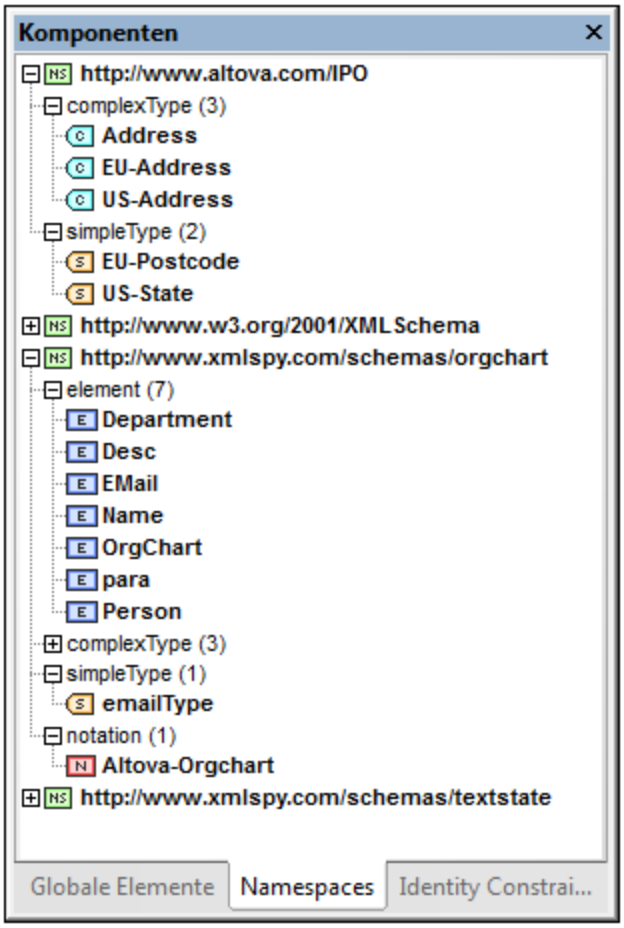


In der Baumstrukturanzeige werden globale Komponenten in die folgenden sieben Gruppen eingeordnet. Beachten Sie, dass ein Komponententyp nur dann in einer Baumstruktur angezeigt wird, wenn mindestens eine Komponente dieses Typs im Schema existiert.

- Element-Deklarationen (Elemente)
- Model Groups (Groups)
- ComplexTypes
- SimpleTypes
- Attribut-Deklarationen (Attribute)
- Attribut-Gruppen
- Notations

Gruppen von Komponententypen auf den Registern "Globale Elemente" und "Namespaces"

Wenn Sie eine Komponententyp-Gruppe auf dem Register "Globale Elemente" oder "Namespaces" in der Baumstruktur erweitern, werden alle Komponenten in dieser Gruppe angezeigt (*siehe Abbildung unten)*. Auf diese Art können Sie schnell zur benutzerdefinierten Komponente navigieren. Wenn Sie auf dem Register "Komponenten" auf die Komponente doppelklicken, wird ihre Definition im Hauptfenster angezeigt.



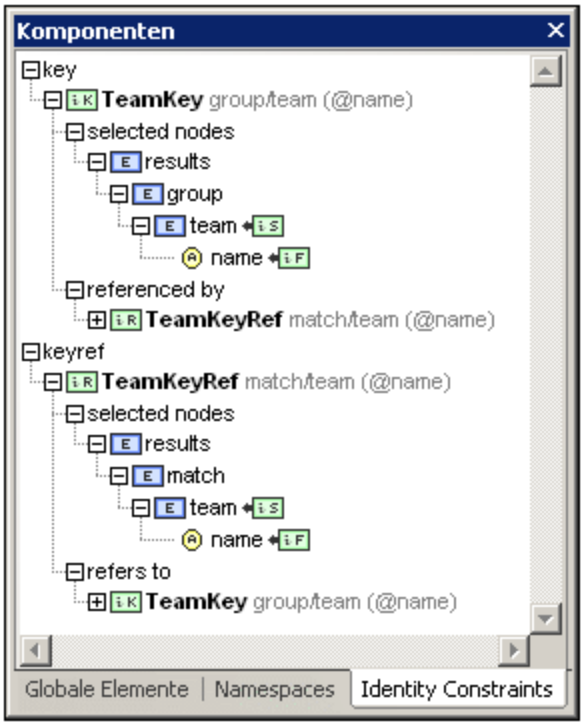
Wenn eine Komponente ein Content Model hat (d.h. wenn es sich um ein Element, eine Group oder einen complexType handelt), wird mit einem Doppelklick darauf das Content Model dieser Komponente in der Content Model-Ansicht (im Hauptfenster) angezeigt. Wenn die Komponente kein Content Model hat (d.h. wenn es sich um einen simpleType, ein Attribut, eine Attribut-Gruppe oder eine Notation handelt), so wird die Komponente in der Schema-Übersicht (im Hauptfenster) markiert.

Anmerkung: Wenn sich die Komponente in einem inkludierten oder importierten Schema befindet, wird dieses geöffnet (wenn es nicht bereits offen ist) und es wird entweder das Content Model der Komponente in der Content Model-Ansicht angezeigt oder die Komponente wird in der Schema-Übersicht markiert.

Identity Constraints

Auf dem Register "Identity Constraints" der Eingabehilfe "Komponenten" (*Abbildung unten*) sehen Sie eine Übersicht über die Identity Constraints eines Dokuments. Identity Constraints werden auf diesem Register nach der Art des Identity Constraint (`unique`, `key`, `keyref`) aufgelistet und in Form einer erweiter- und reduzierbaren Struktur angezeigt.

© 2016-2022 Altova GmbH



Fett gedruckte Einträge sind im aktuellen Schema vorhanden, während in normaler Schrift angezeigte Einträge in untergeordneten Schemas enthalten sind. Wenn Sie auf dem Register "Identity Constraints" auf einen Eintrag doppelklicken, wird diese Schemakomponente in der Content Model-Ansicht[190] angezeigt.

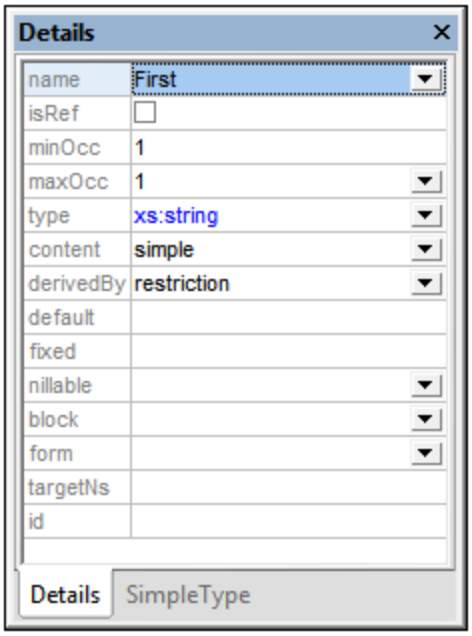
Wenn ein Eintrag auf dem Register "Identity Constraints" ausgewählt ist, stehen die folgenden Kontextmenübefehle zur Verfügung:

- *In Diagramm anzeigen*: Wählt die Schemakomponente in der Content Model-Ansicht[190] aus.
- *Selektorziel/Feldziel in Diagramm anzeigen*: Wählt in der Content Model-Ansicht[190] die Schemakomponente aus, die das Zeil des Selektors oder Felds des Identity Constraint ist. Wenn es mehrere Felder gibt, wird ein Dialogfeld angezeigt, in dem der Benutzer das gewünschte Feld auswählen kann.
- *Gehe zu Identity Constraint*: Wählt den Identiy Constraint in der Schemaübersicht[177] aus.
- *Alle erweitern/reduzieren*: Erweitert bzw. reduziert die Baumstruktur.

Eine Beschreibung des Registers "Identity Constraints" finden Sie im Abschnitt Identity Constraints[220].

4.4.5.2 Details

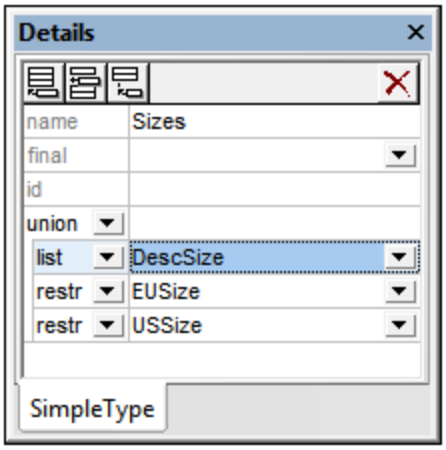
In der Details-Eingabehilfe der Schema-Ansicht stehen editierbare Informationen über den im Hauptfenster ausgewählten Kompositor/die ausgewählte Komponente zur Verfügung. Wenn Sie eine Schemadatei, die Datenbank-Extensions enthält, bearbeiten, wird unter Umständen ein zusätzliches Register mit Informationen über die DB Extensions angezeigt.



Um die Eigenschaften der ausgewählten Komponente oder des derzeit ausgewählten Kompositors zu ändern, doppelklicken Sie auf das zu editierende Feld und bearbeiten Sie es oder geben Sie Text direkt ein. Wenn in dem gewünschten Feld eine Auswahlliste zur Verfügung steht, wählen sie den gewünschten Wert aus der Dropdown-Liste aus. Änderungen, die Sie über die Details-Eingabehilfe vornehmen, werden sofort im Design angezeigt.

simpleType Ableitungen

Über die Eingabehilfe "Details" können Sie derived simpleTypes erstellen, nämlich von der Art `restriction`, `list` und `union`. Wenn im Design ein simpleType ausgewählt ist, enthält die Eingabehilfe "Details" das Register "SimpleType" (*siehe Abbildung unten*).



Wählen Sie in der Auswahlliste "derivation-type" des Registerss "SimpleType" den Ableitungstyp (`restrictiion`, `list` oder `union`) aus wählen Sie in der entsprechenden "member type"-Auswahlliste rechts davon einen simpleType aus den verfügbaren simpleTypes aus. Über die Schaltflächen in der Symbolleiste

© 2016-2022 Altova GmbH

können Sie einen Type auf derselben Ebene anhängen oder einfügen, eine weitere Ableitung auf einer untergeordneten Ebene hinzufügen oder einen abgeleiteten Type löschen. Um zur Definition eines Typs zu gelangen, klicken Sie mit der rechten Maustaste darauf und wählen Sie den Befehl **Gehe zu Typdefinition**. Bei simpleTypes wird ein Meldungsfeld mit Informationen über den simpleType angezeigt.

4.4.5.3 Facets

Ein neuer simpleType (benannt oder anonym) wird durch Einschränkung des Basistyps (also eines bestehenden simpleType) des simpleType erstellt. Dies wird durch Hinzufügen von Facets zur Einschränkung der Werte des Basistyps erzielt. In der Schema-Ansicht können Sie die Facets eines simpleType über die Eingabehilfe "Facets" (*siehe Abbildungen unten*) mühelos grafisch bearbeiten. Die verfügbaren Facets sind auf den Registern der Eingabehilfe "Facets" folgendermaßen gegliedert.

Register	Verfügbare Facets
Facets 235	`minInclusive`, `maxInclusive`, `minExclusive`, `maxExclusive`, `length`, `minLength`, `maxLength`, `totalDigits`, `fractionDigits`, `whiteSpace`, `explicitTimezone`
Patterns 236	`pattern`
Enumerations 237	`enumeration`
Assertions 237	`assertion`
Beispiele 239	`altova:exampleValues` ist eine Annotation und **kein Facet**. Mit Hilfe dieser Annotation werden Beispielwerte in dem von XMLSpy anhand des XML-Schemas generierten XML-Instanzdokument generiert.

Die einzelnen Register werden in den folgenden Abschnitten beschrieben.

Auswählen des simpleType im Design

Ein simpleType (benannt oder anonym) kann in den folgenden Design-Umgebungen ausgewählt werden:

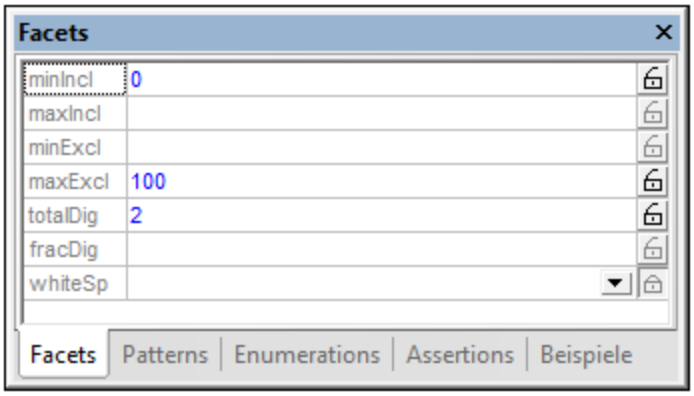
- In der Schema-Übersicht 177 (entweder in der Liste der globalen Komponenten oder auf dem Register "Attribute" unterhalb der Liste der globalen Komponenten) oder
- In der Content Model-Ansicht 190 (entweder im Diagramm oder auf dem Register "Attribute" unterhalb des Diagramms).

Wenn ein simpleType im Design in einer der oben aufgelisteten Design-Umgebungen ausgewählt wird, werden die anwendbaren Facets in der Eingabehilfe "Facets" angezeigt und können dort bearbeitet werden.

Das Register "Facets"

Auf dem Register "Facets" werden nur Facets, die für den im Design ausgewählten Typ verwendet werden können, angezeigt. Wenn z.B. der Typ `xs:string` eingeschränkt werden soll, werden nicht anwendbare Facets wie z.B. `totalDigits` nicht angezeigt.

- Die vier Eingrenzungsfacets (`minInclusive`, `maxInclusive`, `minExclusive`, `maxExclusive`) können nur auf numerische Typen und Datums- und Uhrzeittypen und auf von diesen Typen abgeleitete Typen angewendet werden.
- Die drei length-Facets (`length`, `minLength`, `maxLength`) können nur auf String-basierte Typen, die binären Typen und `anyURI` angewendet werden.
- Das `totalDigits` Facet wird auf `xs:decimal` und Ganzzahltypen sowie auf alle davon abgeleiteten Typen angewendet. Das `fractionDigits` Facet kann nur auf `xs:decimal` angewendet werden.

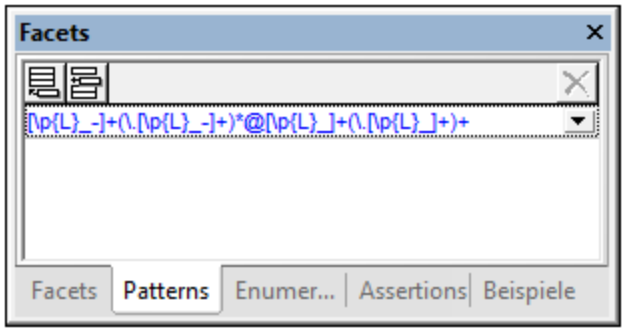


Um einen Wert einzugeben, wählen Sie entweder einen aus der Dropdown-Liste einer Auswahlliste aus (falls vorhanden) oder doppelklicken Sie in das Wertefeld und geben Sie einen Wert ein. Bei Eingabe eines ungültigen Werts werden die dadurch verursachten Konflikte rot angezeigt. Gültige Werte werden blau angezeigt. So können z.B. die Facets `minInclusive` und `maxInclusive` nicht gemeinsam verwendet werden; wenn daher ein Wert für das zweite dieser Facets eingegeben wird, werden die Werte beider Facets rot angezeigt.

Um ein **fixed Facet** (das Facet erhält einen Attributwert `fixed="true"`) zu definieren, klicken Sie auf das Symbol des offenen Schlosses rechts vom Facet, damit statt dessen ein versperrtes Schloss angezeigt wird. In der Abbildung oben wurde das `totalDigits` Facet als "fixed Facet" definiert. Sie können auch mehrere Facets als "fixed" definieren. Um die Fixierung aufzuheben, klicken Sie auf das Symbol des geschlossenen Schlosses, um ein offenes Schloss anzuzeigen.

Das Register "Patterns"

Auf dem Register "Patterns" (*Abbildung unten*) können Sie ein oder mehrere `pattern` Facets zu einer Restriction (Einschränkung) hinzufügen. Das Pattern (eines `pattern` Facet) wird mit der Syntax einer Regular Expression definiert. Mit dem Pattern in der Abbildung unten wird das Pattern von E-Mail-Adressen definiert.



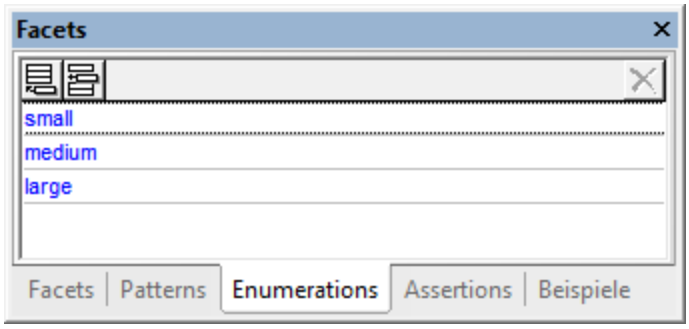
 © 2016-2022 Altova GmbH

Wenn mehrere `pattern` Facets definiert werden, so muss der Wert der XML-Instanz mit einem der definierten Patterns übereinstimmen. So könnte z.B. ein Pattern zum Einschränken von Postleitzahlen zwei `pattern` Facets haben, je eines für US- und eines für EU-Postleitzahlen. Der Wert einer XML-Instanz muss somit mit einem dieser Patterns übereinstimmen, um gültig zu sein.

Fügen Sie durch Klicken auf das **Anhängen-** oder **Einfügen-Symbol** links oben ein `pattern` Facet hinzu und geben Sie anschließend eine Regular Expression zur Definition des gewünschten Patterns ein. Um ein `pattern` zu löschen, wählen Sie es aus und klicken Sie auf das **Löschen**-Symbol in der rechten oberen Ecke.

Das Register "Enumerations"

Auf dem Register "Enumerations" (*Abbildung unten*) können Sie ein oder mehrere `enumeration` Facets zu einer Restriction hinzufügen. Jedes `enumeration` Facet definiert einen gültigen Wert für den Typ. Gemeinsam definiert eine Gruppe von `enumeration` Facets einen Bereich zulässiger Werte. In der Abbildung unten definieren die `enumeration` Facets die zulässigen Größenwerte für die Restriction.



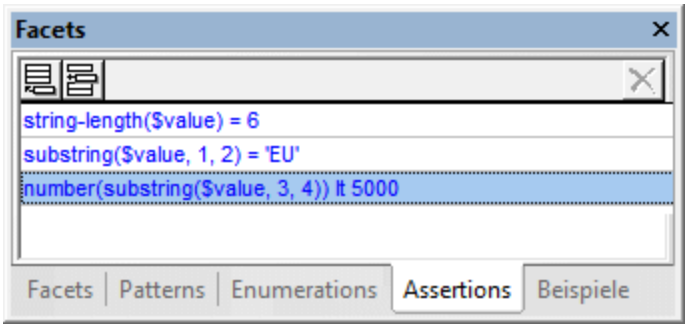
Durch Klicken auf das **Anhängen-** oder **Einfügen-**Symbol in der linken oberen Ecke können Sie ein `enumeration` Facet hinzufügen und anschließend den `enumeration` Wert eingeben. Um ein `enumeration` Facet zu löschen, wählen Sie es aus und klicken Sie auf das **Löschen**-Symbol in der rechten oberen Ecke.

Das Register "Assertions"

Assertions sind ein XSD 1.1 Feature. Daher wird das Register "Assertions" nur im XSD 1.1-Modus [173] angezeigt. Die auf dem Register "Assertions" der Eingabehilfe "Facets" angezeigten Assertion-Facets sind **Assertions für simpleTypes -** im Gegensatz zu Assertions für complexTypes (die in der Schema-Übersicht oder der Content Model-Übersicht, jedoch **nicht** in der Eingabehilfe "Facets" definiert und bearbeitet [217] werden können).

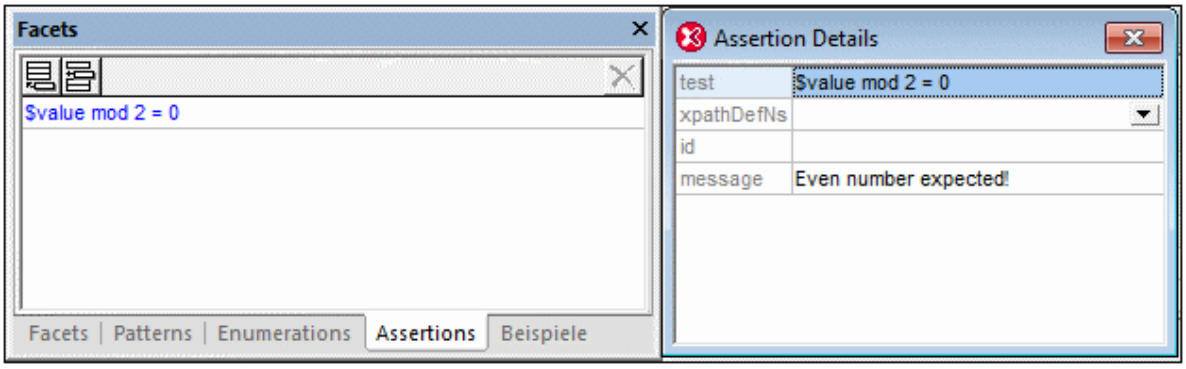
Wenn ein simpleType (Element oder Attribut mit "simple Content") im Design ausgewählt wird, können Sie auf das Register "Assertions" (*siehe Abbildung unten*) wechseln und auf das **Anhängen** oder **Einfügen**-Symbol in der linken oberen Ecke klicken, um dann den XPath 2.0-Ausdruck zur Definition der `assertion` einzugeben. Für den Wert des simpleType muss im XPath-Ausdruck eine spezielle Variable namens **`$value`** verwendet werden. (Beachten Sie, dass der normale `self::node()` Pfadschritt (bzw. die Abkürzung dafür durch einen Punkt '.') im XPath-Ausdruck nicht verwendet werden kann, da es keine Nachfahren, die überprüft werden sollen, gibt, sondern nur einen Wert.)

So überprüft z.B. der XPath-Ausdruck `string-length($value) = 6` (*siehe Abbildung unten*), ob der Wert des simpleType sechs Zeichen hat. Wenn das Element oder Attribut im Instanzdokument sechs Zeichen hat, ist es gemäß der Assertion gültig.



Anmerkung: Syntaxfehler in XPath-Ausdrücken werden markiert, indem der Ausdruck rot angezeigt wird. Da der Datentyp allerdings erst zur Laufzeit ermittelt wird, werden Typ-Fehler bei der Eingabe des XPath-Ausdrucks nicht markiert. Achten Sie darauf, benötigte Typen zu konstruieren. Ein Beispiel für ein Typ-Konstrukt finden Sie im dritten XPath-Ausdruck in der Abbildung oben, der (unter der Voraussetzung, dass die Assertion für einen `xs:string` simpleType definiert wurde) einen String-Wert in eine Zahl konvertiert, bevor ein numerischer Vergleich durchgeführt wird.

Wie in der Abbildung oben gezeigt, können für einen einzigen simpleType mehrere Assertions definiert werden. In diesem Fall müssen alle Assertions zutreffen, damit das Element oder Attribut im Instanzdokument gültig ist. Die Assertions in der Abbildung oben geben an, dass der Wert des Instanzdokuments ein aus sechs Zeichen bestehender String sein muss, der mit den Buchstaben `EU` beginnt und dessen letzte vier Zeichen eine Zahl mit dem Wert `0000` bis `4999` sein muss. Um die Details einer Assertion zu bearbeiten, klicken Sie in der Eingabehilfe "Facets" mit der rechten Maustaste auf die Assertion und klicken Sie im daraufhin angezeigten Menü auf **Details.** Daraufhin wird das Fenster Assertion Details angezeigt (*siehe Abbildung unten).*



Oft ist es nützlich, wenn z.B. mit der Definition einer Assertion auch eine Erläuterung dazu verfügbar ist, damit eine entsprechende Meldung angezeigt werden kann, wenn die Assertion bei der Validierung des XML-Instanzdokuments nicht erfüllt wird. Da in der XML Schema-Spezifikation keine derartige Meldung vorgesehen ist, können Sie in XMLSpy im Altova xml-schema-extensions Namespace `http://www.altova.com/de/xml-schema-extensions` (oder in jedem anderen Namespace) zusammen mit der Definition der Assertion eine Meldung definieren, die bei der Validierung des XML-Instanzdokuments verwendet werden soll. Zum Beispiel:

```
<xs:assert test="count(//MyNode) ge 1" altova:message="Es muss mindestens ein MyNode-
Element vorhanden sein"/> oder
<xs:assertion test="count(//MyNode) ge 1" altova:message="Es muss mindestens ein
MyNode-Element vorhanden sein"/>
```

Wenn die in der Assertion definierte Einschränkung nicht erfüllt wird, zeigt der Validierungsprozessor von XMLSpy zusammen mit der Validierungsfehlermeldung die mit der Assertion verknüpfte Erklärung als Meldung

 © 2016-2022 Altova GmbH

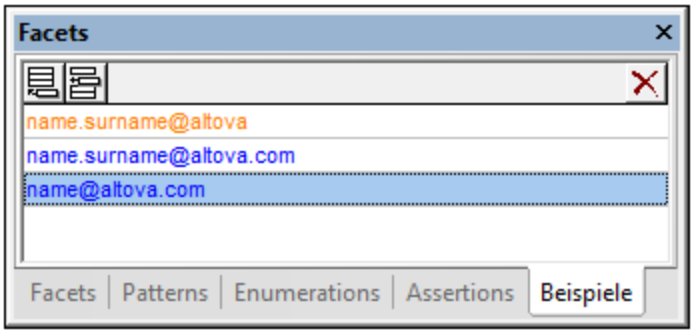
an. Der Validierungsprozessor gibt den Wert eines `assert/@message` Attributs oder eines `assertion/@message` Attributs unabhängig davon aus, in welchem Namespace sich das `message` Attribut befindet. Sie können in der Schema-Ansicht jedoch nur `message` Attribute bearbeiten, die sich im Altova xml-schema-extension Namespace befinden. Um `message` Attribute in anderen Namespaces zu bearbeiten, verwenden Sie die Textansicht.

Nähere Informationen dazu finden Sie unter Assertion-Meldungen 241.

Anmerkung: Erfahrungsgemäß empfiehlt es sich, wo dies möglich ist, andere Facets anstelle von Assertions zu verwenden. So könnte man z.B. die mit der ersten Assertion in der Abbildung definierte Einschränkung besser mit dem `length` Facet (auf dem Register "Facets") definieren.

Das Register "Beispiele"

Auf dem Register "Beispiele" (*Abbildung unten*) können Sie Beispielwerte definieren, die bei der Generierung einer XML-Datei anhand des Schemas (Menübefehl **DTD/Schema | XML-Beispieldatei erzeugen**) verwendet werden können. Wenn ein Beispielwert ungültig ist, wird der Beispielwert zur Warnung orange angezeigt. Der erste Wert in der Abbildung unten ist ungültig, weil er nicht mit dem für E-Mails definierte `pattern` Facet übereinstimmt (siehe Register *Patterns* oben).



Anmerkung: Klicken Sie in der Symbolleiste auf die Schaltfläche **Validierungswarnungen anzeigen** , um Warnungen über ungültige Beispielwerte anzuzeigen. Durch einen ungültigen Beispielwert wird die XSDL-Datei nicht ungültig, wenn Sie in anderer Hinsicht gültig ist.

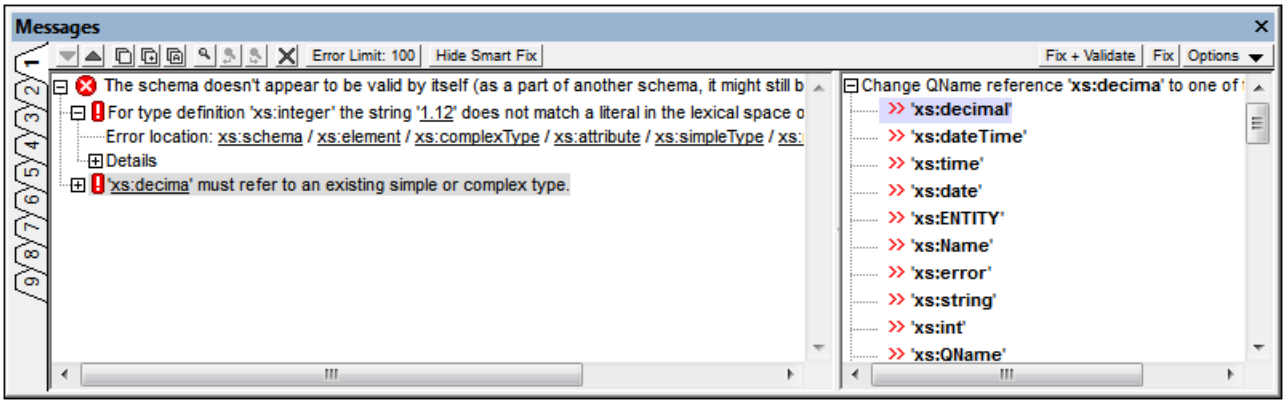
Beispielwerte werden in einem `altova:example` Annotation-Element, das sich im Namespace `http://www.altova.com/de/xml-schema-extensions` befindet, gespeichert. Fügen Sie durch Klicken auf das **Anhängen**- oder **Einfügen**-Symbol links oben und Eingabe des `altova:example`-Werts eine `altova:example` Annotation hinzu. Um die Annotation zu löschen, wählen Sie sie aus und klicken Sie in der rechten oberen Ecke auf das **Löschen**-Symbol.

4.4.6 Validierung und Smart Fixes

Ein XML-Schema-Dokument kann auf seine Gültigkeit überprüft werden. Wählen Sie dazu den Menübefehl **XML | XML validieren (F8)** aus.

Wenn das Dokument gültig ist, wird eine entsprechende Meldung im Fenster "Meldungen" angezeigt.

Wenn das Dokument ungültig ist, werden statt dessen im Fenster "Meldungen" zwei Bereiche (*siehe Abbildung unten*) angezeigt. Im linken Bereich (dem Fehler-Bereich) werden die ersten `x` Fehler oder alle Fehler angezeigt. Der rechte Bereich ist der Smart Fix-Bereich; er enthält eine Liste mit Behebungsvorschlägen für den im linken Bereich ausgewählten Fehler. In der Abbildung unten ist z.B. im Fehlerbereich der zweite Fehler ausgewählt. Im Smart Fix-Bereich auf der rechten Seite werden Behebungsvorschläge zu diesem Fehler aufgelistet. Wenn Sie einen der Behebungsvorschläge auswählen und anschließend auf entweder **Beheben + validieren** oder **Beheben** klicken, wird dieser Behebungsvorschlag im Dokument ausgeführt.



Fehlerbereich

In der Symbolleiste des Fensters stehen die folgenden Funktionen zur Auswahl:

- Scrollen durch die Fehler mit Hilfe des **Pfeils nach unten** bzw. **nach oben**.
- Kopieren einer Meldung oder Kopieren einer Meldung und all ihrer untergeordneten Meldungen in die Zwischenablage.
- Suchen nach bestimmten Wörtern mittels der Funktionen "Suchen", "Weitersuchen" und "Vorheriges suchen". Dies eignet sich v.a., wenn mehrere Fehler ausgegeben wurden.
- Löschen aller Fehler aus dem Fehlerbereich.
- Definieren eines Fehlerlimits für die Anzahl der gefundenen und angezeigten Fehler (`1` bis `999`). Der Standardwert ist `100`. Klicken Sie auf die Schaltfläche, um das Limit zu ändern.
- Ein-/Ausblenden des Smart Fix-Bereichs. Wenn der Smart Fix-Bereich ausgeblendet ist, erscheint in der Symbolleiste die Schaltfläche **Smart Fix anzeigen.** Wenn Sie darauf klicken, wird der Smart Fix-Bereich wieder eingeblendet und die Schaltfläche ändert sich in **Smart Fix ausblenden**. Wenn die **Smart Fix anzeigen/Smart Fix ausblenden** Schaltfläche deaktiviert ist, steht kein Smart Fix zur Verfügung.

Smart Fix-Bereich

In der Symbolleiste des Fensters stehen die folgenden Funktionen zur Auswahl:

- Mit der Schaltfläche **Beheben + validieren** wird der ausgewählte Fehler mit dem ausgewählten Smart Fix-Vorschlag behoben und das Dokument erneut validiert. Etwaige andere Fehler werden im Fehlerbereich aufgelistet.
- Bei Klick auf die Schaltfläche **Beheben** wird der Fehler behoben, doch wird keine erneute Validierung durchgeführt.
- Bei Klick auf die Schaltfläche **Optionen** wird eine Liste angezeigt, aus der Sie auswählen können, was bei Doppelklick auf einen Smart Fix geschehen soll: **Beheben + validieren** oder **Beheben**.

© 2016-2022 Altova GmbH

4.4.7 Assertion-Meldungen

In XML Schema 1.1 können für complexTypes (mittels `xs:assert` Elementen) und simpleTypes (mittels `xs:assertion` Elementen) Assertions definiert werden.

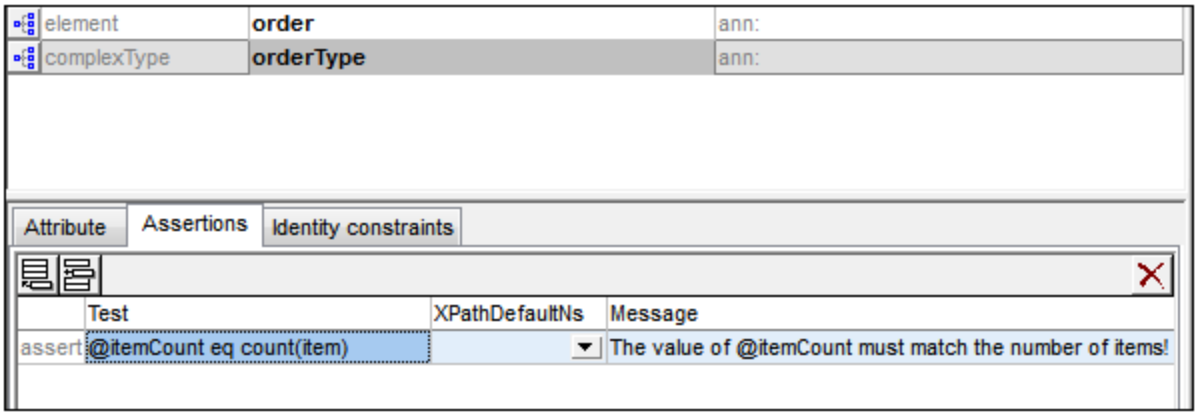
Oft ist es nützlich, wenn z.B. mit der Definition einer Assertion auch eine Erläuterung dazu verfügbar ist, damit eine entsprechende Meldung angezeigt werden kann, wenn die Assertion bei der Validierung des XML-Instanzdokuments nicht erfüllt wird. Da in der XML Schema-Spezifikation keine derartige Meldung vorgesehen ist, können Sie in XMLSpy im Altova xml-schema-extensions Namespace `http://www.altova.com/de/xml-schema-extensions` (oder in jedem anderen Namespace) zusammen mit der Definition der Assertion eine Meldung definieren, die bei der Validierung des XML-Instanzdokuments verwendet werden soll. Zum Beispiel:

```
<xs:assert test="count(//MyNode) ge 1" altova:message="Es muss mindestens ein MyNode-Element vorhanden sein"/> oder
<xs:assertion test="count(//MyNode) ge 1" altova:message="Es muss mindestens ein MyNode-Element vorhanden sein"/>
```

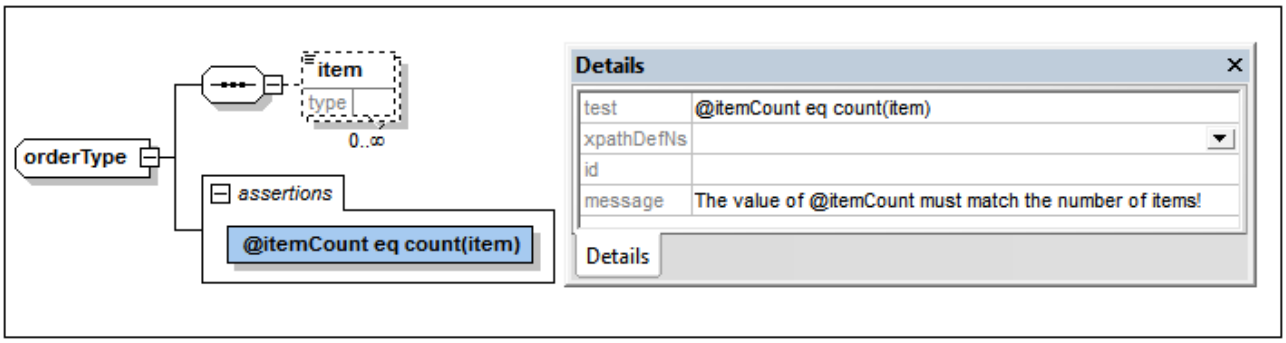
Wenn die in der Assertion definierte Einschränkung nicht erfüllt wird, zeigt der Validierungsprozessor von XMLSpy zusammen mit der Validierungsfehlermeldung die mit der Assertion verknüpfte Erklärung als Meldung an. Der Validierungsprozessor gibt den Wert eines `assert/@message` Attributs oder eines `assertion/@message` Attributs unabhängig davon aus, in welchem Namespace sich das `message` Attribut befindet. Sie können in der Schema-Ansicht jedoch nur `message` Attribute bearbeiten, die sich im Altova xml-schema-extension Namespace befinden. Um `message` Attribute in anderen Namespaces zu bearbeiten, verwenden Sie die Textansicht.

Bearbeiten von xs:assert Meldungen

In der Schema-Ansicht können im Attributes/Assertions/Identity Constraints (AAIDC)-Bereich[217] oder in der Eingabehilfe Details[217] des jeweiligen complexType (für complexTypes) `xs:assert` Elemente erstellt und bearbeitet werden. In der Abbildung unten sehen Sie eine Assertion für den complexType `orderType`. Die Assertion (in diesem Fall ein `xs:assert`) wird zusammen mit einer Assertion-Meldung auf dem Register "Assertions" (der Schema-Übersicht) definiert.

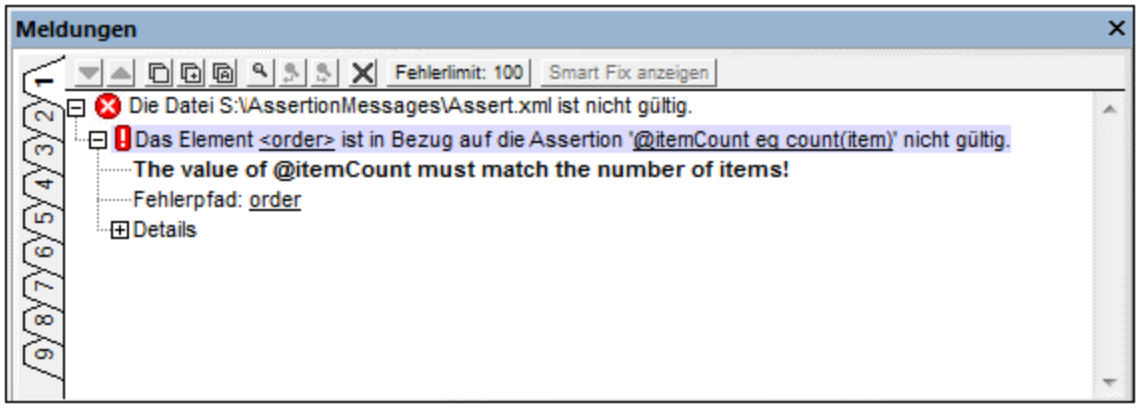


Wenn die Option **Assertions im Diagramm anzeigen**[196] aktiviert ist, können Assertions zu complexTypes auch in der Content Model-Ansicht erstellt und bearbeitet werden. Um eine Assertion-Meldung hinzuzufügen oder zu bearbeiten, wählen Sie die Assertion aus und geben Sie die Assertion-Meldung in die Eingabehilfe "Details" ein (*siehe Abbildung unten*).



Beachten Sie, dass auf diese Art erstellte Assertion-Meldungen sich im Altova xml-schema-extensions Namespace `http://www.altova.com/de/xml-schema-extensions` befinden. Wenn Sie die erste Assertion-Meldung im XML-Schema-Dokument über den AAIDC-Bereich[217] oder die Eingabehilfe "Details"[217] hinzufügen, wird zum `xs:schema` Element automatisch der Altova xml-schema-extensions Namespace deklariert.

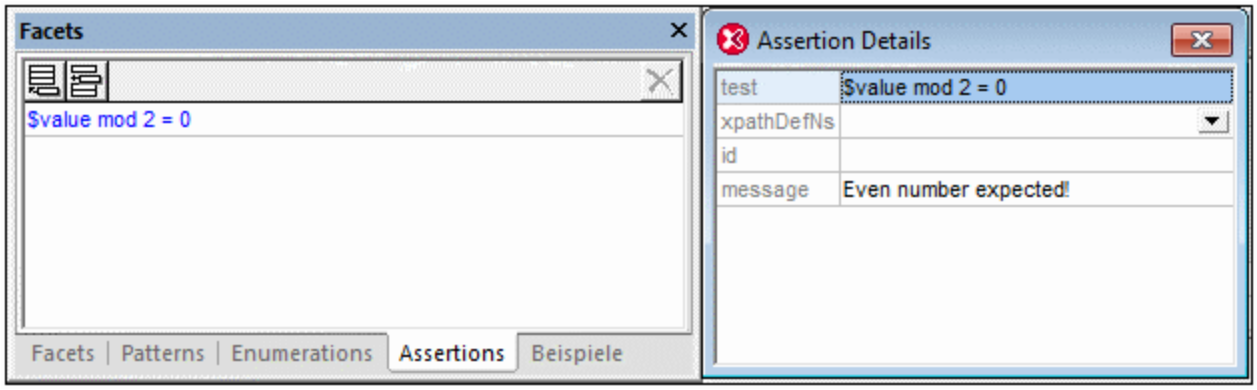
Wenn eine XML-Datei validiert wird und die Assertion bei der Überprüfung als nicht zutreffend ausgewertet wird, wird die für die Assertion definierte Meldung zusammen mit einer Fehlermeldung angezeigt (*siehe Abbildung unten*).



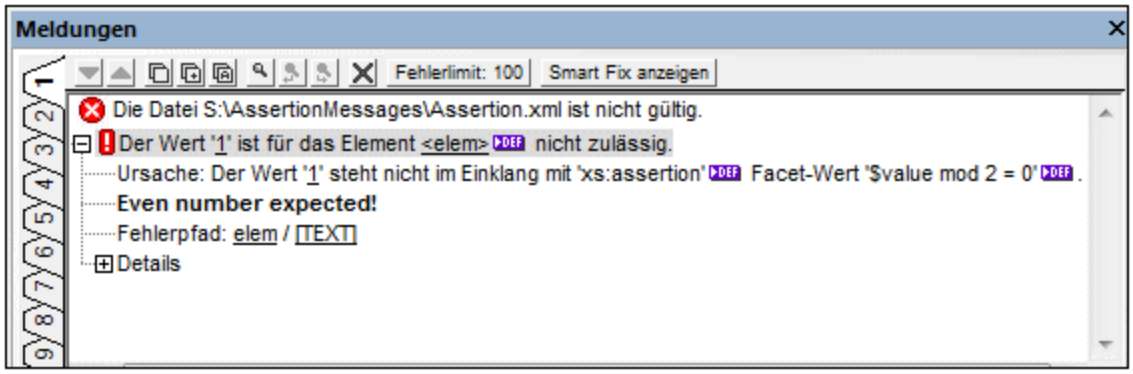
Bearbeiten von xs:assertion Meldungen

In der Schema-Ansicht können in der Eingabehilfe "Facets"[235] für den jeweiligen simpleType `xs:assertion` Elemente (für simpleTypes) erstellt und bearbeitet werden. Um die Assertion-Meldung zu bearbeiten, klicken Sie mit der rechten Maustaste in der Eingabehilfe "Facets" auf die Assertion (*siehe Abbildung unten*), klicken Sie im daraufhin angezeigten Menü auf **Details** und bearbeiten Sie die Meldung im Modusfenster der Assertion Details (*siehe Abbildung unten*).Beachten Sie, dass auf diese Art erstellte Assertion-Meldungen sich im Altova xml-schema-extensions Namespace `http://www.altova.com/de/xml-schema-extensions` befinden. Wenn Sie die erste Assertion-Meldung im XML-Schema-Dokument über das Assertions-Details-Modusfenster eingeben, wird der Namespace automatisch im `xs:schema` Element deklariert.

© 2016-2022 Altova GmbH



Wenn eine XML-Datei validiert wird und die Assertion bei der Überprüfung als nicht zutreffend ausgewertet wird, wird die für die Assertion definierte Meldung zusammen mit einer Fehlermeldung angezeigt (*siehe Abbildung unten*).

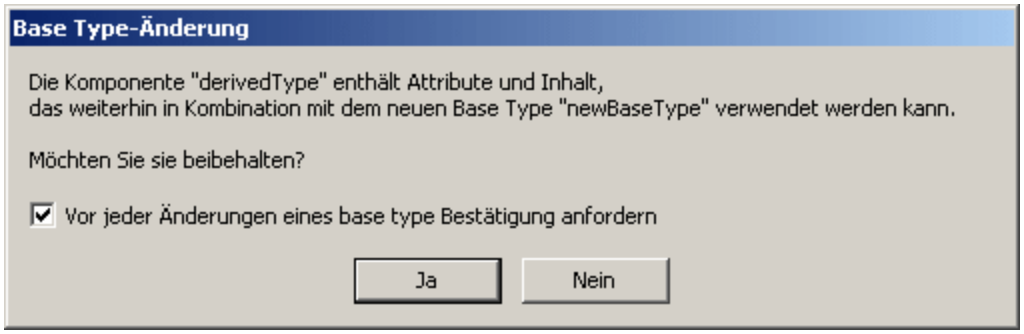


4.4.8 Modifizieren des Base Type

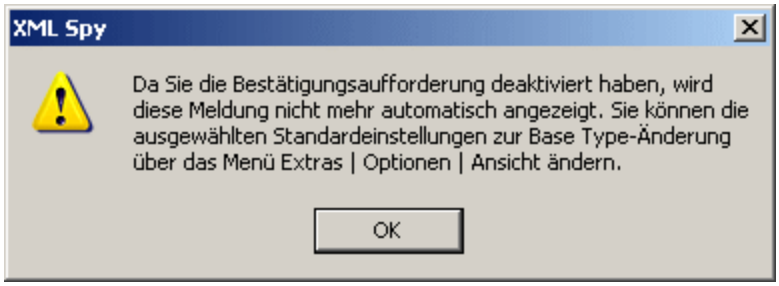
Wenn der Base Type eines abgeleiteten Types in der Schema-Ansicht geändert wird, können Inhalt, Attribute, Facets und Beispielwerte, die innerhalb des abgeleiteten Type definiert sind, auf eine von zwei Arten behandelt werden:

- Sie können erhalten werden, wenn sie in Kombination mit dem neuen Base Type noch anwendbar sind.
- Sie können automatisch entfernt werden, unabhängig davon ob sie in Kombination mit dem neuen Base Type noch anwendbar sind oder nicht.

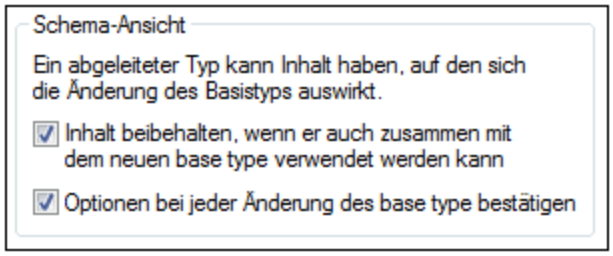
Wenn der Base Type eines abgeleiteten Type, der noch Inhalt, Attribute, Facets oder Beispielwerte enthält, geändert wird, erscheint das Dialogfeld "Base Type-Änderung".



Wenn das Kontrollkästchen der Bestätigungsaufforderung deaktiviert wird, erscheint eine Meldung (*siehe Abbildung unten*) mit dem Hinweis, dass die Bestätigung im Abschnitt "Ansicht" im Dialogfeld "Optionen" (**Extras | Optionen | Ansicht**[1617]) wieder eingeschaltet werden kann.



Im Bereich "Schema-Ansicht" (*siehe Abbildung unten*) im Abschnitt "Ansicht" im Dialogfeld "Optionen" (**Extras | Optionen | Ansicht**[1617]) können Sie festlegen, ob Inhalte beibehalten sollen und ob für jede Modifikation des Base Type eine Bestätigung des Benutzers erforderlich ist.



Aktivieren Sie die entsprechenden Kontrollkästchen um Inhalt beizubehalten und eine Bestätigung zu fordern wenn Sie wünschen, dass dies die Standardoptionen sein sollen.

4.4.9 Smart Restrictions

Bei Einschränkung eines complexType mittels Restriction werden Teile des Content Models des Basistyps im abgeleiteten Typ neu geschrieben. Dies kann verwirrend sein, wenn es sich um ein komplexes Content Model handelt, da es beim Editieren des abgeleiteten Typs oft nicht einfach ist, sich genau zu erinnern, wie das Content Model des Basistyps aussieht.

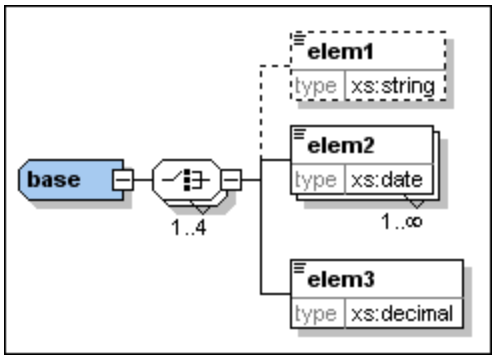
© 2016-2022 Altova GmbH

Mit Hilfe von Smart Restrictions werden die beiden Content Models in der grafischen Ansicht des abgeleiteten Content Model kombiniert und in Zusammenhang gesetzt. Im abgeleiteten complexType sehen Sie alle Partikel des Basis-complexType und in welcher Beziehung sie zum abgeleiteten Typ stehen. Außerdem bieten Smart Restrictions visuelle Anhaltspunkte und zeigen alle Möglichkeiten, wie der Basistyp mittels Restriction eingeschränkt werden kann, wodurch es einfacher wird, den abgeleiteten Typ korrekt einzuschränken.

So aktivieren Sie die Option "Smart Restrictions":

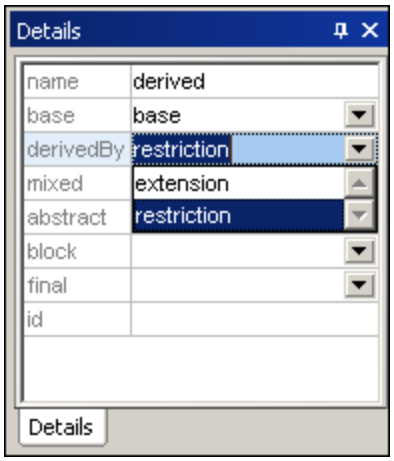
- Klicken Sie auf das Symbol "Smart Restrictions" .

Im folgenden Beispiel sehen Sie, wie sich Smart Restrictions auswirken.
Der folgende complexType ist der Basis-Typ, der in diesem Beispiel verwendet wird:

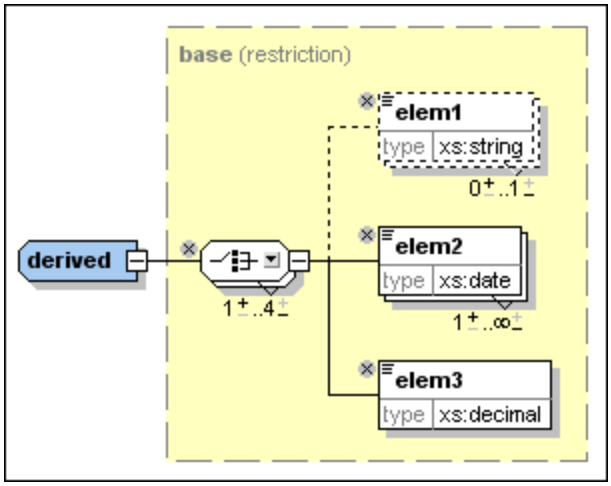


Der complexType "derived" wird folgendermaßen vom "Base" Type abgeleitet:

1. Erstellen Sie im Schema einen neuen complexType und nennen Sie ihn "derived".
2. Wählen Sie in der Eingabehilfe "Details" den Eintrag "base" aus der Dropdown-Liste **base** aus und den Eintrag "restriction" aus der Dropdown-Liste **derivedBy**.

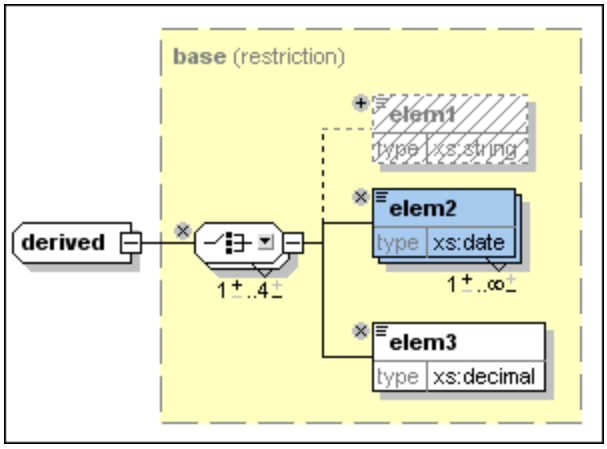


Wenn die Option "Smart Restrictions" aktiviert ist, so sieht der neue abgeleitete Typ folgendermaßen aus:



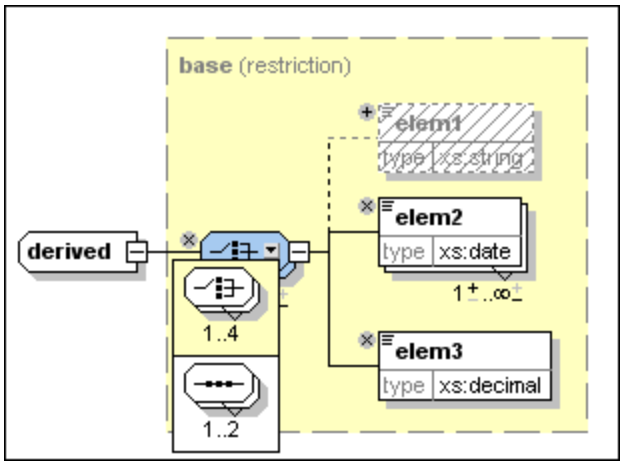
Beachten Sie die folgenden Steuerelemente, mit denen Sie den abgeleiteten Typ in diesem Beispiel durch Restriction einschränken können:

- Mit Hilfe des Symbols  können Sie Elemente aus dem abgeleiteten Typ entfernen, die sich im Basis-Typ befinden. Hier wurde elem1 gelöscht. Um es erneut hinzuzufügen, klicken Sie auf dieses Symbol .

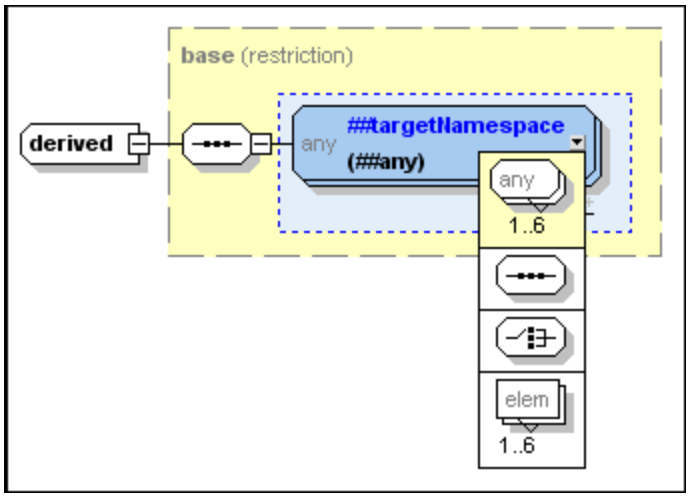


- Klicken Sie auf den Abwärtspfeil neben dem "Choice"-Kompositor, um die folgende Liste aufzurufen, über die Sie die Model Group "Choice" in eine Model Group "Sequence" ändern können:

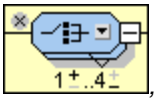
© 2016-2022 Altova GmbH

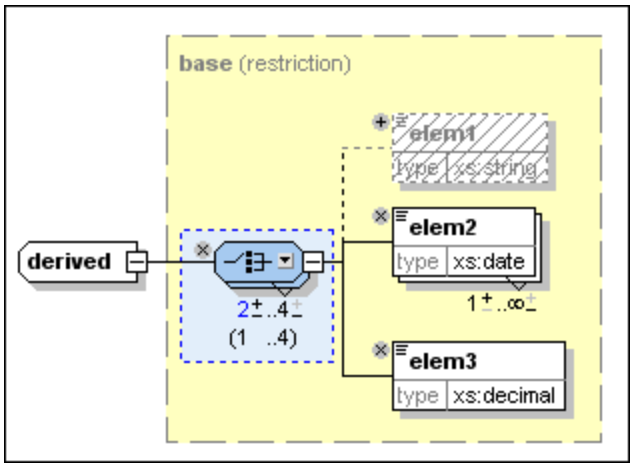


Sie können auch Wildcards auf dieselbe Art ändern - siehe Beispiel:



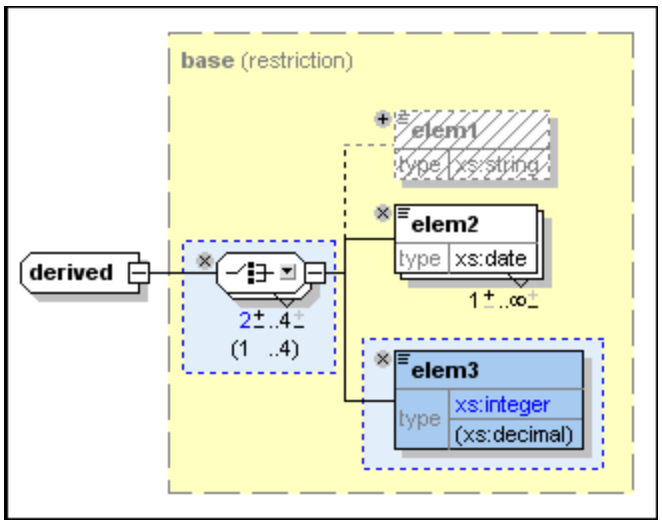
Ein ausführliche Liste darüber, welche Partikel durch welche anderen Partikel ersetzt werden können, finden Sie in der XML-Schema-Spezifikation.

- Klicken Sie auf das folgende Symbol , um die Anzahl der "Occurrences" der Model Group zu ändern. Durch Klicken auf das +-Zeichen über der 1 können Sie die Mindestanzahl der Occurrences erhöhen, durch Klicken auf das Minus-Zeichen unterhalb von "4" können Sie die Maximalanzahl der Occurrences verringern. Diese Steuerelemente werden angezeigt, wenn es sich beim Occurrence-Bereich um einen realen Bereich (z.B. 2-5) handelt und nicht um einen bestimmten Wert (z.B. 4-4). Sie werden auch angezeigt, wenn der Occurrence-Bereich falsch ist.



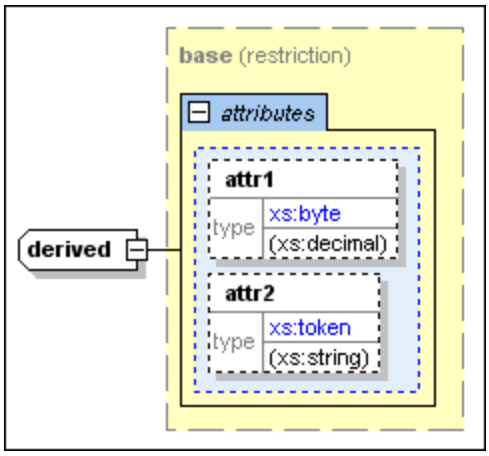
Hier sehen Sie, dass die Minimum Occurrence für dieses Element in 2 geändert wurde. Beachten Sie, dass die Model Group nun einen blauen Hintergrund aufweist, was bedeutet, dass sie nicht mehr mit der Model Group im Base complexType identisch ist. Außerdem wird der Occurrence-Bereich der Model Group im Basiselement nun in Klammern angezeigt.

- Sie können die Datentypen von Attributen oder Elementen ändern, wenn es sich beim neuen Datentyp um eine gültige Restriction des in der XML Schema-Spezifikation definierten Basis-Typs handelt. So können Sie z.B. den Datentyp von elem3 im Datentyp "derived" von Decimal in Integer ändern. Das Element wird danach mit blauem Hintergrund angezeigt, um hervorzuheben, dass es nicht mit dem Element im Basis-Typ identisch ist. Der Typ, den das Element im Basis-Typ hat, wird in Klammern angezeigt:

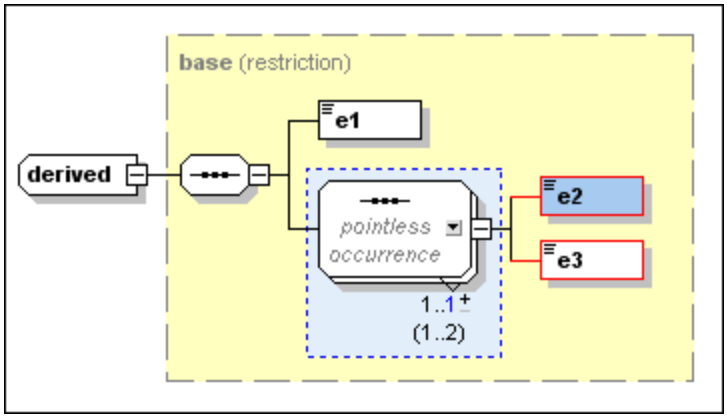


In diesem Beispiel sehen Sie Attribute, deren Datentypen im abgeleiteten complexType durch Restriction eingeschränkt wurden.

 © 2016-2022 Altova GmbH



- Smart Restrictions informieren Sie auch über *überflüssige Vorkommen* (pointless occurrences) im Content Model. Ein "überflüssiges Vorkommen" wird z.B. dann angezeigt, wenn eine im Content Model vorhandene Sequenz unnötig ist. In diesem Beispiel sehen Sie ein solches "überflüssiges Vorkommen":



Anmerkung: Überflüssige Vorkommen werden nur angezeigt, wenn das Content Model einen Fehler enthält. In manchen Fällen enthält ein Content Model ein "überflüssiges Vorkommen" ist aber trotzdem gültig. In diesen Fällen wird das überflüssige Vorkommen nicht explizit angezeigt, um nicht zu Verwirrung zu führen.

Nähere Informationen über "überflüssige Vorkommen" (Pointless occurrences) finden Sie in der XML-Schema-Spezifikation.

4.4.10 xml:base, xml:id, xml:lang, xml:space

Der Namespace `http://www.w3.org/XML/1998/namespace` ist, gemäß der XML Namespace-Spezifikation definitionsgemäß an das Präfix `xml:` gebunden, d.h. dieser Namespace muss mit dem dafür reservierten Präfix `xml:` verwendet werden. Es gibt vier Attribute in diesem Namespace, die Children von jedem XML-Element in jedem XML-Dokument (Schema oder Instanz) sein können:

- `xml:base` (zum Setzen der Basis-URI eines Elements)
- `xml:id` (zum Definieren der eindeutigen ID eines Elements)

- `xml:lang` (zur Ermittlung der in diesem Element verwendeten Sprache)
- `xml:space` (um zu definieren, wie Leerzeichen im Element behandelt werden sollen)

Nachdem der XML Namespaces Namespace in das XML-Schema-Dokument importiert wurde, können in der Schema-Ansicht diese vier `xml:` Attribute zur Verwendung in jedem Element im Schema referenziert werden.

Um eines dieser Attribute in einem Element zu deklarieren, gehen Sie folgendermaßen vor:

1. Deklarieren Sie den XML Namespaces Namespace für dieses Schema-Dokument und binden Sie den Namespace an das Präfix `xml:`. Wenn eines der vier `xml:` Attribute im Dokument verwendet wird, würde sein Name anschließend um den korrekten Namespace-Bestandteil erweitert.
2. Importieren Sie den XML Namespaces Namespace. Der Validator von XMLSpy erkennt den Namespace und stellt die vier `xml:` Attribute als globale Attribute zur Verfügung, die in diesem Schema referenziert werden können.
3. Fügen Sie das erforderliche `xml:` Attribut als Child eines Elements ein. Das Attribut wird als Referenz zum globalen Attribut "imported" deklariert.

Deklarieren des XML Namespaces Namespace

Sie können den XML Namespaces Namespace (`http://www.w3.org/XML/1998/namespace`) durch Eingabe über das Dialogfeld "Schema-Einstellungen", in dem alle für dieses Schema deklarierten Namespaces gespeichert und bearbeitet werden können, deklarieren. Der Namespace muss an das Präfix `xml:` gebunden werden. (Alternativ dazu könnten Sie den Namespace (mit dem `xml:` Präfix) in der Textansicht im Element `xs:schema` deklarieren.)

Importieren des XML Namespaces Namespace

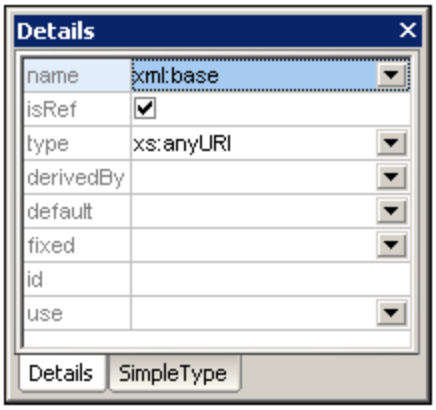
Erstellen Sie in der Schemaübersicht eine globale Import-Deklaration für den XML Namespaces Namespace. Klicken Sie dazu im oberen Bereich des Schemaübersichtsfensters auf die Schaltfläche "Einfügen"  oder "Anhängen"  und wählen Sie im Menü, das daraufhin angezeigt wird, den Eintrag **Import** aus. Geben Sie den XML Namespaces Namespace als den zu importierenden Namespace ein. Die Import-Deklaration sollte in der Textansicht folgendermaßen aussehen:

```
<xs:import namespace="http://www.w3.org/XML/1998/namespace"
schemaLocation="http://www.w3.org/XML/1998/namespace"/>.
```

Hinzufügen des xml: Attributs

Wählen Sie in der Schemaübersicht das Element aus, für das das `xml:` Attribut hinzugefügt werden soll und fügen Sie ein Attribut dazu hinzu. Klicken Sie in der Eingabehilfe "Details" (*Abbildung unten*) auf den Dropdown-Pfeil der Auswahlliste "Name" und wählen Sie das erforderliche `xml:` Attribut aus, z.B. `xml:base`. Wenn Sie gefragt werden, ob das globale Attribut referenziert werden soll, klicken Sie auf **Ja**. Das Attribut wird als Referenz hinzugefügt.

© 2016-2022 Altova GmbH



XInclude und xml:base

Wenn das `include` Element von XInclude durch die im `href` Attribut des Include-Elements referenzierte XML-Datei ersetzt wird, wird das Element der obersten Ebene des geparsten XML-Dokuments mit einem `xml:base` Attribut inkludiert. Wenn dieses XML-Dokument validiert werden soll, muss im Schema ein xml:base Attribut zum/zu den relevanten Element(en) definiert werden.

4.4.11 Zurück und Vorwärts: Navigieren von einer Position zu einer anderen

Mit den Befehlen **Zurück** und **Vorwärts** können Sie in der Schema-Ansicht von einer bereits angesehen Position zu einer anderen Position springen. Dies ist nützlich, da Sie beim Klicken durch Schemakomponenten in der Schema-Ansicht manchmal eine zuvor angezeigte Komponente sehen möchten. Wenn Sie in der Schema-Ansicht in der Symbolleiste einmal auf die Schaltfläche **Zurück** klicken, gelangen Sie zur zuvor angezeigten Position. Durch wiederholtes Klicken auf **Zurück** können Sie bis zu 500 der letzten besuchten Komponenten anzeigen. Um anschließend wieder vorwärts zu navigieren, klicken Sie in der Symbolleiste auf die Schaltfläche **Vorwärts.**

Die Tastaturkürzel für diese beiden Befehle sind:

-  ***Zurück:*** **Alt + Pfeil nach links**
-  ***Vorwärts:*** **Alt + Pfeil nach rechts**

Zurück/Vorwärts im Gegensatz zu Rückgängig/Wiederholen

Beachten Sie bitte, dass die Befehle **Zurück** und **Vorwärts** nicht identisch mit den Befehlen **Rückgängig (Strg+Z)** und **Wiederholen (Strg+Y)** sind. Es handelt sich hierbei um unterschiedliche Ausführungsschritte. Durch einmaliges Klicken auf **Zurück** gelangen Sie zur zuvor angezeigten Komponente, wie sie zuvor angezeigt wurde. Durch einmaliges Klicken auf **Rückgängig** wird die letzte Bearbeitung rückgängig gemacht, unabhängig davon, wann diese Änderung erfolgt ist.

Zusätzliche Anmerkungen

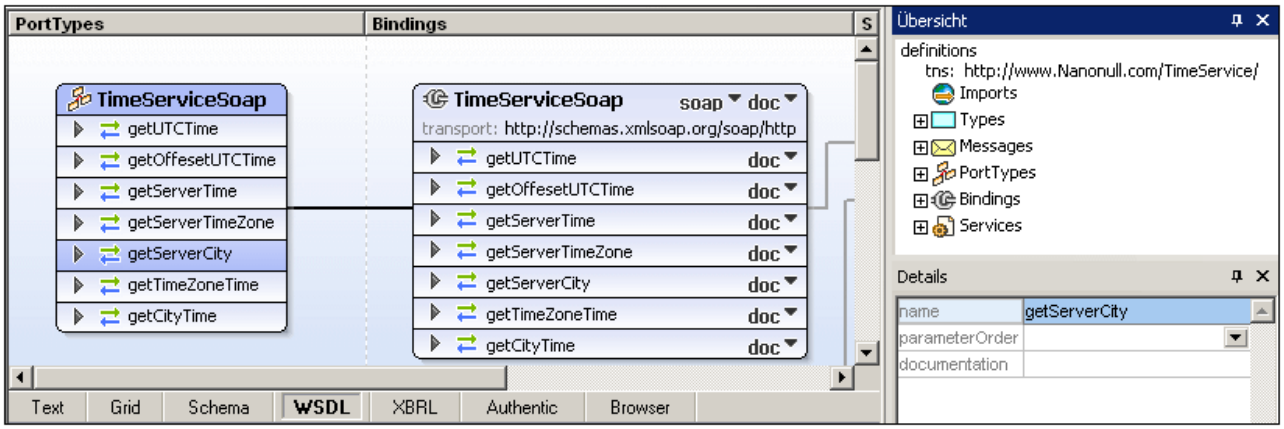
Beachten Sie bitte die folgenden Punkte:

- Mit der Schaltfläche **Zurück** können Sie die vorherigen 500 Positionen erneut anzeigen.
- Die Funktion Zurück/Vorwärts ist schemaübergreifend aktiv. Wenn ein Schema seither geschlossen wurde oder derzeit in einer anderen Ansicht geöffnet ist, wird es in der Schema-Ansicht geöffnet bzw. wird in die Schema-Ansicht gewechselt.
- Wenn eine Komponente, die in einer vorherigen Position angezeigt wurde, gelöscht wird, dann kann diese Komponente nicht angezeigt werden. Wenn eine solche Komponente Teil einer vorherigen Position war, wird diese Position ohne die gelöschte Komponente angezeigt. Wenn die Komponenten die gesamte Position umfasste, steht die gesamte Position nicht mehr zur Verfügung. In diesem Fall gelangen Sie durch Klicken auf die Schaltfläche **Zurück** zu der Position vor der nicht mehr verfügbaren Position.

© 2016-2022 Altova GmbH

4.5 WSDL-Ansicht

In der WSDL-Ansicht (*Abbildung unten*) sehen Sie eine Benutzeroberfläche zum grafischen Editieren von WSDL 1.1- und WSDL2.0-Dokumenten. Die WSDL-Ansicht steht zur Verfügung, wenn ein WSDL-Dokument aktiv ist und auf das Register "WSDL-Ansicht" geklickt wird. Die Struktur und Komponenten eines WSDL-Dokuments werden mit Hilfe von grafischen Mitteln im Hauptfenster[254] erstellt. Über die Eingabehilfen[258] stehen zusätzliche Editiermöglichkeiten zur Verfügung.



Das Hauptfenster[254] (bestehend aus den Abschnitten PortTypes (WSDL 1.1) oder Interfaces (WSDL 2.0), Bindings und Services) und die Eingabehilfen[258] (Übersicht[258] und Details[265]) werden in den Unterabschnitten dieses Abschnitts näher beschrieben. (Eine Beschreibung zum Arbeiten mit Projekten finden Sie im Abschnitt Menübefehle unter Menü "Projekt"[1316].)

Funktionen in der WSDL-Ansicht

In der WSDL-Ansicht stehen die folgenden Funktionen zur Verfügung:

- Grafische Anzeige aller WSDL-Elemente im Hauptfenster, gruppiert nach PortTypes (WSDL 1.1) oder Interfaces (WSDL 2.0), Bindings und Services.
- Direkte Bearbeitung von WSDL-Elementen mittels Drag & Drop.
- Die Möglichkeit, WSDL-Elemente, die in der grafischen Ansicht zu sehen sind, hinzuzufügen, anzuhängen und zu löschen (kontextsensitives Menü).
- Die Möglichkeit, Werte in der Details-Eingabehilfe einzugeben und zu bearbeiten.
- WSDL-Validierung gegen einen W3C Working Draft.
- Import oder Einbettung von XML-Schemas in das WSDL-Dokument
- Wechseln in die Schema-Ansicht zur Bearbeitung von Schemas
- Bearbeiten von Schema-Typen in der WSDL-Ansicht.
- Generierung von WSDL-Dokumentation in MS Word oder HTML.
- Generierung eines Diagramms (PNG-Bild) des WSDL-Dokuments im Hauptfenster.
- Drucken der Ansicht im WSDL-Fenster

Anzeige von Dateien

Beachten Sie die folgenden Punkte:

- Wenn Sie eine WSDL-Datei öffnen, wird diese automatisch in der WSLD-Ansicht geöffnet.

- Sie können ein WSDL-Dokument auch in der Text-Ansicht und der Enhanced Grid-Ansicht öffnen. Klicken Sie dazu auf das entsprechende Register.
- Wenn die WSDL-Datei eine Referenz zu einem XML-Schema enthält, kann das Schema durch Auswahl des Befehls **WSDL | Types | Schema in Schema-Ansicht bearbeiten** angezeigt und bearbeitet werden. Daraufhin wird die Schema-Datei in der Schema-Ansicht geöffnet.
- Wenn eine mit der Datei verknüpfte Schema-Datei geöffnet ist, können Sie die Ansicht der WSDL-Datei nicht wechseln (z.B. von der WSDL-Ansicht in die Text-Ansicht). Stellen Sie daher sicher, dass Sie die Änderungen an der Schema-Datei gespeichert und die Datei geschlossen haben, bevor Sie die Ansicht für die WSDL-Datei wechseln.

Es gibt zwei Eingabehilfen zum Bearbeiten von WSDL-Dokumenten: Übersicht[258] und Details[265]. Beide Eingabehilfen können durch Doppelklick auf die Titelleiste an- und abgedockt werden. In angedocktem Zustand kann die Funktion "Automatisch ausblenden" aktiviert werden, indem Sie auf das Pin-Symbol in der Titelleiste klicken. Wenn die Eingabehilfe automatisch ausgeblendet wird, wird sie minimiert als Register am Rand des Applikationsfensters angezeigt. Um eine automatisch ausgeblendete Eingabehilfe wieder anzudocken, platzieren Sie den Mauszeiger über ihr Register, um sie auszurollen und klicken Sie in der Titelleiste auf das Pin-Symbol.

Siehe auch: Weitere Informationen zur Arbeit mit WSDL-Dokumenten finden Sie in den Abschnitten WSDL Tutorial[789] und Menübefehle | Menü "WSDL"[1509].

4.5.1 Hauptfenster

Im Hauptfenster wird das WSDL-Dokument bearbeitet. Es ist vertikal in drei Abschnitte unterteilt: (i) Port Types (WSDL 1.1) oder Interfaces (WSDL 2.0)[255], (ii) Bindings[256] und (iii) Services[257]. Die Beziehung zwischen einem PortType und einem Binding und einem Binding und einem Service wird jeweils durch Konnektorlinien angezeigt. Jeder dieser drei Abschnitte wird weiter unten ausführlich beschrieben.

Symbole im Hauptfenster

Die folgenden Symbole werden im Hauptfenster verwendet:

 PortType in WSDL 1.1, Interface in WSDL 2.0

 Binding

 Service

 Fault-Element

 Operation: Der grüne Pfeil steht für die Inputs, der blaue für die Outputs. Je nach Art der Operation wird das entsprechende Symbol verwendet.

 Message

 Message Part (Parameter)

 XSD-Element

 XSD simpleType oder complexType

 Port

Hinzufügen neuer Port Types, Interfaces, Bindings und Services

Um einen neuen Port Type (in WSDL 1.1-Dokumenten), ein neues Interface (in WSDL 2.0-Dokumenten), ein Binding oder Service hinzuzufügen, rechtsklicken Sie im Hauptfenster außerhalb eines Komponentenkastens auf eine beliebige Stelle und wählen Sie im Kontextmenü nun den entsprechenden Befehl aus.

© 2016-2022 Altova GmbH

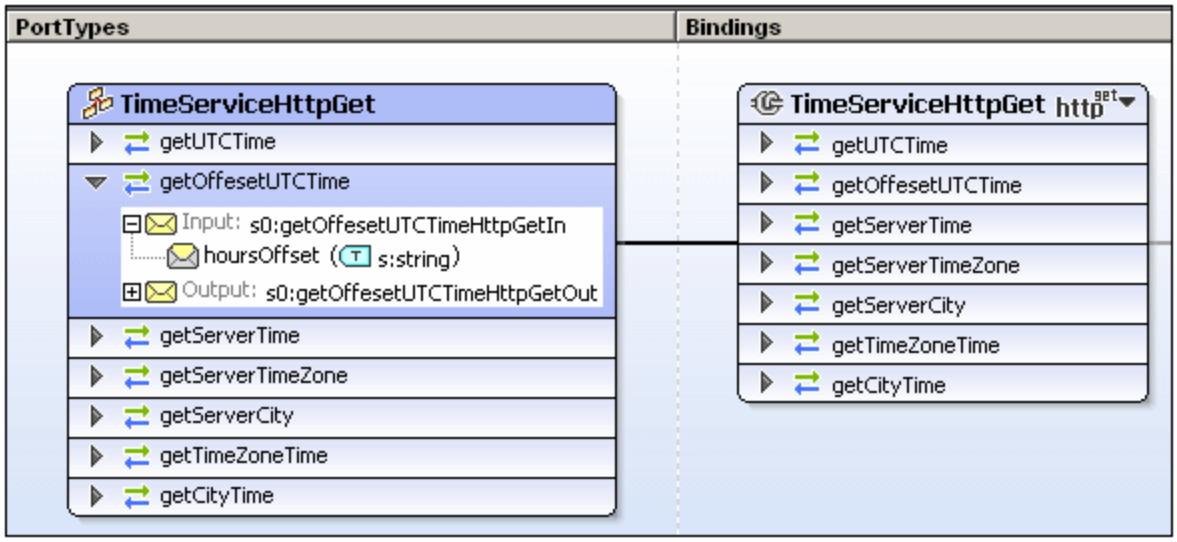
Drag-and-Drop-Funktionen

Es stehen die folgenden Drag-and-Drop-Funktionen zur Verfügung:

- Im Hauptfenster können mittels Drag-and-Drop Verknüpfungen zwischen PortTypes (WSDL 1.1) oder Interfaces (WSDL 2.0) und Bindings und Services erstellt werden.
- In WSDL 2.0-Dokumenten können Sie Elemente in der Eingabehilfe "Übersicht" auf Interface Faults sowohl im Hauptfenster als auch in der Eingabehilfe "Übersicht" ziehen.

PortTypes (WSDL 1.1), Interfaces (WSDL 2.0)

Der Abschnitt "PortTypes" (WSDL 1.1) enthält alle PortTypes, die im WSDL-Dokument definiert sind (*in der Abbildung unten sehen Sie nur einen portType im Abschnitt PortTypes*). Der Abschnitt "Interfaces" (in WSDL 2.0-Dokumenten) enthält alle im WSDL-Dokument definierten Interfaces.



Jeder PortType bzw. jedes Interface wird grafisch in Form eines Kästchens dargestellt, das die Operationen (Operations) für diesen PortType bzw. dieses Interface enthält. Komponenten können direkt im Kästchen bearbeitet werden. Die Hauptcharakteristiken von PortType- und Interface-Kästchen sind im Folgenden aufgelistet:

- Operationen können erweitert angezeigt werden, sodass Sie deren Messages sehen. Kicken Sie dazu auf das Symbol links vom Operationsnamen.
- In WSDL 1.1 kann eine Message einen Message Part enthalten. Eine solche Message kann erweitert werden, sodass der Message Part zu sehen ist.
- Wenn Sie mit der rechten Maustaste auf ein PortType-Kästchen klicken (entweder auf PortType, Operation, Message oder Message Part), erscheint ein Kontextmenü, in dem Sie die entsprechende Aktion auswählen können. Wenn Sie z.B. mit der rechten Maustaste auf einen PortType-Namen klicken, können Sie unter anderem einen neuen PortType oder eine Operation an den ausgewählten PortType anhängen oder ein Binding für den ausgewählten PortType erstellen.
- Die optionalen WSDL 2.0 Interface-Eigenschaften *extends*, *styleDefault* und *documentation* werden ausgeblendet, wenn sie leer sind. Sie können im Kontextmenü des Interface über den Befehl **Bearbeiten** bearbeitet werden.

- Eigenschaften von Operationen in WSDL 2.0-Dokumenten können über das Kontextmenü der Operation mit dem Befehl **Bearbeiten** bearbeitet werden.
- Beachten Sie: Wenn eine Komponente ausgewählt ist, können ihre Details in der Eingabehilfe "Details" bearbeitet werden.
- Die Dokumentation zu PortTypes und Interfaces wird am unteren Rand der einzelnen Kästchen angezeigt.

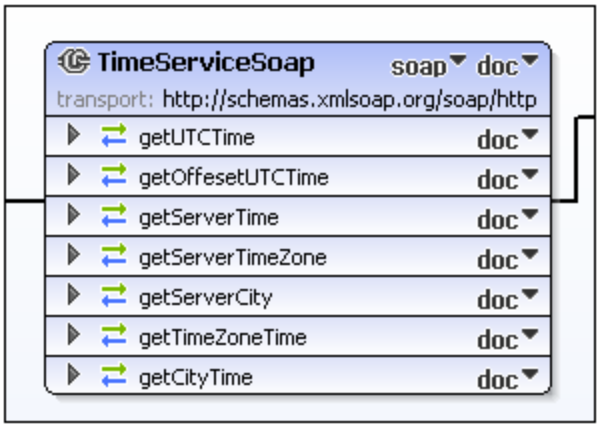
Die Verknüpfung eines PortType oder eines Interface mit einem Binding wird im Hauptfenster durch eine schwarze Konnektorlinie angezeigt, die das PortType- oder Interface-Kästchen mit dem Binding-Kästchen verbindet; das Binding-Kästchen befindet sich im Hauptfenster im Abschnitt "Bindings".

Bindings

In einem Binding sind das Message-Format und die Protokoll-Informationen für Folgendes definiert:

- Operationen, die durch einen bestimmten PortType definiert sind (WSDL 1.1)
- Operationen und Fault-Elemente, die durch ein bestimmtes Interface definiert sind (WSDL 2.0)

In WSDL 1.1 können Bindings für SOAP 1.1- oder SOAP 1.2-Endpunkte oder für HTTP 1.1 GET und POST Verben erstellt werden. In WSDL 2.0 können Bindings für SOAP 1.1- oder SOAP 1.2-Endpunkte oder für HTTP erstellt werden. Jedes Binding wird im Hauptfenster im Abschnitt "Bindings" durch ein Binding-Kästchen dargestellt (*Abbildung unten*). Das Binding-Kästchen enthält alle Operationen und/oder Fault-Elemente des damit verknüpften PortType bzw. Interface (*siehe Abbildung unten*).



Ein Binding kann auf eine der folgenden Arten mit einem Port Type oder Interface verknüpft werden:

- Rechtsklicken Sie auf einen Port Type oder ein Interface und wählen Sie den Befehl **Binding für portType erstellen** bzw. **Binding für Interface erstellen**.
- Rechtsklicken Sie auf ein WSDL 1.1-Binding und bearbeiten Sie die Eigenschaft *PortType*.
- Rechtsklicken Sie auf ein WSDL 2.0-Binding und wählen Sie den Befehl **Bearbeiten | Interface**.

Um das Binding zu definieren, wählen Sie in der ersten Auswahlliste rechts vom Namen des Binding (*Abbildung unten*) das gewünschte Protokoll aus. In WSDL ist das entweder `soap 1.1`, soap 1.2, `http-get` oder `http-post` aus, um die Art des Binding zu definieren. Bei Auswahl von eines SOAP-Protokolls können Sie (in der zweiten Auswahlliste) zusätzlich definieren, ob der Stil `doc` oder `rpc` sein soll. In WSDL 2.0-Dokumenten kann die Eigenschaft hinzugefügt oder über den Befehl **Bearbeiten** aus dem Kontextmenü des Bindings bearbeitet werden.

© 2016-2022 Altova GmbH



In WSDL 1.1 werden MIME Encodings (auch als MIME-Bindings bezeichnet) auf Message-Ebene definiert. Klicken Sie zum Definieren eines MIME Encoding mit der rechten Maustaste auf die Message (*Abbildung unten*) und hängen Sie die entsprechende MIME-Definition an. In der Abbildung unten wurden MIME-Definitionen für die Message `Output` erstellt.



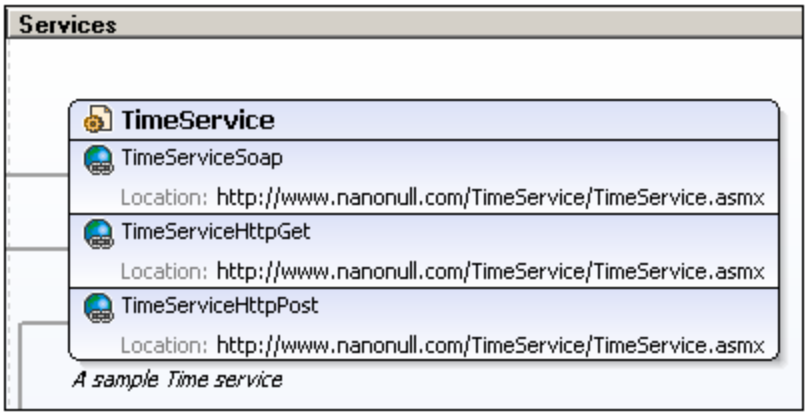
Wenn Sie mit der rechten Maustaste auf einen Eintrag in einem Binding-Kästchen klicken, wird ein kontextsensitives Menü angezeigt. Bindings können angehängt oder gelöscht werden; es können Extensibility-Elemente bearbeitet werden und Messages können definiert werden. Beachten Sie außerdem: Wenn ein Binding-Kästchen oder ein Eintrag in einem Binding-Kästchen ausgewählt ist, werden die Definitionen in der Eingabehilfe "Details" angezeigt und können dort bearbeitet werden.

Sie können für ein Binding einen Port erstellen. Klicken Sie dazu mit der rechten Maustaste auf die Titelleiste eines Binding-Kästchens und wählen Sie den Befehl **Port für Binding erstellen** (WSDL 1.1-Dokumente) bzw. **Endpoint für Binding erstellen** (WSDL 2.0-Dokumente). Der dazugehörige Port wird (im Hauptfenster im Abschnitt "Services") in einem Service-Kästchen erstellt. Die Verknüpfung zwischen einem Binding und einem Port wird durch eine schwarze Konnektorlinie angezeigt.

Dokumentation zu Bindings wird am unteren Rand der einzelnen Binding-Kästchen angezeigt.

Services

In einem Service sind eine Gruppe von miteinander in Zusammenhang stehenden Ports (WSDL 1.1) oder Endpoints (WSDL 2.0) zusammengruppiert. Es wird im Hauptfenster im Abschnitt "Services" in Form eines Service-Kästchens dargestellt (*Abbildung unten*. Jedes Service-Kästchen besteht aus einer oder mehreren Port- oder Endpoint-Deklarationen (*siehe Abbildung unten*).



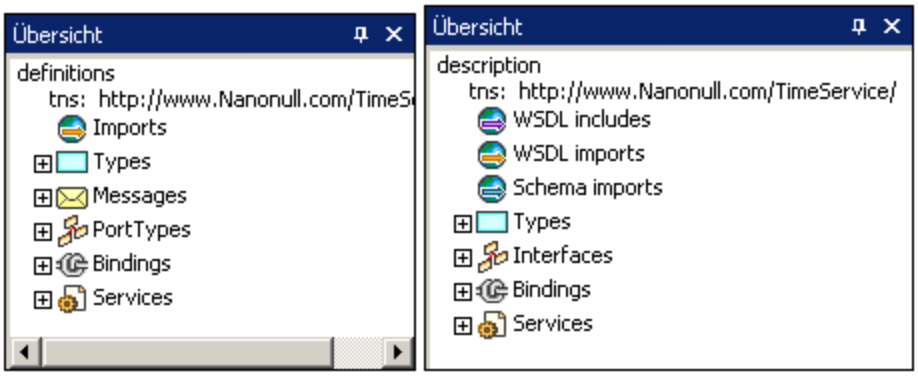
Der Name des Service, des Port oder ses Endpoint, das mit dem Port oder Endpoint verknüpfte Binding und die Adressinformationen zu einem Port oder Endpoint können direkt im Service-Kästchen oder in der Eingabehilfe "Details" bearbeitet werden. Wenn Sie mit der rechten Maustaste auf ein Service-Kästchen oder einen bestimmten Eintrag im Service-Kästchen klicken, wird ein Kontextmenü mit Befehlen zu diesem Service oder Eintrag geöffnet.

Dokumentation zu Sercices wird am unteren Rand der einzelnen Service-Kästchen angezeigt.

4.5.2 Eingabehilfe Übersicht

In der **Eingabehilfe "Übersicht"** (*Abbildung unten*) sehen Sie eine Übersicht über das WSDL-Dokument, wobei die verschiedenen Komponenten des Dokuments in Strukturkategorien gruppiert sind und die Target Namespaces, die importierten Schemas und die inkludierten/importierten WSDL-Dokumente aufgelistet werden. Zusätzlich zu PortTypes (oder Interfaces in WSDL 2.0), Messages (WSDL 1.1), Bindings und Services werden auch die verschiedenen im Dokument definierten Typen aufgelistet.

Außerdem können Sie in der Eingabehilfe "Übersicht" importierte und inkludierte XML-Schemas und WSDL-Dateien verwalten.



Eingabehilfe "Übersicht" in WSDL 1.1 (links) und WSDL 2.0 (rechts)

© 2016-2022 Altova GmbH

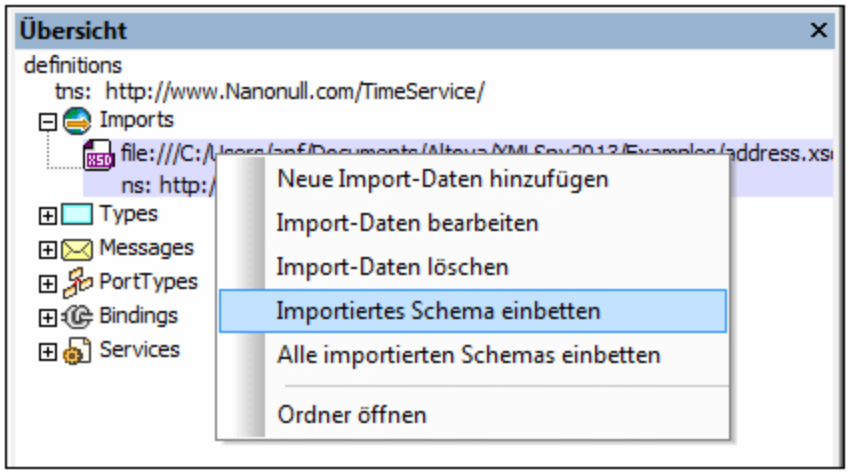
In jeder Kategorie werden die Komponenten in einer Baumstruktur angezeigt. Sie können die Einträge in der Baumstruktur erweitern und reduzieren um deren Inhalt anzuzeigen bzw. auszublenden. Wenn eine Komponente in der Eingabehilfe "Übersicht" ausgewählt wird, werden die Komponente und ihre Eigenschaften in der Eingabehilfe "Details" 265 angezeigt, wo die Eigenschaften bearbeitet werden können. Die Namen von WSDL- und Schema-Komponenten, die in der Baumstruktur angezeigt werden, können auch direkt in dieser Struktur bearbeitet werden. Extern definierte Komponenten, also Komponenten in inkludierten oder importierten WSDL-Dokumenten oder Schemas werden grau angezeigt und können nicht bearbeitet werden. Die einzelnen Kategorien in der Eingabehilfe "Übersicht" werden im Folgenden erläutert.

Target Namespace (WSDL 1.1 und 2.0)

Wird in der Baumstruktur durch tns bezeichnet. Der Target Namespace kann in der Eingabehilfe "Übersicht" bearbeitet werden. Alle anderen Namespaces müssen in der Text-Ansicht bearbeitet werden.,

Imports (WSDL 1.1)

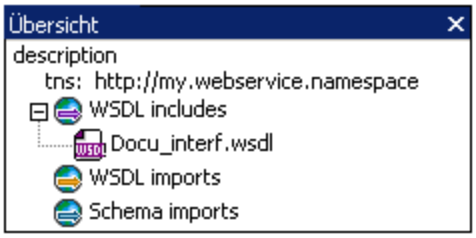
XML-Schema (XSD)-Dateien und WSDL-Dateien können in das aktive Dokument importiert werden. Um ein XML-Schema oder eine WSDL-Datei zu importieren, rechtsklicken Sie auf den Eintrag "Imports" oder eine bereits importierte Datei in der Liste "Imports" und wählen Sie den Befehl **Neue Import-Daten hinzufügen**. Wenn Sie mit der rechten Maustaste in der Liste "Imports" auf eine bereits importierte Datei klicken, erscheint ein Kontextmenü, in dem Sie wählen können, ob Sie eine neue importierte Datei hinzufügen möchten, die importierte Datei durch eine andere Datei ersetzen möchten (**Importdaten bearbeiten**) oder die importierte Datei löschen möchten (**Import-Daten löschen**). Außerdem können Sie die Datei von dem angegebenen Pfad aus auch öffnen. Die Datei wird in der WSDL- (`.wsdl` Datei) oder der Schema-Ansicht (`.xsd` Datei), geöffnet und kann dort editiert werden.



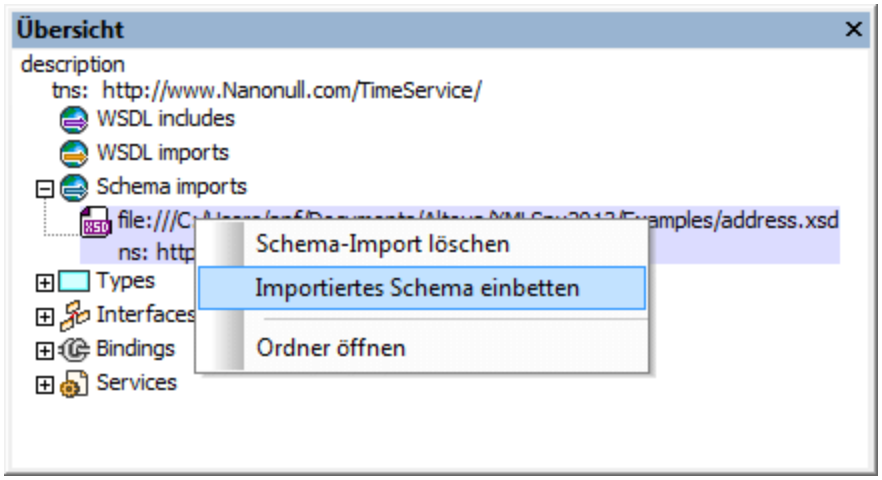
Im Anschluss daran kann ein importiertes XML-Schema in die WSDL-Datei eingebettet werden. Beim Einbetten des importierten Schemas wird das Schema als Inline-Schema im Element `types` erstellt und das `import` Element wird entfernt. Um ein importiertes Schema einzubetten, klicken Sie mit der rechten Maustaste in der Liste *Imports* auf den Eintrag für das Schema und wählen Sie den Befehl **Importiertes Schema einbetten** oder **Alle importierten Schemas einbetten.** Der zweite Befehl, der auf alle importierten Schemas angewendet wird, ist auch im Kontextmenü des Eintrags *Imports* aktiviert.

WSDL Includes, WSDL Imports, Schema Imports (WSDL 2.0)

XML-Schema (XSD)-Dateien können importiert und WSDL-Dateien können in das aktive WSDL-Dokument inkludiert oder importiert werden. Um eine Datei zu inkludieren oder zu importieren, klicken Sie mit der rechten Maustaste auf das gewünschte Element, wählen Sie den gewünschten Befehl aus (*WSDL includes, WSDL imports, Schema imports*) und fügen Sie die Datei hinzu. Der Namespace einer importierten Datei wird automatisch anhand des Target Namespace der importierten Datei generiert.



Wenn Sie mit der rechten Maustaste auf eine inkludierte oder importierte Datei klicken, erscheint ein Kontextmenü, in dem Sie die Datei entweder löschen oder in XMLSpy öffnen können. Die Datei wird in der WSDL- (`.wsdl` Datei) oder der Schema-Ansicht (`.xsd` Datei), geöffnet und kann dort editiert werden. Ein importiertes XML-Schema kann anschließend in die WSDL-Datei eingebettet werden (*siehe Abbildung unten*).



Beim Einbetten des importierten Schemas wird das Schema als Inline-Schema im Element `types` erstellt und das `import` Element wird entfernt. Um ein importiertes Schema einzubetten, klicken Sie mit der rechten Maustaste in der Liste *Imports* auf den Eintrag für das Schema und wählen Sie den Befehl **Importiertes Schema einbetten** oder **Alle importierten Schemas einbetten.** Der zweite Befehl, der auf alle importierten Schemas angewendet wird, ist auch im Kontextmenü des Eintrags *Imports* aktiviert.

Types (WSDL 1.1 und 2.0)

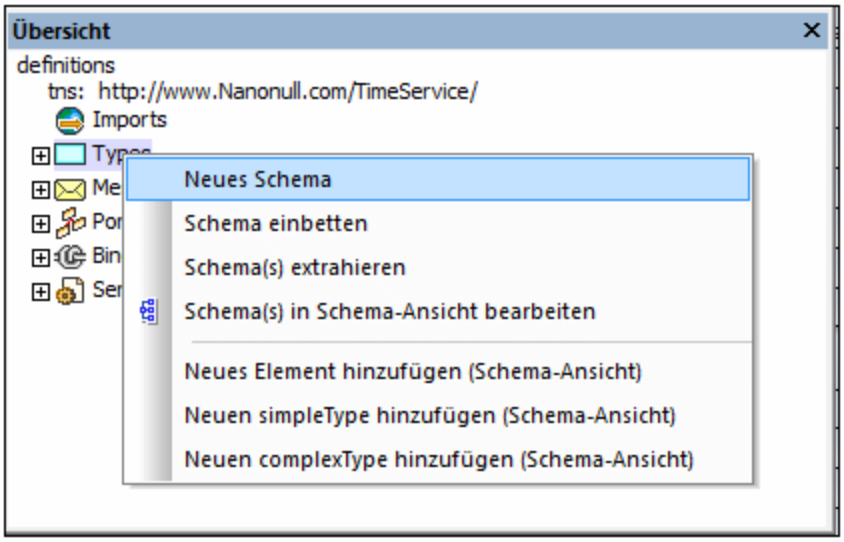
Hier sind alle im WSDL-Dokument (schwarz) und in allen importierten Schemas oder importierten WSDL-Dokumenten (grau) definierten Typen aufgelistet.

© 2016-2022 Altova GmbH



Es stehen die folgenden Funktionen zur Verfügung.

- *Neues Schema erstellen:* Klicken Sie mit der rechten Maustaste auf den Eintrag *Types* und wählen Sie den Befehl **Neues Schema** (*siehe Abbildung unten*). Daraufhin wird in der WSDL-Datei ein neues leeres eingebettetes Schema erstellt und alle eingebetteten Schemas einschließlich des neuen werden in der Schema-Ansicht geöffnet. Nachdem Sie das Schema-Dokument bearbeitet haben, werden die Änderungen beim Speichern der Schema-Datei in das WSDL-Dokument geschrieben. Sie können das Schema-Dokument anschließend schließen. Diese Funktion eignet sich, wenn Sie im WSDL-Dokument ein neues eingebettetes Schema erstellen möchten.



- *Einbetten von Schemas:* Klicken Sie mit der rechten Maustaste auf das Element *Types* und wählen Sie den Befehl **Schema einbetten**. Daraufhin wird ein Dialogfeld "Datei öffnen" angezeigt, in dem Sie zur gewünschten Schema-Datei navigieren können. Wenn Sie auf **OK** klicken, wird das Schema innerhalb des Elements `types` als Inline-Element erstellt. Wenn das ausgewählte Schema bereits importiert wurde, werden Sie gefragt, ob das bereits importierte Schema eingebettet werden soll.

- *Extrahieren von Schemas:* Klicken Sie mit der rechten Maustaste auf das Element *Types* und wählen Sie den Befehl **Schema(s) extrahieren**. Jedes der eingebetteten Schemas wird als temporäre Datei in der Schema-Ansicht geöffnet und für jede Datei wird ein Dialogfeld "Speichern unter" angezeigt. Wenn Sie die Schema-Datei speichern, wird das Schema extrahiert, unter dem angegebenen Pfad gespeichert und anschließend in die WSDL-Datei importiert. Die Schema-Datei ist dann nicht mehr als Inline-Schema sondern als externes, importiertes Schema vorhanden.

- *Bearbeiten von Schemas:* Sie können eingebettete Schemas in der Schema-Ansicht bearbeiten. Klicken Sie dazu mit der rechten Maustaste entweder auf das Element *Types* oder auf den Namen der Schemakomponente in der Liste *Types* und wählen Sie dann den Befehl **Schema(s) bearbeiten** bzw. **Schema bearbeiten.** Daraufhin wird anhand der `types` Definitionen im WSDL-Dokument eine temporäre XSL-Datei on-the-fly generiert. Dieses XSD-Dokument wird in der Schema-Ansicht angezeigt und kann bearbeitet werden. Nachdem Sie die Bearbeitung des XSD-Dokuments abgeschlossen haben, werden die Änderungen beim Speichern wieder in den `types` Definitionen im WSDL-Dokument gespeichert. Wenn Sie das XSD-Dokument schließen, ohne die Änderungen zu speichern, werden die `types` Definitionen im WSDL-Dokument nicht geändert.

- *Hinzufügen von Schemakomponenten:* Sie können ein XML-Schema-Element (WSDL 1.1 und 2.0), einen simpleType (WSDL 1.1) oder einen complexType (WSDL 1.1) hinzufügen. Klicken Sie dazu mit der rechten Maustaste entweder auf das Element *Types* oder auf den Namen einer Schemakomponente in der Liste *Types* und wählen Sie den entsprechenden **Hinzufügen** Befehl aus. Daraufhin wird anhand der `types`-Definition im WSDL-Dokument on-the-fly eine temporäre XSD-Datei generiert und in der Schema-Ansicht angezeigt. Diese Datei enthält die neue Komponente, die unbenannt ist. Sie können diese Komponente anschließend bearbeiten. Wenn Sie die Datei speichern, wird die neue Komponente in die `types`-Definitionen des WSDL-Dokuments geschrieben.

- *Löschen von Schemakomponenten:* Um eine Schemakomponente zu löschen, klicken Sie in dder Liste *Types* mit der rechten Maustaste darauf und wählen Sie im Kontextmenü den Befehl **Löschen.**

Messages (WSDL 1.1)

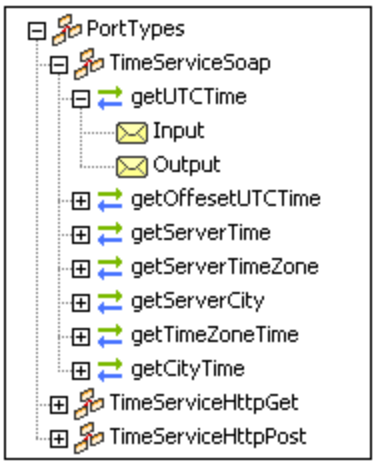
Wenn eine Message oder ihre Unterkomponente ausgewählt ist, werden die Eigenschaften der Message bzw. Unterkomponente in der Eingabehilfe "Details" angezeigt, wo sie auch bearbeitet werden können. Zusätzlich dazu haben Sie im Kontextmenü die folgenden Auswahlmöglichkeiten:

- Wenn eine Message in der Eingabehilfe "Übersicht" ausgewählt ist, können Sie zusätzlich einen Message Part zu der Message hinzufügen, die Message löschen oder eine neue Message hinzufügen.
- Wenn in der Eingabehilfe "Übersicht" ein Message Part ausgewählt ist, können Sie einen weiteren Message Part zu dieser Message hinzufügen oder den ausgewählten Message Part löschen.
- Mit dem Befehl "Synchronisieren" wird die ausgewählte Message bzw. der ausgewählte Message Part im jeweiligen PortType-Kästchen markiert.

PortTypes (WSDL 1.1)

Bei portTypes stehen die folgenden Funktionalitäten über das Kontextmenü zur Verfügung:

© 2016-2022 Altova GmbH

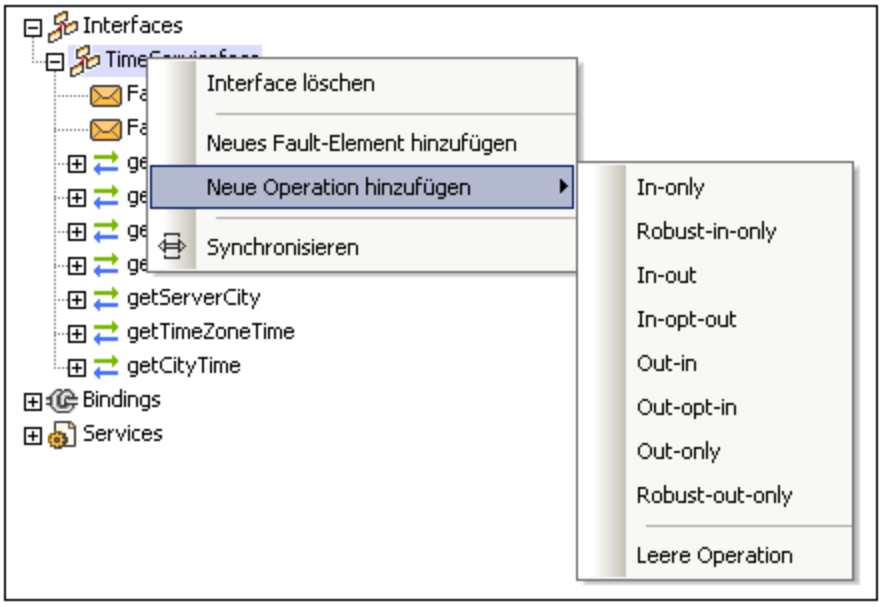


- Wenn der Eintrag *PortTypes* ausgewählt ist, können PortTypes hinzugefügt werden
- Wenn ein PortTyp ausgewählt ist, können PortTypes hinzugefügt, der ausgewählte PortType gelöscht und Operationen zum ausgewählten PortType hinzugefügt werden.
- Wenn eine Operation ausgewählt ist, können zusätzliche Operationen angehängt werden, die ausgewählte Operation kann gelöscht werden und Elemente (input, output, fault) können zur ausgewählten Operation hinzugefügt werden.
- Wenn ein Message-Element (input, output, fault) ausgewählt ist, können zusätzliche Messages hinzugefügt werden und die ausgewählte Message kann gelöscht werden.
- Mit dem Befehl "Synchronisieren" wird der ausgewählte PortType, die ausgewählte Operation bzw. die ausgewählte Message markiert.

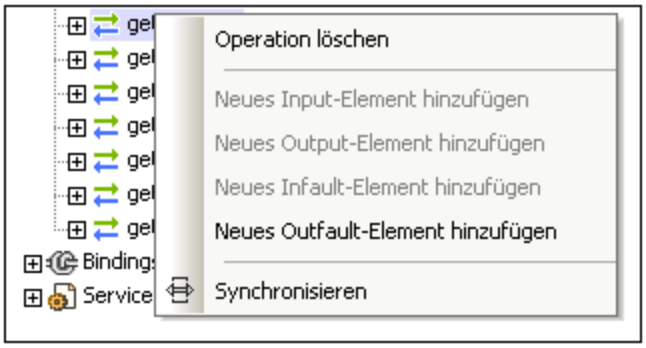
Interfaces (WSDL 2.0)

Interfaces können über das Kontextmenü verwaltet werden.

- Um ein Interface hinzuzufügen, rechtsklicken Sie auf das Element *Interfaces* und wählen Sie den Menübefehl **Neues Interface hinzufügen**.
- Bei Rechtsklick auf ein Interface wird ein Menü (*siehe Abbildung unten*) mit Befehlen angezeigt, mit denen das ausgewählte Interface gelöscht und Fault-Elemente und Operationen zur Definition des ausgewählten Interface hinzugefügt werden können. Welche Art von Operation hinzugefügt werden soll, kann im Untermenü des Befehls **Neue Operation hinzufügen** definiert werden. Zu allen Bindings, die das Interface referenzieren, wird eine entsprechende Binding-Operation hinzugefügt. Wenn eine Operation gelöscht wird, werden auch die referenzierenden Binding-Operationen gelöscht.



- Bei Rechtsklick auf eine Operation werden Befehle angezeigt (*siehe Abbildung unten*), mit denen die ausgewählte Operation gelöscht und Elemente (wie infault und outfault) zur Operation hinzugefügt werden können.



- Bei Rechtsklick auf eine Message erscheint ein Menü, über das Sie die ausgewählte Message löschen können.
- Wenn Sie den Befehl **Synchronisieren** auswählen, wird das ausgewählte Interface, die ausgewählte Operation oder Message im Design markiert.

Bindings (WSDL 1.1 und 2.0)

Wenn ein Binding ausgewählt ist, können zusätzliche Bindings an die bereits vorhandenen Bindings angehängt werden, das ausgewählte Binding kann gelöscht werden und Operationen können für das ausgewählte Binding eingefügt werden. Wenn eine Operation oder eine Message ausgewählt ist, stehen dieselben Optionen wie für Operationen und Messages in der Kategorie "PortTypes" (WSDL 1.1) oder Interfaces (WSDL 2.0) zur Verfügung. Wenn Sie auf den Befehl "Synchronisieren" klicken, wird das ausgewählte Binding, die ausgewählte Operation oder Message markiert.

© 2016-2022 Altova GmbH

Services (WSDL 1.1 und 2.0)

Wenn ein Service ausgewählt ist, können zusätzliche Services hinzugefügt werden, der ausgewählte Service kann gelöscht werden und Ports können für den ausgewählten Service hinzugefügt werden. Wenn Sie auf den Befehl "Synchronisieren" klicken, wird der ausgewählte Service bzw. der ausgewählte Port markiert.

4.5.3 Eingabehilfe Details

In der Eingabehilfe "Details" werden die Eigenschaften des im Hauptfenster oder in der Eingabehilfe "Übersicht" ausgewählten Eintrags angezeigt (*Abbildung unten*). Diese Eigenschaften können in der Eingabehilfe "Details" bearbeitet werden.



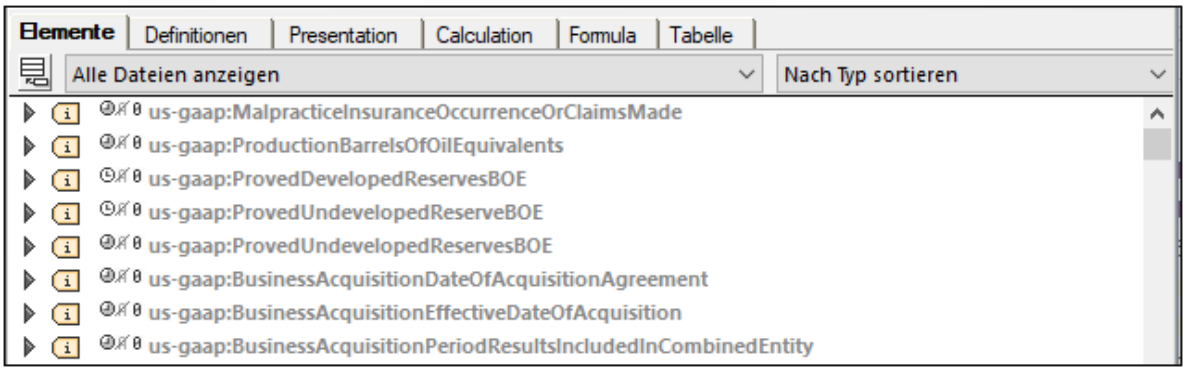
Wenn z.B., wie in der Abbildung oben gezeigt, der Port `TimeServiceSoap` (des `TimeService` Service) ausgewählt wird, so werden die Eigenschaften von `TimeServiceSoap` in der Eingabehilfe "Details" angezeigt, wo sie auch bearbeitet werden können. Um ein Textfeld wie z.B. `name` oder `description` zu bearbeiten, doppelklicken Sie in das Feld und bearbeiten Sie den Text. Bei einigen Eigenschaften wie z.B. `binding` in der Abbildung oben, wo man aus mehreren Optionen auswählen kann, können Sie die gewünschte Option aus einer Auswahlliste auswählen.

4.6 XBRL-Ansicht

Die XBRL-Ansicht ist eine grafische Benutzeroberfläche, über die Sie XBRL-Taxonomien bearbeiten können. Er bietet Funktionen zur schnellen Validierung sowie Fehlermeldungen, die dem Benutzer beim schnellen Erstellen korrekter Taxonomien helfen.

Die XBRL-Ansicht besteht aus den folgenden Teilen:

- einem Hauptfenster mit sechs Registern: Elemente[266], Definitionen[271], Presentation[271], Calculation[271], Formula[271], Tabelle[271] (*Abbildung unten*). Die wichtigsten Funktionen dieser Register werden in den Unterabschnitten dieses Abschnitts beschrieben.



- drei umfangreichen Eingabehilfen[274]: Übersicht, globale Elemente, Details. Mit Hilfe dieser Eingabehilfen können Sie Taxonomiedateien verwalten und die Taxonomie im Hauptfenster bearbeiten.
- einem Fenster "Meldungen"[122], in dem Meldungen über die Gültigkeit der Taxonomie angezeigt werden.

Dieser Abschnitt enthält eine Beschreibung des Hauptfensters und der Eingabehilfe der XBRL-Ansicht sowie Anleitungen zu deren Verwendung. Nähere Informationen zu diesem Thema finden Sie im Abschnitt XBRL[829]. Eine Beschreibung der Befehle finden Sie unter Menü "XBRL"[1533].

Weitere Funktionen der XBRL-Ansicht

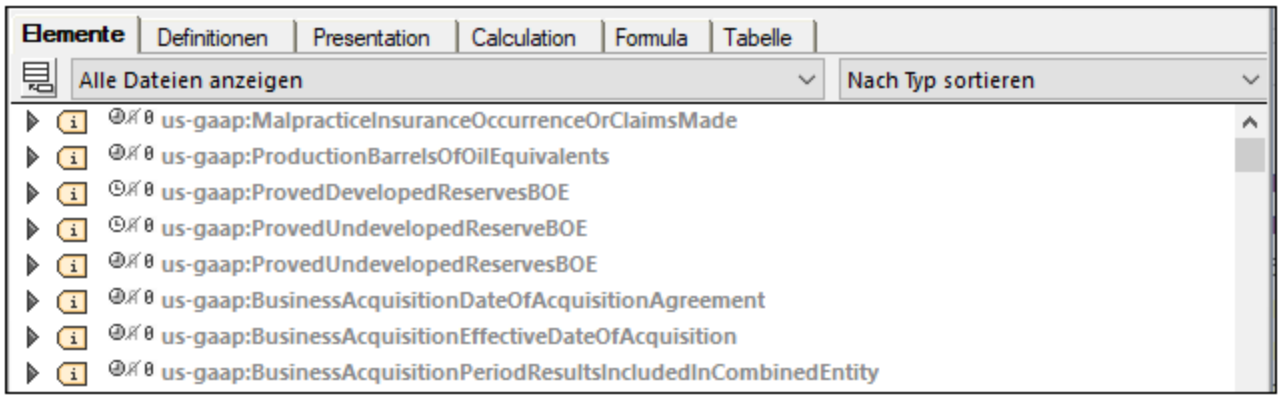
Neben den in diesem Abschnitt beschriebenen Bearbeitungsfunktionen stehen die folgenden nützlichen Funktionen zur Verfügung:

- Dokumentation generieren[1541]: Mit diesem Befehl wird ausführliche Dokumentation in den Formaten HTML, Word und RTF generiert.
- Druck der aktuellen Ansicht[1293] dient zum Ausdrucken der aktuellen Ansicht mit Hilfe des XMLSpy Befehls **Datei | Drucken**.

4.6.1 Hauptfenster: Register "Elemente"

Auf dem Register **Elemente** des Hauptfensters (*Abbildung unten*) werden die Concepts der Taxonomie angezeigt, darunter standardmäßig auch die in importierten Taxonomien enthaltenen Concepts. Concepts in der aktuellen Taxonomie werden schwarz angezeigt; Concepts in importierten Taxonomien werden grau angezeigt.

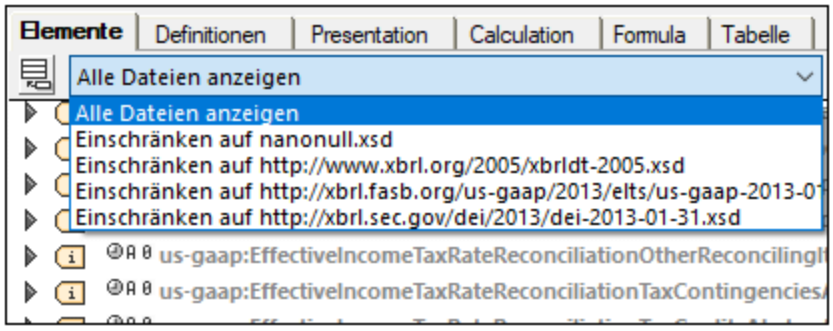
© 2016-2022 Altova GmbH



Zusätzliche Informationen finden Sie im Abschnitt XBRL[829] und unter der Beschreibung der Befehle im Menü "XBRL"[1533].

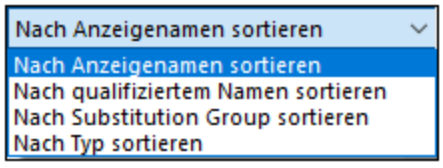
Auswählen der anzuzeigenden Taxonomie

In der Auswahlliste *Datei* unterhalb der Register des Hauptfensters (*Abbildung unten*) können Sie auswählen, ob die Concepts aus der aktuellen Taxonomie oder die Concepts aus der aktuellen Taxonomie und die Concepts von importierten Taxonomien angezeigt werden sollen. Wählen Sie den Eintrag *Alle Dateien anzeigen*, damit die importierten Taxonomien angezeigt werden. Wenn Sie große importierte Taxonomien aus der Anzeige herausfiltern, lässt sich die Bearbeitung erheblich beschleunigen.



Sortieren von Elementen

In der Auswahlliste zum *Sortieren* unterhalb der Register des Hauptfensters und auf der rechten Seite (*Abbildung unten*) können Sie die Elemente im Hauptfenster sortieren.



Das Sortierkriterium kann eines der folgenden sein:

- *Nach Anzeigenamen sortieren:* Die Elemente werden nach Ihrem Namen sortiert in alphabetischer Reihenfolge angezeigt. Die Namen enthalten auch das Präfix, d.h. **abc:yname** wird vor **bcd:xname** angezeigt.
- *Nach qualifiziertem Namen sortieren:* Die Elemente werden vollständig aufgelöst, d.h. ihre Namespaces werden erweitert und die erweiterten Namen werden in alphabetischer Reihenfolge aufgelistet. Wenn also das Namespace-Präfix **abc** an den Namespace **http:people.altova.com** gebunden ist, wird der Elementname **abc:yname** zu **http:people.altova.com:yname** aufgelöst.
- *Nach Substitution Group sortieren:* Es gibt vier Substitution Groups: *items*; *tuples*; *hypercubeItems*; *dimensionItems*.
- *Nach Typ sortieren:* Bezieht sich auf das "Type" Attribut des XBRL-Elements.

Suchen von Elementen im Hauptfenster

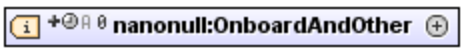
Um im Hauptfenster ein Element zu suchen, drücken Sie die Tastenkombination **Strg+F.** Daraufhin erscheint ein Suchdialogfeld, in das Sie die gewünschten String eingeben können. Auch die in der Taxonomie verwendeten Namespace-Präfixe können durchsucht werden.

Informationen zu Concepts (die <element> Elemente)

Jedes Taxonomie-Concept ist in einem XML-Schema-Element `<element>` (*siehe Liste unten*) definiert. Sie sehen diese einfache Textdefinition, wenn Sie in die Textansicht[144] des Taxonomiedokuments wechseln. (Um die Textdefinitionen eines importierten Taxonomiedokuments zu sehen, muss das importierte Taxonomiedokument in XMLSpy geöffnet werden.)

```
<xs:element id="icui_UnrealizedHoldingLoss"
       name="UnrealizedHoldingLoss"
       substitutionGroup="xbrli:item"
       type="xbrli:monetaryItemType"
       xbrli:balance="credit"
       xbrli:periodType="instant"
       abstract="false"
       nillable="true"/>
```

Jedes Element (oder Concept) wird auf dem Register "Elemente" mit einem Symbol angezeigt, das die Substitution Group (Item, Tuple, Hypercube oder Dimension) bezeichnet. Außerdem werden Symbole, die den Wert des `balance`-, `periodType`-, `abstract`- und `nillable`-Attributs des Concept kennzeichnen, links vom Namen des Concept angezeigt (*Abbildung unten*).



Um den Namen des Concept zu bearbeiten, doppelklicken Sie darauf und bearbeiten Sie ihn.

Substitution Group

Der Substitution Group-Wert eines Concept wird durch das Symbol des Concept angegeben:

 `xbrli:item`

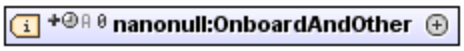
 `xbrli:tuple`

 `xbrldt:hypercubeItem`

© 2016-2022 Altova GmbH

 xbrldt:dimensionItem

In der Abbildung unten sehen Sie ein Element mit einem substitutionGroup Wert xbrli:item.



Die Attribute balance, periodType, abstract, nillable

Die zusätzlichen Symbole links vom Elementnamen (*siehe Abbildung oben*), bezeichnen den Wert der Hauptattribute des Concept. Diese sind von links nach rechts:

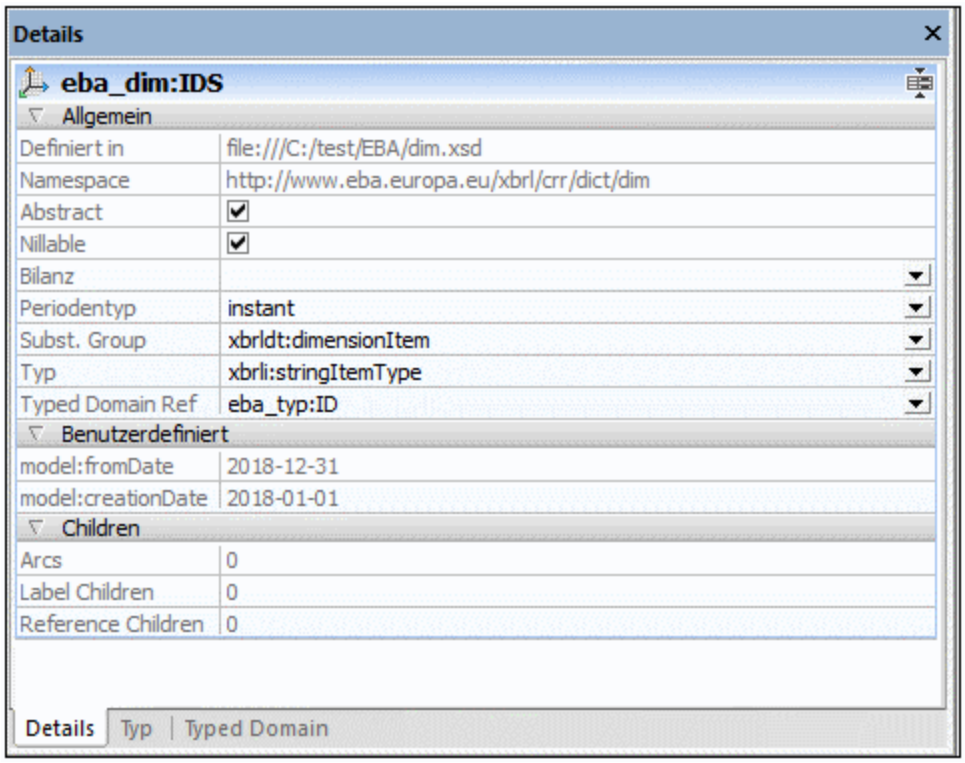
- xbrli:balance: ein Plus-Symbol für einen Wert credit, ein Minus-Symbol für debit
- xbrli:periodType: ein Uhrsymbol mit einem grauen Abschnitt zwischen den Zeigern steht für einen Wert duration, ein weißer Abschnitt steht für instant
- xs:abstract: schwarzes A Symbol, wenn true, grau, wenn false
- xs:nillable: schwarzes 0 Symbol, wenn true, grau, wenn false

Beachten Sie, dass die oben aufgelisteten xbrli: Attribute aus dem XBRL-Schema stammen und die xs: Attribute aus dem XML-Schema.

In der Abbildung oben, zeigt das Plus-Symbol an, dass der Wert des xbrli:balance Attributs credit ist. Die Werte der anderen Attribute in der Abbildung oben (xbrli:periodType, xs:abstract, xs:nillable), sind der Reihe nach: duration, false, (d.h. nicht abstract) und true (d.h. nillable).

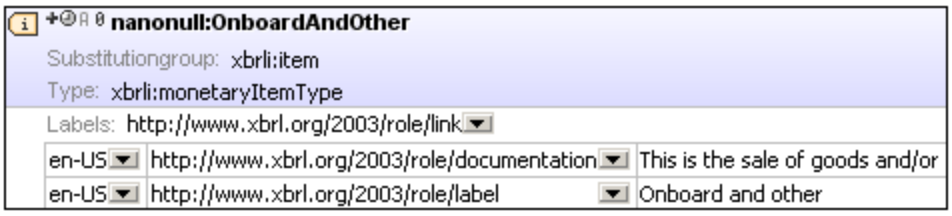
Die Werte dieser vier Attribute werden in einem Popup-Fenster angezeigt, wenn der Cursor über eines der vier Symbole platziert wird. Wenn Sie auf eines dieser Symbole klicken, erscheint eine Liste mit den für dieses Attribut zulässigen Werten. Sie können einen davon auswählen und auf diese Art den Wert eines Concept schnell bearbeiten.

Diese Attributwerte können auch in der Eingabehilfe "Details" (*Abbildung unten*) bearbeitet werden.



Die Attribute substitutionGroup, type

Wenn Sie auf das Pfeilsymbol links vom Elementnamen klicken, wird die Anzeige des Elements erweitert, sodass die Werte der beiden Attribute `xs:substitutionGroup` und `xs:type`, grafisch angezeigt werden (*Abbildung unten*).

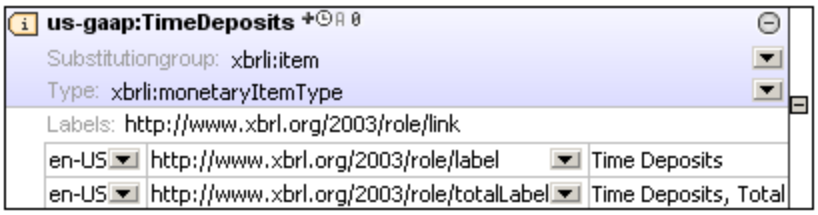


In der Abbildung oben lautet der Wert der Attribute `xs:substitutionGroup` und `xs:type` `xbrli:item` bzw. `xbrli:monetaryItemType`. Diese Werte können bearbeitet werden, indem Sie aus der Dropdown-Liste der entsprechenden Auswahlliste einen anderen Wert auswählen. Beide Attributwerte können auch in der Eingabehilfe "Details" (*siehe Abbildung weiter oben*) durch Doppelklick in das entsprechende Wertefeld in der Eingabehilfe "Details" und Bearbeitung des Werts durch Eingabe über die Tastatur bearbeitet werden.

Label Links

Um einen Sub-Eintrag für einen Label hinzuzufügen, rechtsklicken Sie auf ein Element und wählen Sie im Kontextmenü den Befehl **Label Linkrole hinzufügen.** Daraufhin wird eine Label-Zeile hinzugefügt (*Abbildung unten*), in der Sie die Eigenschaften des Labels (language, label type und value) bearbeiten können. Sie können Labels durch Klicken auf das + bzw. + Symbol auf der rechten Seite des Kastens erweitern bzw. reduzieren.

© 2016-2022 Altova GmbH



Die Eigenschaften "language" und "label type" der einzelnen Labels können durch Auswahl der gewünschten Eigenschaft aus der entsprechenden Auswahlliste bearbeitet werden (*Abbildung oben*).

Reference Links

Um einen Sub-Eintrag für eine Reference hinzuzufügen, rechtsklicken Sie auf ein Element und wählen Sie im Kontextmenü den Befehl **Reference Linkrole hinzufügen.** Daraufhin wird ein Reference-Kasten hinzugefügt, in dem Sie die Eigenschaften der Reference durch Klicken in das Feld "Reference" und Bearbeiten der Reference URI ändern können. Sie können References durch Klicken auf das + bzw. + Symbol auf der rechten Seite des Kastens erweitern bzw. reduzieren.

4.6.2 Hauptfenster: Die Register "Definitionen", "Presentation", "Calculation", "Formula", "Tabelle"

Das Hauptfenster enthält Register für jede der drei Beziehungen zwischen Concepts:

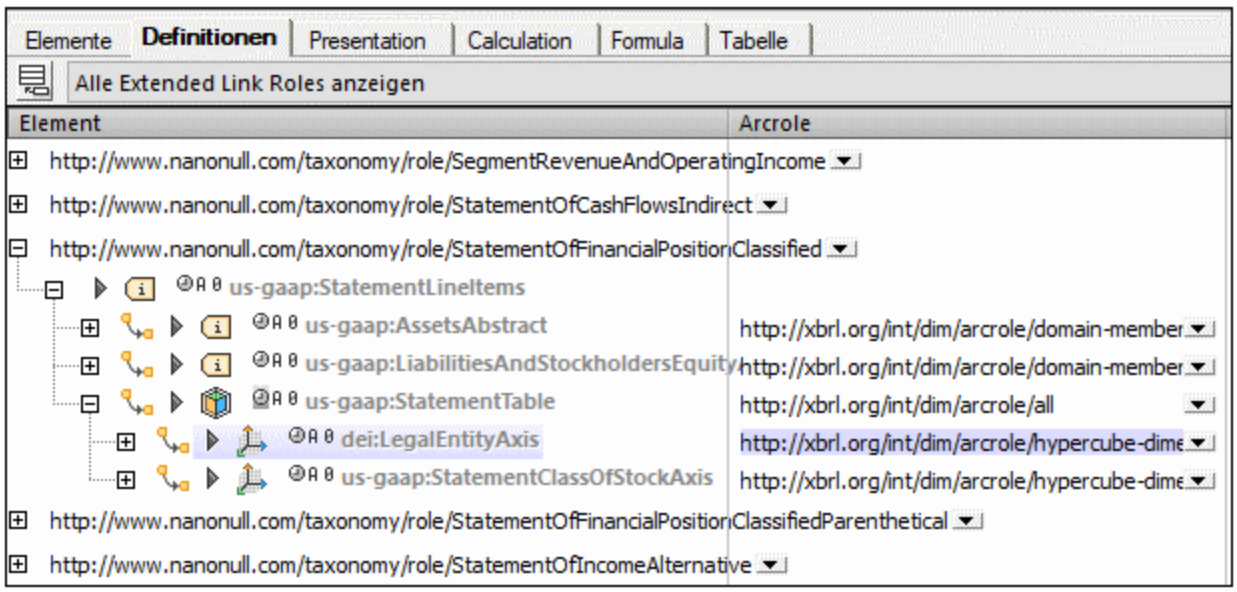
- Definitionsbeziehungen (Definition Relationships), die auf dem Register "Definitionen" angezeigt werden
- Darstellungsbeziehungen (Presentation Relationships), die auf dem Register "Presentation" angezeigt werden
- Berechnungsbeziehungen (Calculation Relationships), die auf dem Register "Calculation" angezeigt werden
- Formeldefinitionen und -beziehungen, die auf dem Register "Formula" angezeigt werden
- Tabellendefinitionen, die auf dem Register "Tabelle" angezeigt werden

Auf jedem Register (Definition, Presentation, Calculation, Formula und Tabelle) werden die Taxonomiebeziehungen dieser Art angezeigt (*Abbildung unten*). Sie können darauf die Beziehungen grafisch bearbeiten. Eine Beziehung zwischen zwei Concepts (ob eine Definitions-, Darstellungs- oder Berechnungsbeziehung) wird durch Erstellung eines Arc von einem Concept zum anderen Concept erstellt. Dieser *from–to* Arc wird in der grafischen Anzeige in Form eines gekrümmten Pfeils angezeigt. Die Beziehung zwischen zwei Concepts (in der Richtung *from–to*) wird Arcrole genannt. Die Arcrole eines Arc wird in der Eingabehilfe "Details" angezeigt, wenn das Element am *to*-Ende einer Beziehung ausgewählt ist, im Fall einer Definitionsbeziehung in einer Arcrole-Spalte auf dem Register "Definitionen" (*siehe Abbildung unten*).

Die Art, wie Relationship Arcs angezeigt werden, ist dieselbe für alle Beziehungsarten (Definition, Darstellung und Berechnung). In diesem Abschnitt wird jedes Register separat behandelt, wobei der Abschnitt über das Register "Definitionen" die grundlegende Beschreibung der Anzeige von Beziehungen enthält. Zusätzliche oder spezifische Informationen über Presentation und Calculation Arcs befinden sich in den jeweiligen Abschnitten. Nähere Informationen finden Sie in den Abschnitten, Erstellen von Beziehungen: Teil 1 866 und Erstellen von Beziehungen: Teil 2 869.

Register "Definitionen"

Auf dem **Register "Definitionen"** sehen Sie alle Definitionen der Taxonomie. Diese Definitionen werden in den in Definitionsbeziehungen (`.xml`) Datei/en enthaltenen Definition Arcs definiert. Auf dem Register "Definitionen" der XBRL-Ansicht wird die Struktur, die sich aus der Gruppe der Definitions Arcs ergibt, in einer erweiterbaren/reduzierbaren Baumstruktur angezeigt (*Abbildung unten*).



In dieser grafischen Anzeige der Definitionen wird jeder Definition Arc als gekrümmter Pfeil mit zwei Endpunkten (einem *from* Endpunkt und einem *to* Endpunkt) angezeigt. Die Art der Beziehung zwischen den beiden Elementen am jeweiligen Endpunkt wird in der *Arcrole* Spalte des Elements am *to* Endpunkt angezeigt. So wird z.B. in einer `hypercube-dimension` Beziehung die Beziehung (oder Arcrole) mit dem Element aufgelistet, das den `dimension` Teil des Elementpaars bildet. Arcrole URIs können auch in der Eingabehilfe "Details" eingegeben werden.

Nähere Informationen zu Definitionsbeziehungen finden Sie im Abschnitt Erstellen von Beziehungen: Teil 1 866.

Register "Presentation"

Presentation Arcs haben dieselben Attribute wie Definition Arc und funktionieren auf dieselbe Art (siehe Register "Definitionen" oben). Arcrole URIs werden in der Eingabehilfe "Details" eingegeben. Ein wichtiges Darstellungsattribut ist das `order` Attribut, welches die Reihenfolge festlegt, in der Child-Elemente eines einzigen übergeordneten Elements dargestellt werden. Auf dem Register "Presentation" werden solche Child-Elemente in der richtigen aufsteigenden Reihenfolge angezeigt. Der Wert des order-Attributs kann schnell geändert werden, indem Sie ein Child-Element in der geordneten Liste an eine andere Stelle ziehen. Außerdem kann der Wert des `order` Attributs auch in der Eingabehilfe "Details" (*siehe Abbildung unten*) geändert werden.

Nähere Informationen zu Darstellungsbeziehungen finden Sie im Abschnitt Erstellen von Beziehungen: Teil 2 869.

 © 2016-2022 Altova GmbH

Register "Calculation"

Calculation Arcs haben dieselben Attribute wie Definition Arcs und funktionieren ähnlich wie Definition Arcs (siehe Register "Definitionen" oben). Arcrole URIs werden in der Eingabehilfe "Details" eingegeben. Es gibt zwei Arten von Arcroles:

- diejenigen, die die Werte der Elemente zu einem anderen Element addieren; und
- diejenigen, die keine Summierungsbeziehung sondern eine gleichrangige Beziehung darstellen

Im ersteren Fall bestimmt das `weight` Attribut, wie viel vom Wert des Elements zum Aggregierungselement addiert werden soll. Der Wert `1.0` gibt an, dass 100% des Werts addiert werden sollen. Ein negativer Wert gibt an, dass der Wert vom Aggregierungselement subtrahiert werden soll. Der Wert des `weight` Attributs kann auf dem Register "Calculation" sowie in der Eingabehilfe "Details" bearbeitet werden.

Register "Formula"

Auf dem Register "Formula" können Formulas definiert und verwaltet werden. Zusammen mit den Eingabehilfen "Übersicht" und "Details" können Sie auf dem Register "Formula" XBRL-Formulas erstellen und bearbeiten und Beziehungen zwischen Formula-Komponenten im Diagramm definieren. Nähere Informationen dazu finden Sie im Abschnitt XBRL Formula Editor 878

Register "Tabelle"

Tabellen bieten eine alternative Möglichkeit, Ansichten von in XBRL-Taxonomien definierten Concepts zu definieren. Diese Definitionen sind in Tabellen-Linkbases, die Sie auf dem Register "Tabelle" erstellen und bearbeiten können, enthalten. Eine Anleitung, wie Sie Tabellen-Linkbases bearbeiten, finden Sie im Abschnitt XBRL-Tabellendefinitionseditor 904.

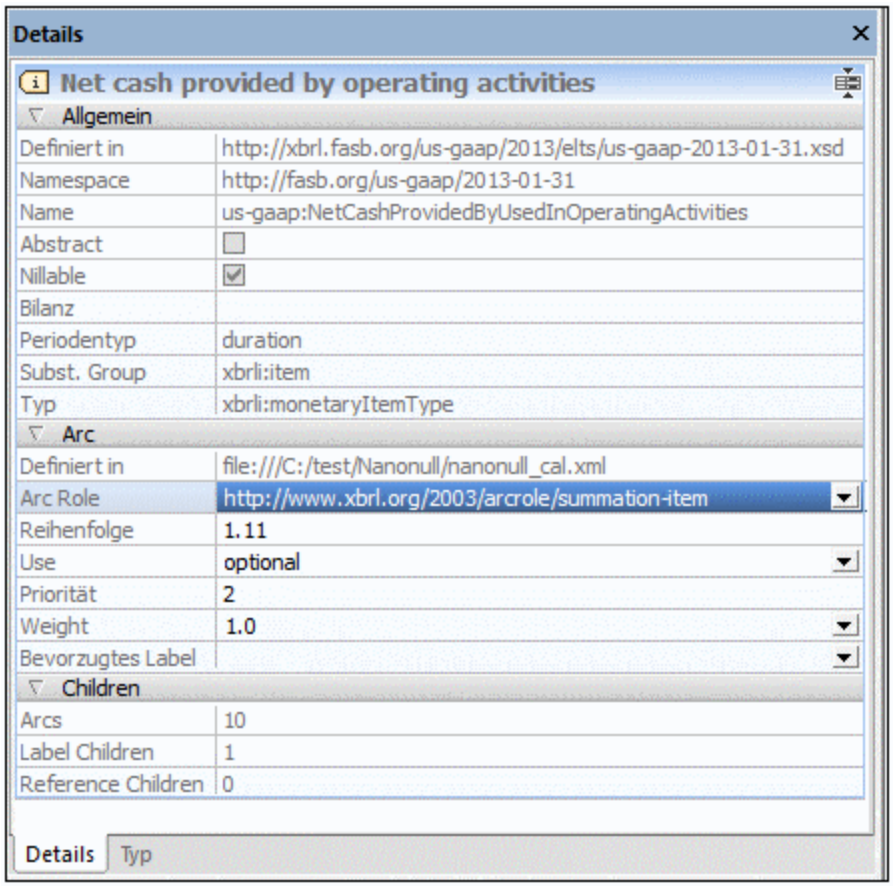
Bearbeiten in der grafischen Ansicht

Es stehen die folgenden Bearbeitungsmöglichkeiten zur Verfügung:

- Sie können Elemente aus der Eingabehilfe "Globale Elemente" in eine *to* Beziehung eines Elements in der Baumstruktur ziehen.
- Sie können eine Arcrole für ein Element erstellen oder bearbeiten, indem Sie die gewünschte Arcrole (in der Spalte *Arcrole*) auswählen. Die in der Arcrole definierte Beziehung drückt eine *from–to* Beziehung aus, wobei das ausgewählte Element am *to*-Endpunkt der Beziehung verwendet wird.
- Sie können Elemente an andere Stellen in der Baumstruktur ziehen. Auf diese Art können Sie den Wert des `order` Attributs schnell ändern.
- Sie können die Eigenschaften eines Elements bearbeiten, indem Sie auf eines seiner Eigenschaftssymbole klicken und sie bearbeiten oder indem Sie den Eigenschaftskasten erweitern und anschließend die Eigenschaften im Eigenschaftskasten bearbeiten.

Bearbeiten in der Eingabehilfe "Details"

Wenn ein Element im Hauptfenster ausgewählt ist, werden die Eigenschaften seines Definition Arc in der Eingabehilfe "Details" im Abschnitt "Arc" (*siehe Abbildung unten*) angezeigt. Die Werte dieser Eigenschaften können durch Doppelklick in ein Wertefeld und Eingabe des gewünschten Werts oder, falls verfügbar, durch Auswahl eines Werts in der Dropdown-Liste seiner Auswahlliste bearbeitet werden.



Zusätzlich dazu werden die Eigenschaften des Arc des ausgewählten Elements im Abschnitt *Arc* der Eingabehilfe "Details" angezeigt. Bei dem Arc handelt es sich je nachdem, welches Register gerade aktiv ist, um einen Definitions Arc, einen Presentation Arc oder einen Calculation Arc. Die Arcrole kann in das Feld *Arcrole* eingegeben werden.

Label- und Reference-Beziehungen werden unter der Überschrift *Children* angezeigt. Diese Beziehungen können auf dem Register "Elemente" bearbeitet werden.

Zusätzliche Informationen finden Sie im Abschnitt XBRL[829].

4.6.3 Eingabehilfen in der XBRL-Ansicht

Die XBRL-Ansicht enthält die folgenden Eingabehilfen:

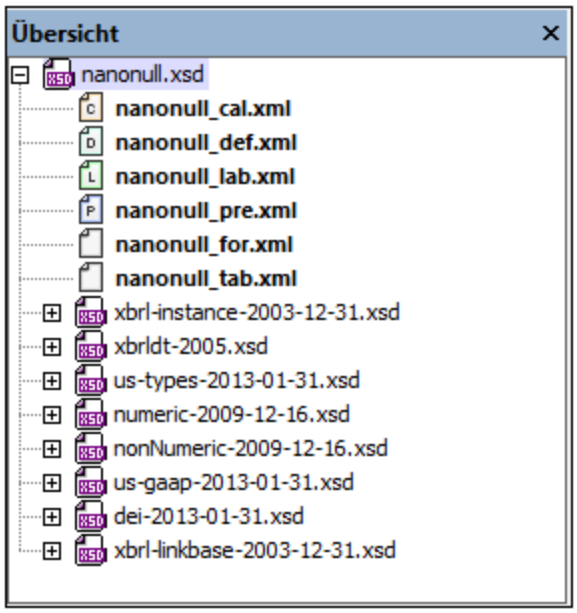
- Eine Eingabehilfe Übersicht[275]
- Eine Eingabehilfe Globale Elemente[276]
- Eine Eingabehilfe Details[278]

Die Validierungsinformationen werden im Fenster "Meldungen"[122] angezeigt. Zusätzliche Informationen finden Sie im Abschnitt XBRL[829].

© 2016-2022 Altova GmbH

Eingabehilfe "Übersicht"

In der Eingabehilfe "Übersicht" werden die Taxonomiedateien[848] in Form einer Baumstruktur angezeigt (*Abbildung unten*).



Diese Struktur ist folgendermaßen gegliedert:

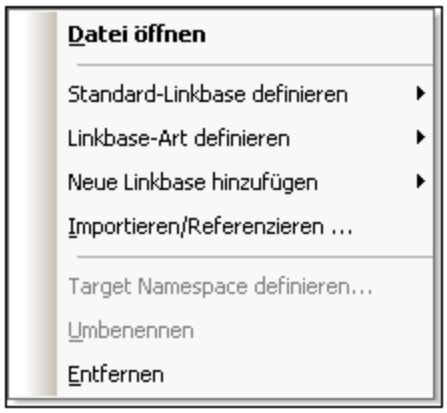
- Die Concepts-Hauptdatei (ein XML-Schema-Dokument) wird an der Root der Struktur angezeigt und ist die aktuell aktive Datei.
- Die Relationship Linkbase-Dateien (XML-Dateien) sind mit einem farbigen Dateisymbol mit einem Buchstaben versehen, der dem Anfangsbuchstaben der Beziehungsart entspricht.

  steht für eine Definitions Linkbase;
  steht für eine Labels Linkbase;
  steht für eine Calculations Linkbase;
  steht für eine Presentation Linkbase;
  steht für eine Reference Linkbase;
  steht für eine Formula Linkbase.

- Importierte Schemas werden auf niedrigeren Ebenen der Hierarchie aufgelistet. Alle XML-Schema-Dateien (`.xsd`) sind durch ein XSD-Symbol gekennzeichnet.

Diese Informationen, die in der Eingabehilfe "Übersicht" angezeigt werden, stammen aus dem Element `/schema/annotation/appinfo` der Concepts-Hauptdatei. Nähere Informationen zu diesem Element finden Sie unter Taxonomiedateien[848].

Wenn Sie mit der rechten Maustaste auf eine Datei in der Eingabehilfe "Übersicht" klicken, erscheint ein Kontextmenü (*Abbildung unten*), in dem die folgenden Befehle zur Verfügung stehen:

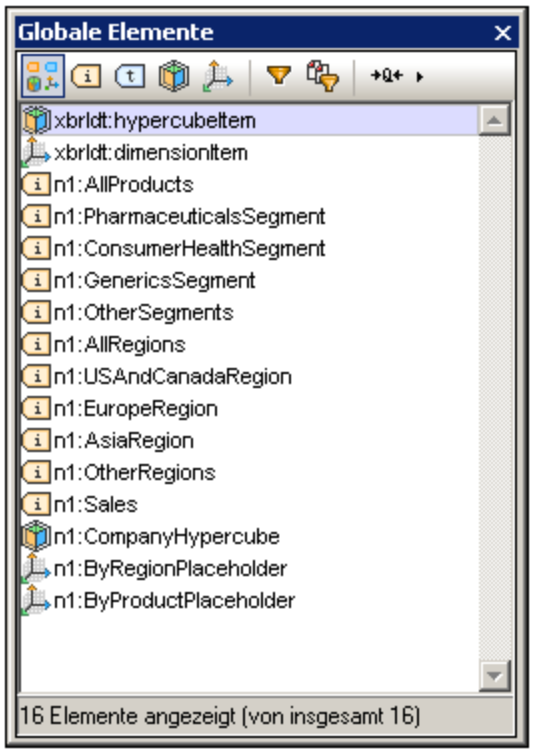


- *Datei öffnen:* Öffnet das ausgewählte XML-Schema oder die ausgewählte XML-Datei in XMLSpy.
- *Standard-Linkbase definieren:* Wenn mehrere Dateien einer einzigen Beziehungsart (z.B. Presentation Relationships) vorhanden sind, so kann eine davon als Standard-Linkbase definiert werden. Neu erstellte Beziehungen dieser Beziehungsart werden in dieser Standard-Linkbase gespeichert. Die Standard-Linkbase jeder Beziehungsart wird fett angezeigt.
- *Linkbase-Art definieren:* Definiert die Beziehungsart der ausgewählten Linkbase. Wählen Sie im Untermenü, das ausgerollt wird, die gewünschte Beziehungsart aus. Mit der Beziehungsart **Alle** wird festgelegt, dass die ausgewählte Linkbase für alle Beziehungsarten verwendet wird.
- *Neue Linkbase hinzufügen:* Erstellt eine neue Linkbase-Datei für Beziehungen einer der fünf Arten (Definition, Presentation, Calculation, Label, Resource). Die hinzugefügte Datei kann durch Rechtsklick und Auswahl des Kontextmenübefehls **Umbenennen** umbenannt werden. Daraufhin wird ein neues `linkbaseRef` Element zur Concepts-Datei hinzugefügt; dieses Element referenziert die neu hinzugefügte Linkbase.
- *Importieren/Referenzieren:* Importiert eine Standard-Taxonomie oder erstellt eine Referenz zu einem bestehenden XML-Schema oder einer Linkbase. Wenn Sie die Standard-Taxonomie-Option auswählen, wird ein Fenster mit einer Liste der Standard-Taxonomien angezeigt. Wählen Sie die Taxonomie aus, die importiert werden soll.; Nähere Informationen siehe Importieren einer Taxonomie 852. Wenn Sie eine Referenz zu einem XML-Schema oder einer Linkbase erstellen, wird ein neues `linkbaseRef` Element, das die Referenz enthält, zur Concepts-Datei hinzugefügt. Wenn Sie auf die Option "XML-Schema" oder "Linkbase" klicken, erscheint ein Dialogfeld, in dem Sie die gewünschte Datei suchen können.
- *Target Namespace definieren:* Definiert den Target Namespace und deklariert diesen Namespace als "in scope" im Geltungsbereich im `xs:schema` Element (d.h. für die gesamte Taxonomie). Siehe Befehlsbeschreibung **XBRL | Target Namespace definieren** 1538.
- *Umbenennen:* Dient zum Umbenennen der ausgewählten Datei.
- *Entfernen:* Entfernt die ausgewählte Datei aus der Übersicht und entfernt ihr `linkbaseRef` Element aus der Concepts-Datei.

Eingabehilfe "Globale Elemente"

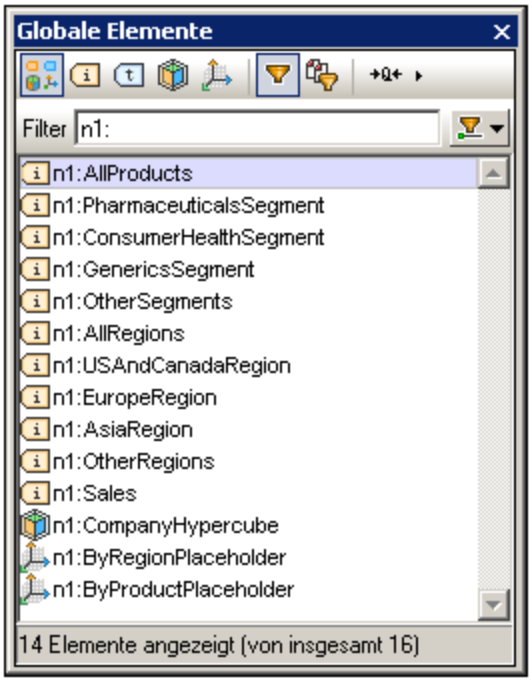
In der Eingabehilfe "Globale Elemente" (*Abbildung unten*) werden alle Items, Tuples, Hypercubes und Dimensions eines Taxonomiedokuments angezeigt. Elemente können aus der Eingabehilfe "Globale Elemente" in das Hauptfenster gezogen werden.

© 2016-2022 Altova GmbH



In der Eingabehilfe "Globale Elemente" stehen die folgenden Funktionen zur Verfügung:

- *Filtern nach Typ:* Jeder Elementtyp (Item, Tuple, Hypercube und Dimension) kann durch Klicken auf die entsprechende Schaltfläche in der Symbolleiste der Eingabehilfe "Globale Elemente" ein- bzw. ausgeblendet werden. Wenn Sie auf die Schaltfläche *Alle Elemente anzeigen* klicken, werden alle Elemente angezeigt. Der ausgewählte Filter erscheint markiert. (In der Abbildung oben *Alle anzeigen*).
- *Filtern nach Text:* Wenn Sie auf die Schaltfläche *Textfilter* klicken, wird die Textfilterleiste angezeigt (*siehe Abbildung unten*).



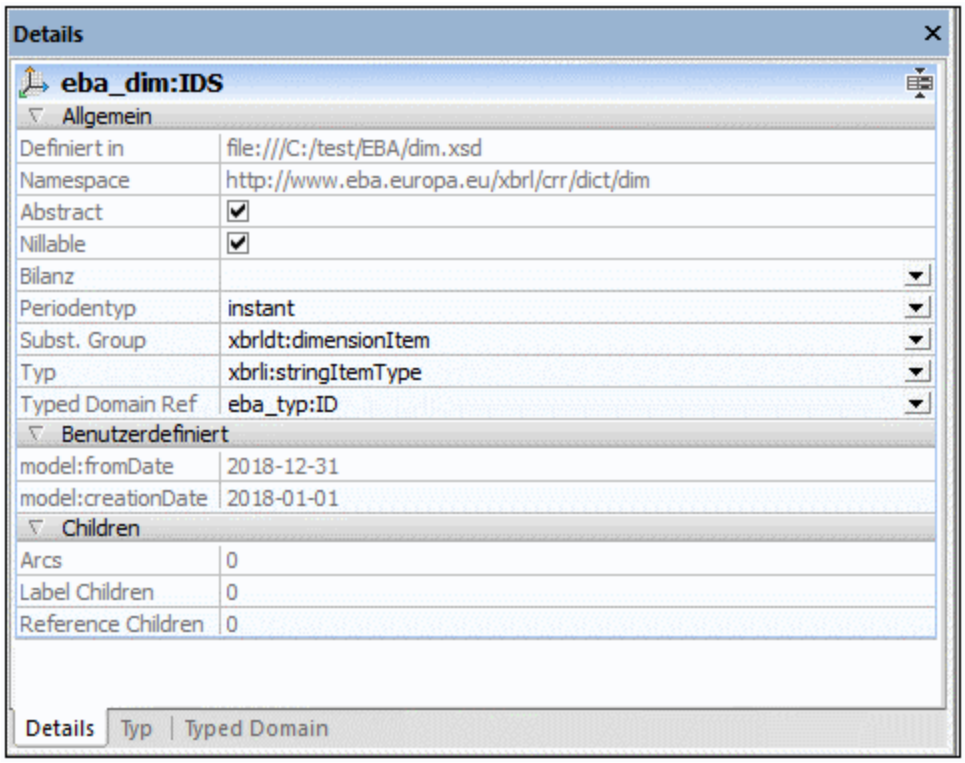
Wählen Sie nach Auswahl des Textfilters in der Auswahlliste auf der rechten Seite der Textfilterleiste (*in der Abbildung oben unter dem Cursorpfeil*) eine Bedingung, nach der die Filterung durchgeführt werden soll. Zur Verfügung stehen die Optionen *Enthält*, *Enthält nicht* und *Beginnt mit*. In der Abbildung oben wurde die Bedingung *Beginnt mit* ausgewählt. Geben Sie als nächstes den Text für den Filter in das Textfilterfeld ein. Die angezeigten Elemente werden dementsprechend gefiltert. In der Abbildung oben wurde die Liste nach Namen, die mit `n1` beginnen, gefiltert. Beachten Sie, dass: (i) die Filter auf die Namen der Einträge als Textstrings angewendet werden und (ii) die Namen der Einträge über die Schaltfläche "Format" (siehe unten) geändert werden können und die Änderungen im Namensformat berücksichtigt werden.

- *Filtern nach Quellen*: Die gewünschten Quellen können in einem Popup-Fenster ausgewählt werden und nur die Elemente in den ausgewählten Quellen werden angezeigt.
- *Format:* Für die Art, wie Elementnamen angezeigt werden sollen, gibt es drei Formatoptionen: *Kurzer qualifizierter Name, Erweiterter qualifizierter Name* und *Label*. Beim kurzen qualifizierten Namen werden die Präfixe verwendet, die den entsprechenden Namespaces zugewiesen sind; bei erweiterten qualifizierten Namen (die Schaltfläche ist ) wird der gesamte Namespace verwendet; und bei Label werden die mit den Elementen verknüpften Labels verwendet. Beachten Sie, dass die Auswahl des Formats sich insofern auf die Option *Textfilter* auswirkt, als der Filter auf Elemente angewendet wird, wie sie entsprechend der aktuell ausgewählten *Format*-Option aufgelistet sind.
- *Drag-and-Drop-Funktion:* Elemente in der Eingabehilfe "Globale Elemente" können mit der Maus in Beziehungen in jeder der Beziehungsansichten (Definitionen, Darstellungen, Berechnungen) des Hauptfensters gezogen werden.

Eingabehilfe "Details"

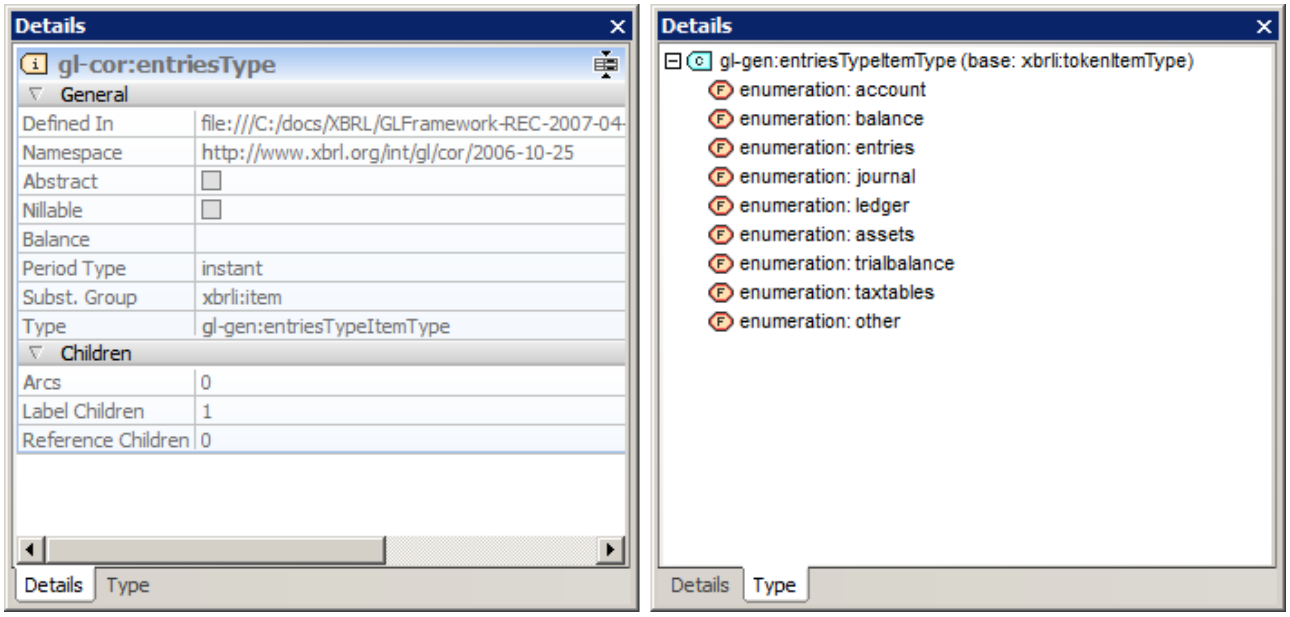
Wenn ein Element im Hauptfenster ausgewählt ist, werden in der Eingabehilfe "Details" (*Abbildung unten*) seine Eigenschaften angezeigt. Wenn das Element benutzerdefinierte Attribute hat, wird in der Eingabehilfe ein Abschnitt namens "*Benutzerdefiniert*" mit den Attributen angezeigt.

© 2016-2022 Altova GmbH



Die Eigenschaften einiger Elemente können in der Eingabehilfe "Details" bearbeitet werden: z.B. *Abstract*, *Nillable*, *Balance*, *Period Type*, *Substitution Group* und *Type*.

Die Eingabehilfe "Details" hat außerdem ein Register "Typ", auf dem Sie den Typ des ausgewählten Item als Root-Node einer Baumstruktur sehen (*siehe Abbildung unten*). Für Concepts vom Typ `enum:enumerationItemType` können erweiterbare Erweiterungen mit mehrsprachigen Labels gemäß der Extensible Enumerations Recommendation vom 29.Oktober 2014 definiert werden. Für Items dieses Typs stehen zusätzliche Einträge für die Typ-Attribute `enum:domain`, `enum:linkrole` und `enum headUsable` zur Verfügung.



Wenn der Typ weder ein xbrli-item-type noch ein vordefinierter XSD-Typ ist, werden in der Eingabehilfe in Klammer Informationen über seinen Basistyp angezeigt. Die Baumstruktur bietet ein Kontextmenü, über das Sie die Typdefinitionen des Concept in der Schema-Ansicht anzeigen oder bearbeiten können.

4.6.4 Einstellungen für die XBRL-Ansicht

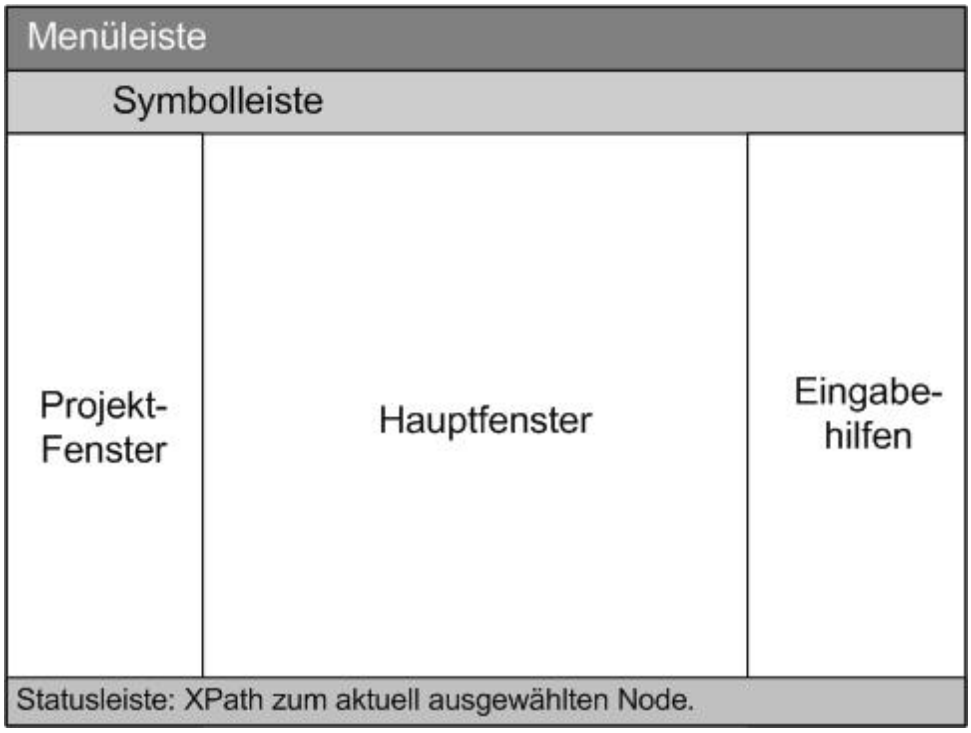
Für die XBRL-Ansicht können Sie zwei Arten von Einstellungen konfigurieren:

- die Schriftarten und Farben von Komponenten in der XBRL-Ansicht. Diese Einstellungen werden über den Abschnitt "Schriftarten und Farben" des Dialogfelds "Optionen" (**Extras | Optionen**[1632]) aufgerufen.
- das Layout und Format der Ansicht sowie Label-Standardeinstellungen. Diese Einstellungen stehen im Dialogfeld "Einstellung für XBRL-Ansicht"[1545], das mit dem Befehl **XBRL | Ansichtseinstellungen** aufgerufen wird, zur Verfügung.

© 2016-2022 Altova GmbH

4.7 Authentic-Ansicht

Die Authentic-Ansicht besteht aus einer Menüleiste am oberen Rand des Fensters und drei Bereichen: dem Projekt-Fenster, dem Hauptfenster und den Eingabehilfen-Fenstern (siehe Abbildung unten).



Eine ausführliche Beschreibung der Benutzeroberfläche der Authentic-Ansicht[622] finden Sie im Abschnitt **Authentic**[606] | **Benutzeroberfläche der Authentic-Ansicht**[622].

4.8 Browser-Ansicht

Die Browser-Ansicht wird normalerweise zur Anzeige folgender Dateien verwendet:

- XML-Dateien, die mit einer XSLT-Datei verknüpft wurden. Wenn Sie in die Browser-Ansicht wechseln, wird die XML-Datei mit Hilfe eines XSLT Stylesheet on-the-fly transformiert. Das Ergebnis wird direkt in der Browser-Ansicht angezeigt.
- HTML-Dateien, die direkt im HTML-Format erstellt wurden oder mittels einer XSLT-Transformation einer XML-Datei erzeugt wurden.

Um XML- und HTML-Dateien in der Browser-Ansicht anzuzeigen, klicken Sie auf das Register **Browser**.

Browser-Programme in der Browser-Ansicht

Standardmäßig wird in der Browser-Ansicht derzeit Microsoft Internet Explorer als Browser verwendet. Wenn Sie lieber den neueren Browser von Microsoft, Edge WebView2, für die Browser-Ansicht verwenden möchten, aktivieren Sie die entsprechende Option im Abschnitt "Ansicht" 1617 des Dialogfelds "Optionen" 1602.

Anmerkung: Da Microsoft Edge WebView2 das Chromium-Software-Projekt, auf dem der Browser Chrome von Google basiert, verwendet, bietet die Browser-Ansicht bei Verwendung von WebView2 auch eine gute Vorschau auf die Anzeige einer Webseite in Chrome.

Anmerkungen zu Microsoft Internet Explorer

Für die Browser-Ansicht wird Microsoft Internet Explorer 5.0 oder höher oder Microsoft Edge WebView2 benötigt (*siehe oben*).

Beachten Sie zur Verwendung von Internet Explorer in der Browser-Ansicht die folgenden Punkte:

- Zur Anzeige von XML-Dateien, die mit Hilfe eines XSLT Stylesheet transformiert wurden, sollten Sie unbedingt Internet Explorer 6.0 oder höher verwenden, da diese Versionen MSXML 3.0 verwenden, einen XML-Parser, die den XSLT 1.0 Standard vollständig unterstützt. Eventuell sollten Sie auch MSXML 4.0 installieren.
- Die Unterstützung von XSL in IE 5 ist nicht zu 100 % kompatibel mit der offiziellen XSLT Recommendation. Wenn Sie daher in der Browser-Ansicht Probleme mit IE 5 haben, sollten Sie auf IE 6 oder höher upgraden.
- Außerdem sollten Sie im Allgemeinen überprüfen, welche XSLT-Version Ihre Version von Internet Explorer unterstützt.
- Wenn bei der korrekten Anzeige von HTML in Internet Explorer Probleme auftreten, inkludieren Sie den folgenden **meta**-Tag in das **head**-Element Ihres HTML-Dokuments:

```
<head>
... <meta http-equiv="X-UA-Compatible" content="ie=edge">...
</head>
```

Developer Tools in der Browser-Ansicht

Sie können Ihren HTML-Code mit Hilfe der Developer Tools des Edge Browsers analysieren, debuggen und testen. Um die Tools aufzurufen, klicken Sie mit der rechten Maustaste in das Browser-Ansichtsfenster und wählen Sie den Befehl **Developer Tools öffnen**.

© 2016-2022 Altova GmbH

Funktionen der Browser-Ansicht

In der Browser-Ansicht stehen die folgenden Funktionen zur Verfügung. Sie können über das Menü **Browser**, **Datei** und das Menü **Bearbeiten** aufgerufen werden.

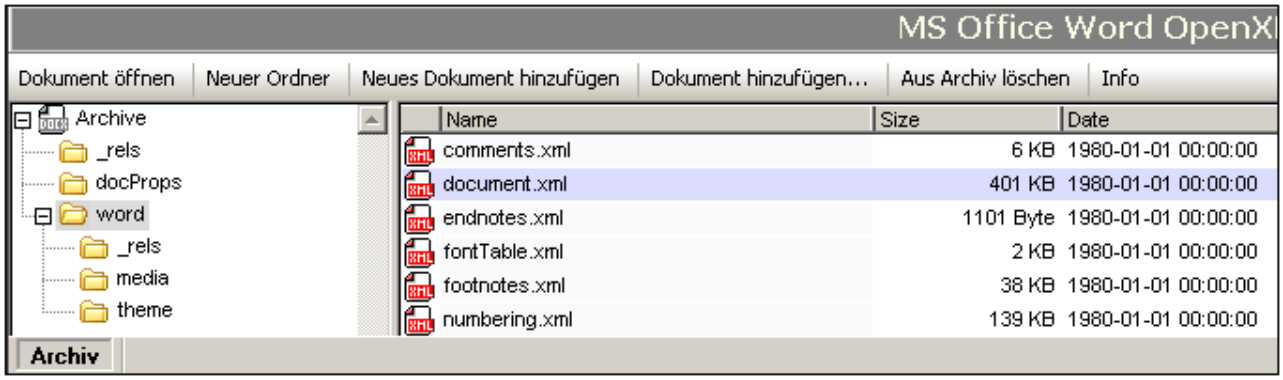
- *Öffnen in einem separaten Fenster:* Wenn die Browser-Ansicht ein separates Fenster ist, können Sie die Browser-Ansicht Seite an Seite mit der Bearbeitungsansicht desselben Dokuments anzeigen. Klicken Sie dazu auf den Menübefehl **Browser | Separates Fenster**. Dies ist ein Ein/Aus-Befehl, mit dem Sie zwischen der Browser-Ansicht in (i) einem separaten Fenster und (ii) derselben Ansicht auf einem Register im Hauptfenster wechseln können. Diese Befehle stehen auch im Dropdown-Menü der **Browser-Ansicht** (am unteren Rand des Hauptfensters) zur Verfügung. Im Dialogfeld "Optionen" können Sie auf dem Register "Ansicht" 1617 festlegen, ob die Browser-Ansicht standardmäßig in einem separaten Fenster angezeigt werden soll.
- *Vorwärts und Zurück:* die allgemeinen Browser-Befehle zum Navigieren durch Seiten, die in der Browser-Ansicht geladen wurden. Diese Befehle befinden sich im Menü **Browser**.
- *Schriftgröße:* Kann über das Menü **Browser** angepasst werden.
- *Abbrechen, Aktualisieren, Drucken:* Weitere Browser-Standardbefehle. Diese Befehle befinden sich in den Menüs **Browser** und **Datei**.
- *Suchen:* Dient zum Suchen von Textstrings. Dieser Befehl befindet sich im Menü **Bearbeiten**.
- *Info-Fenster:* In diesem Fenster stehen Optionen zur Verfügung, um die aktive HTML-Seite mit einem beliebigen auf dem Rechner installierten Browser anzuzeigen und die installierten Browser zu öffnen oder zu entfernen.

4.9 Archivansicht

Eine Office Open XML (OOXML)-Datei[955], eine ZIP-Datei[961] (z.B. WinZip oder WinRAR) oder eine EPUB-Datei[963] kann in der Archivansicht-Ansicht geöffnet und bearbeitet werden. OOXML-, ZIP- und EPUB-Dateien können in der ARCHIVE-VIEW%>-Ansicht nicht nur in ihrer Struktur geändert werden, einzelne Dateien im Archiv können über die ARCHIVE-VIEW%>-Ansicht auch geöffnet, in einer der Bearbeitungsansichten von XMLSpy bearbeitet und anschließend wieder im Archiv gespeichert werden.

Archivdateien und die Archivansicht

Wenn eine Archivdatei (OOXML-, ZIP- oder EPUB-Datei) in XMLSpy erstellt oder geöffnet[957] wird, wird sie in der Archivansicht-Ansicht (*Abbildung unten)* geöffnet. Sie können gleichzeitig mehrere Archive öffnen, wobei jede Archivdatei in einem separaten Archivansicht-Ansichtsfenster angezeigt wird. Die Art der Archivdatei wird in der rechten oberen Ecke der Archivansicht-Ansicht angezeigt. In der Abbildung unten handelt es sich bei der Archivdatei um das Format MS Office Word Open XML.



Ordneransicht

Die Ordneransicht befindet sich auf der linken Seite des Archivansicht-Ansichtsfensters und zeigt die Ordnerstruktur des ZIP-Archivs an. Die Ordner auf den einzelnen Ebenen werden in alphabetischer Reihenfolge angezeigt. Um die Unterverzeichnisse eines Ordners zu sehen, klicken Sie auf das Plus-Symbol links vom Ordner. Wenn ein Ordner kein Plus-Symbol auf seiner linken Seite hat, so enthält er keine Unterverzeichnisse. Um die Dokumentdateien (in der Folge Dokumente genannt) in einem Ordner anzuzeigen, wählen Sie den Ordner aus. Die Dateien werden anschließend im Hauptfenster angezeigt. In der Abbildung oben befinden sich die im Hauptfenster angezeigten Dokumente im Ordner `word`. Dieser Ordner enthält zwei Unterverzeichnisse: `_rels` und `theme`.

Hauptfenster

Im Hauptfenster werden die Dokumente in dem Ordner aufgelistet, der in der Ordneransicht ausgewählt ist. Die Dokumente werden in alphabetischer Reihenfolge jeweils mit ihrer nicht komprimierten Größe und dem Datum und der Uhrzeit der letzten Änderung angezeigt. Um ein Dokument in der Archivansicht-Ansicht zu öffnen, doppelklicken Sie darauf. Das Dokument wird in einem separaten XMLSpy Fenster geöffnet.

Befehlsschaltflächen

Die Befehlsschaltflächen befinden sich entlang des oberen Rands des Archivansicht-Ansichtsfensters.

© 2016-2022 Altova GmbH

- ***Dokument öffnen:*** Ist aktiv, wenn ein Dokument im Hauptfenster ausgewählt ist. Wenn Sie darauf klicken, wird das ausgewählte Dokument geöffnet. Ein Dokument kann auch durch Doppelklick auf das im Hauptfenster aufgelistete Dokument geöffnet werden.
- ***Neuer Ordner:*** Fügt einen neuen Ordner zu dem Ordner hinzu, der in der Ordneransicht aktuell ausgewählt ist. Der Ordner muss sofort nach seiner Erstellung in der Ordneransicht benannt werden. Ein Ordner kann später nicht mehr umbenannt werden. Der neue Ordner wird beim Speichern der Archivdatei im Archiv gespeichert.
- ***Neues Dokument hinzufügen:*** Fügt zu dem gerade in der Ordneransicht ausgewählten Ordner ein neues Dokument hinzu. Das neue Dokument wird über das XMLSpy-Dialogfeld "Neues Dokument erstellen" erstellt und in einem separaten XMLSpy Fenster geöffnet. Das Dokument muss unmittelbar, nachdem es in der Dokumentliste des ausgewählten Ordners aufgelistet wird, benannt werden. Das Dokument wird erst dann im Archiv gespeichert, wenn es in seinem Bearbeitungsfenster gespeichert wird oder wenn die Archivdatei gespeichert wird.
- ***Dokument hinzufügen:*** Öffnet ein Dialogfeld zum Durchsuchen, in dem Sie das gewünschte Dokument suchen können. Das Dokument wird zur Liste im Hauptfenster hinzugefügt, in dem die Dokumente im derzeit ausgewählten Ordner aufgelistet werden, und in einem separaten XMLSpy Fenster geöffnet. Damit das Dokument im Archiv gespeichert wird, muss es entweder im eigenen Fenster oder in der Archivdatei gespeichert werden.
- ***Aus Archiv löschen:*** Löscht das (im Hauptfenster) ausgewählte Dokument oder den ausgewählten Ordner (Ordneransicht) aus dem Archiv. Die Archivdatei muss gespeichert werden, damit die Dokumente tatsächlich gelöscht werden.
- ***Info:*** Blendet das Info-Fenster ein bzw. aus. *Siehe unten*.

Info-Fenster

Das Info-Fenster wird durch Klicken auf die Befehlsschaltfläche "Info" ein- und ausgeblendet. Das Info-Fenster enthält allgemeine Informationen zur Archivdatei, wie z.B. die Anzahl der darin enthaltenen Dateien, deren komprimierte und unkomprimierte Größe und das Komprimierungsverhältnis.

4.10 Gebräuchliche Tastaturkürzel

Die folgende Liste enthält die Standard-Tastaturkürzel häufig verwendeter Bearbeitungsbefehle. Sie können die Standard-Tastaturkürzel im Dialogfeld "Anpassen" auf dem Register "Tastatur" 1589 ändern.

Funktionstasten-Tastaturkürzel (einschl. Validierung und Transformation)

F1	Menü "Hilfe"
F1 + Alt	Letzte Datei öffnen
F3	Weitersuchen
F4 + STRG	Aktives Fenster schließen
F4 + Alt	XMLSpy schließen
F5	Aktualisieren
F6 + STRG	Offene Fenster der Reihe nach aufrufen
F7	Wohlgeformtheit prüfen
F8	Validieren
F10	XSL-Transformation
F10 + STRG	XSL:FO-Transformation

Datei- und Applikationsbefehle

Alt + F1	Letzte Datei öffnen
STRG + O	Datei öffnen
STRG + N	Datei neu
STRG + P	Datei drucken
STRG + S	Datei speichern
STRG + F4	Aktives Fenster schließen
STRG + F6	Offene Fenster der Reihe nach aufrufen
STRG + TAB	Zwischen offenen Dokumenten wechseln
Alt + F4	XMLSpy schließen

Diverse Tastaturkürzel

Pfeil nach oben/nach unten	Cursor oder Auswahlleiste verschieben
Esc	Bearbeitungen verwerfen oder Dialogfeld schließen
Eingabetaste	Auswahl bestätigen
Entf	Zeichen oder Auswahl löschen
Umschalt + Entf	Ausschneiden

Bearbeitungsbefehle

STRG + A	Alles markieren

© 2016-2022 Altova GmbH

STRG + F	Suchen
STRG + G	Gehe zu Zeile/Zeichen
STRG + H	Ersetzen
STRG + V	Einfügen
STRG + X	Ausschneiden
STRG + Y	Wiederherstellen
STRG + Z	Rückgängig machen

5 XML

In diesem Abschnitt wird beschrieben, wie Sie in XMLSpy mit XML-Dokumenten arbeiten. Es werden die folgenden Aspekte behandelt:

- Erstellen, Öffnen und Speichern von XML-Dokumenten[289]. In diesem Abschnitt werden einige wichtige XMLSpy-Einstellungen zur Erstellung von Dateien erläutert.
- XML-Dokumente können in der Textansicht[293], der Grid-Ansicht[296] und der Authentic-Ansicht[335] bearbeitet werden. Sie können die Ansicht auswählen, die sich für Ihre Zwecke am besten eignet und während der Bearbeitung zwischen den Ansichten wechseln. Jede der Ansichten bietet andere Vorteile.
- Sie können XML-Fragmente[343] aus externen Quellen schnell und einfach zu Ihrem XML-Dokument hinzufügen.
- Anleitung zur Verwendung der verschiedenen XML-Validierungsfunktionen[339] von XMLSpy.
- Eingabehilfen[337] für XML-Dokument haben bestimmte Funktionen, die hier beschrieben werden.
- Wie XML-Dokumente mit XSLT und XQuery verarbeitet werden[345]. Es werden verschiedene XMLSpy Funktionen im Zusammenhang damit erläutert. In einem Abschnitt zu PDF-Schriftarten[347] wird erklärt, wie Schriftarten beim Generieren von PDF-Ausgabedokumenten verarbeitet werden.
- Diverse andere Funktionen[430] für die Arbeit mit XML-Dokumenten werden beschrieben.

Altova Website: XML Editor

© 2016-2022 Altova GmbH

5.1 Erstellen, Öffnen und Speichern von XML-Dokumenten

Beim Erstellen, Öffnen oder Speichern von XML-Dokumenten spielen die folgenden Umstände eine Rolle:

- in welcher Ansicht das XML-Dokument geöffnet wird: Textansicht, Grid-Ansicht oder Authentic-Ansicht
- bei Erstellung eines neuen XML-Dokuments: Ob ein Schema (XML-Schema oder DTD) automatisch oder manuell zugewiesen wird oder ob kein Schema zugewiesen wird
- wenn einem XML-Dokument ein Schema zugewiesen wird: Ob das Dokument beim Öffnen und/oder Speichern automatisch validiert werden soll

Standardansicht

Es gibt applikationsweite Einstellungen, mit denen festgelegt wird, in welcher Ansicht XML-Dokumente (neue und vorhandene) geöffnet werden sollen. Diese Einstellungen werden im Dialogfeld "Optionen" (**Extras | Optionen**) vorgenommen.

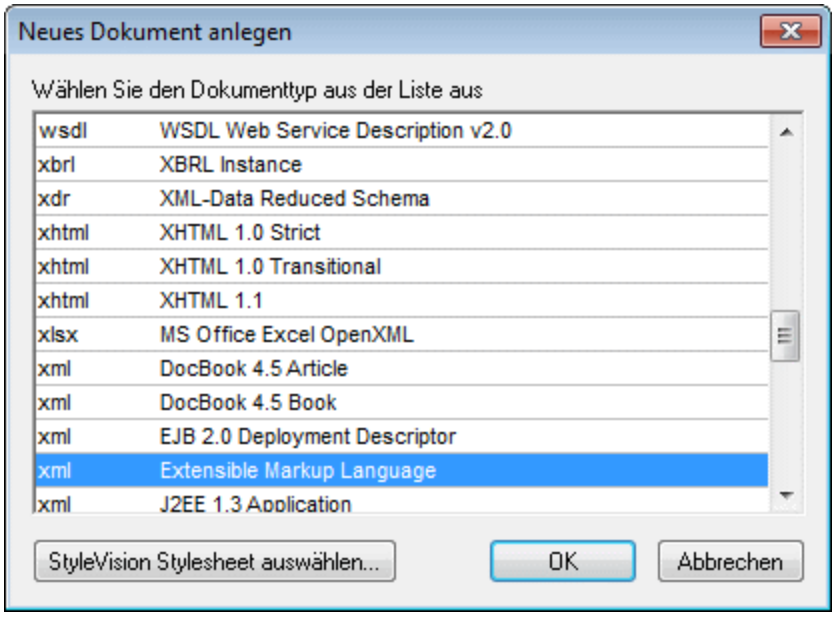
Wählen Sie im Dialogfeld "Optionen" im Abschnitt ***Dateiarten*** die Dateiart `.xml` aus und aktivieren Sie die gewünschte Bearbeitungsansicht (Text oder Grid). Beachten Sie dass: (i) die Schema- und die WSDL-Ansicht nur für XML-Schemas bzw. WSDL-Dokumente verwendet werden können; und dass (ii) die Browser-Ansicht eine reine Ansicht ist, in der Sie keine Bearbeitungen vornehmen können.

Im Abschnitt ***Dateiarten*** können Sie außerdem <%PRODUCT%> als Standardeditor für die ausgewählte Dateiart auswählen.

Ein XML-Dokument kann in der Authentic-Ansicht bearbeitet werden, wenn ihm ein StyleVision Power Stylesheet (SPS) zugewiesen wurde. Wenn eine XML-Datei, die mit einem SPS verknüpft ist, geöffnet wird, können Sie festlegen, dass sie direkt in der Authentic-Ansicht geöffnet wird. Aktivieren Sie dazu im Dialogfeld "Optionen" im Abschnitt ***Ansicht*** die Option *...Dateien immer in der Authentic-Ansicht öffnen*. Wenn diese Option nicht aktiviert ist, wird die Datei in der Standardansicht geöffnet, die im Abschnitt ***Dateiarten*** für `.xml` Dateien definiert ist (*siehe oben*).

Zuweisen von Schemas

Wenn eine neue XML-Datei erstellt werden soll, wählen Sie den Menübefehl **Datei | Neu**. Daraufhin erscheint das Dialogfeld "Neues Dokument anlegen" (*Abbildung unten*).



Beachten Sie, dass verschiedene Optionen für den XML-Dokumenttyp zur Auswahl stehen. Mit der Option *Extensible Markup Language* wird ein allgemeines XML-Dokument erstellt. Jeder der anderen Optionen ist mit einem Schema verknüpft, z.B. der DocBook-DTD. Wenn Sie eine dieser Optionen auswählen, wird ein XML-Dokument erstellt, (i) dem das entsprechende Schema automatisch zugewiesen wurde, und (ii) das eine Dokumentskelettstruktur hat, die gemäß dem zugewiesenen Schema gültig ist. Beachten Sie, dass Sie Ihr eigenes XML-Skelettdokument erstellen können. Wenn Sie es im Ordner `Template` des Applikationsordners speichern, steht es im Dialogfeld "Neues Dokument anlegen" zur Auswahl zur Verfügung.

Wenn Sie die allgemeine Dokumentart Extensible Markup Language auswählen, werden Sie aufgefordert, dem Dokument ein Schema (DTD oder XML-Schema) zuzuweisen. Zu diesem Zeitpunkt können Sie wählen, ob Sie ein Schema dafür auswählen möchten oder ob Sie weitermachen und ein XML-Dokument erstellen möchten, dem kein Schema zugewiesen ist.

Sie können dem Dokument natürlich über das Menü **DTD/Schema** jederzeit auch später während der Bearbeitung ein Schema zuweisen.

Automatische Validierung

Wenn einem vorhandenes XML-Dokument ein Schema zugewiesen ist, kann es beim Öffnen und/oder Speichern validiert werden. Die Einstellung dafür wird im Dialogfeld "Optionen" im Abschnitt ***Datei*** vorgenommen (**Extras | Optionen**).

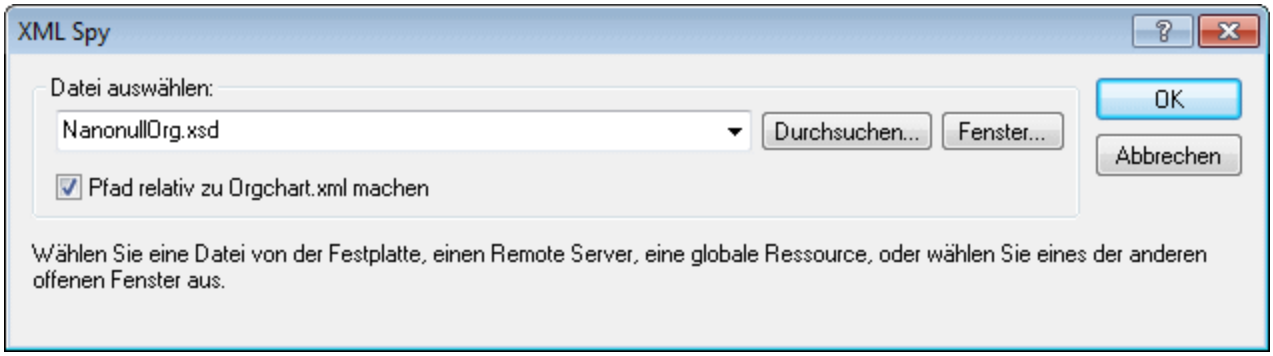
Die automatischen Validierungseinstellungen im Abschnitt ***Datei*** können mit einer Einstellung im Abschnitt "Dateiarten" kombiniert werden, sodass die automatische Validierung für bestimmte Dateiarten deaktiviert wird. Über die Einstellungen auf den beiden Registern können Sie die automatische Validierung für bestimmte Dateiarten definieren.

© 2016-2022 Altova GmbH

5.2 Zuweisen von Schemas und Validierung

Altova Website: XML Validator, XML-Validierung

Ein Schema (DTD oder XML-Schema) kann einem XML-Dokument zugewiesen werden, wenn es erstellt wird[289]. Ein Schema kann auch später jederzeit mittels der Befehle **DTD zuweisen** oder **Schema zuweisen** aus dem Menü **DTD/Schema** zugewiesen oder geändert werden.



Der Pfad zur Schemadatei, die in das XML-Dokument eingefügt wird, kann relativ gemacht werden. Aktivieren Sie dazu im Dialogfeld das entsprechende Kontrollkästchen.

Globale Ressourcen für Schemas

Eine globale Ressource ist ein Alias für eine Datei oder einen Ordner. Die Zieldatei bzw. der Zielordner kann über die GUI durch Wechseln der aktiven Konfiguration der globalen Ressource (über den Menübefehl **Extras | Aktive Konfiguration**) geändert werden. Mit Hilfe globaler Ressourcen können Sie daher zwischen verschiedenen Schemas wechseln, was v.a. für Testzwecke nützlich ist. Eine Beschreibung zur Verwendung globaler Ressourcen finden Sie im Abschnitt Globale Altova-Ressourcen[1061].

XML-Schema plus DTD

Ein sehr nützliches Feature einer DTD, das XML-Schema nicht hat, ist die Verwendung von Entities. Wenn Sie allerdings Entities in Ihren anhand eines XML-Schemas validierten XML-Dokumenten verwenden möchten, können Sie eine `DOCTYPE` Deklaration zum XML-Dokument hinzufügen und Ihre Entity-Deklarationen darin inkludieren.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE OrgChart [
  <!ENTITY name-int "value">
  <!ENTITY name-ext SYSTEM "extfile.xml">
]>
<OrgChart xmlns="http://www.xs.com/org"
          xsi:schemaLocation="http://www.xs.com/org OrgChart.xsd">
  ...
</OrgChart>
```

Nachdem Sie die Entities in der DTD deklariert haben, können sie im XML-Dokument verwendet werden. Das Dokument ist dann wohlgeformt und gültig. Beachten Sie allerdings, dass externe geparste Entities in der Authentic-Ansicht nicht unterstützt werden.

Navigieren zu Schemadefinitionen

Wenn das XML-Dokument geöffnet ist, können Sie die DTD bzw. das XML-Schema, auf der/dem es basiert, durch Klicken auf den Befehl "Gehe zu DTD" bzw. "Gehe zu Schema" im Menü **DTD/Schema** direkt öffnen. Sie können auch den Cursor in einen Node im XML-Dokument setzen und mit dem Befehl "Gehe zu Definition" aus dem Menü **DTD/Schema** zur Schemadefinition navigieren.

Validieren und Überprüfen der Wohlgeformtheit

Um ein Dokument zu validieren und/oder die Wohlgeformtheit zu prüfen, verwenden Sie den Befehl **XML validieren (F8)** bzw. **Wohlgeformtheit prüfen (F7)** aus dem Menü "XML" oder verwenden Sie die entsprechenden Befehle in der Symbolleiste. Fehler werden im Fenster "Meldungen" angezeigt. Wenn ein XML-Dokument ungültig ist, bietet der XML-Validator auf Basis der Informationen im Schema Smart Fixes zur Behebung des Fehlers an[1353].

Sie können XML-Dokumente auch mit Hilfe eines RaptorXML-Servers[1088] validieren.

© 2016-2022 Altova GmbH

5.3 XML in der Textansicht

XMLSpy bietet zusätzlich zu den unten beschriebenen allgemein verfügbaren Bearbeitungsfunktionen in der Textansicht einige spezielle XML-Textbearbeitungsfunktionen, die unter Textansicht[144] im Abschnitt "Bearbeitungsansichten" beschrieben sind.

- Ein- und Auskommentieren von Text[293]
- Versehen von XML-Zeichen mit Escape und Entfernen von Escape[293]
- Einfügen von Dateipfaden[294]
- Einfügen von XML-Fragmenten mittels XInclude[294]
- Kopieren von XPath und XPointer-Ausdrücken in die Zwischenablage[294]
- Speichern eines Base64-kodierten Bildstring als Bild[295]

Ein- und Auskommentieren von Text

Text kann in einem XML-Dokument mit Hilfe der Trennzeichen für den Beginn und das Ende eines Kommentars `<!--` bzw. `-->` auskommentiert werden. In XMLSpy können diese Kommentar-Trennzeichen ganz einfach mit dem Befehl **Bearbeiten | Ein-/Auskommentieren** eingefügt werden.

Um einen Textblock auszukommentieren, wählen Sie den Text aus und wählen Sie anschließend entweder im Menü **Bearbeiten** oder im Kontextmenü den Befehl **Ein-/Auskommentieren** aus. Der auskommentierte Text wird ausgegraut (*siehe Abbildung unten*).

```
<Department>
    <Name>Administration</Name>
    <Person>
    <Person>
    <Person>
    <!--<Person>
        <First
        <Last></Last>
        <PhoneExt></PhoneExt>
        <EMail></EMail>
        <LeaveTotal></LeaveTotal>
        <LeaveUsed></LeaveUsed>
        <LeaveLeft></LeaveLeft>
    </Person>-->
</Department>
```

Um die Auskommentierung eines Textblocks rückgängig zu machen, wählen Sie den auskommentierten Block **OHNE** die Kommentar-Trennzeichen aus und wählen Sie entweder im Menü **Bearbeiten** oder im Kontextmenü den Befehl **Ein-/Auskommentieren** aus. Die Kommentar-Trennzeichen werden entfernt und der Text ist nun nicht mehr ausgegraut.

Anmerkung zu Leerzeilen

Leere Zeilen in XML-Dokumenten werden beim Wechseln der Ansicht oder beim Speichern des Dokuments gelöscht. Wenn leere Zeilen beibehalten werden sollen, setzen Sie diese innerhalb von Kommentar-Trennzeichen.

Versehen von XML-Zeichen mit Escape und Entfernen des Escape

Die fünf XML-Sonderzeichen (*siehe Liste unten*) können mit den entsprechenden Entity-Referenzen (*siehe unten*) mit Escape versehen werden, bzw. kann das Escape aufgehoben werden. Markieren Sie dazu den

gewünschten Textblock und wählen Sie den Menübefehl **XML-Zeichen mit Escape versehen** oder **Escape für XML-Zeichen aufheben**. Je nachdem, welchen Befehl Sie ausgewählt haben, werden die XML-Sonderzeichen in diesem Textblock daraufhin mit Escape versehen oder das Escape wird aufgehoben.

<	`<`
>	`>`
&	`&`
'	`'`
"	`"`

Beispiel:

`<a></a>` kann mit dem Befehl **XML-Zeichen mit Escape versehen** als `<a></a>` dargestellt werden und
`<a></a>` kann mit dem Befehl **Escape für XML-Zeichen aufheben** als `<a></a>` angezeigt werden.

Einfügen von Dateipfaden

Mit dem Befehl **Bearbeiten | Dateipfad einfügen**[1302] können Sie zu der gesuchten Datei navigieren und den Dateipfad an der gewünschten Stelle im bearbeiteten XML-Dokument einfügen. Mit diesem Befehl können Sie einen Dateipfad schnell und korrekt eingeben. Nähere Informationen dazu finden Sie unter der Beschreibung zum Befehl[1301].

Einfügen von XML-Fragmenten mittels XInclude

Mit dem Befehl **Bearbeiten | XInclude einfügen**[1302] können Sie mittels XInclude den Inhalt eines ganzen XML-Dokuments oder ein Dokumentfragment in das bearbeitete Dokument einfügen. Mit diesem Befehl können Sie ganze XML-Dokumente (über den XInclude-Mechanismus) oder Fragmente von XML-Dokumenten (über einen XPointer-Ausdruck des XInclude-Mechanismus) schnell und einfach einfügen. Nähere Informationen dazu finden Sie unter der Beschreibung zum Befehl[1301].

Kopieren von XPath- und XPointer-Ausdrücken in die Zwischenablage

Der XPath- und der XPointer-Ausdruck des ausgewählten Node (die die Position des Node im XML-Dokument ausdrücken) können mit dem Befehl **Bearbeiten | XPath kopieren**[1301] bzw. **Bearbeiten | XPointer kopieren**[1301] in die Zwischenablage eingefügt werden. Auf diese Art erhalten Sie den korrekten XPath- bzw. XPointer-Ausdruck, der einen bestimmten Node als Ziel hat.

Angenommen beim ausgewählten Node in der Text- oder Grid-Ansicht handelt es sich um das dritte `Office` Element eines Dokuments namens `Offices`. In diesem Fall lautet der kopierte XPath-Ausdruck `/Offices/Ofice[3]`. Und wenn die `Office` Elemente kein anderes gleichrangiges Element haben, das vor dem dritten `Office` Element steht, so lautet der kopierte XPointer-Ausdruck `element(/1/3)`.

Die kopierten Ausdrücke können anschließend an jeder gewünschten Stelle eingefügt werden. So kann z.B. ein XPath-Ausdruck in ein XSLT-Stylesheet und ein XPointer-Ausdruck in das `href` Attribut eines `xinclude` Elements eingefügt werden.

Eine nähere Beschreibung zu den Befehlen finden Sie in den entsprechenden Abschnitten zu diesen Befehlen im Abschnitt "Menübefehle".

© 2016-2022 Altova GmbH

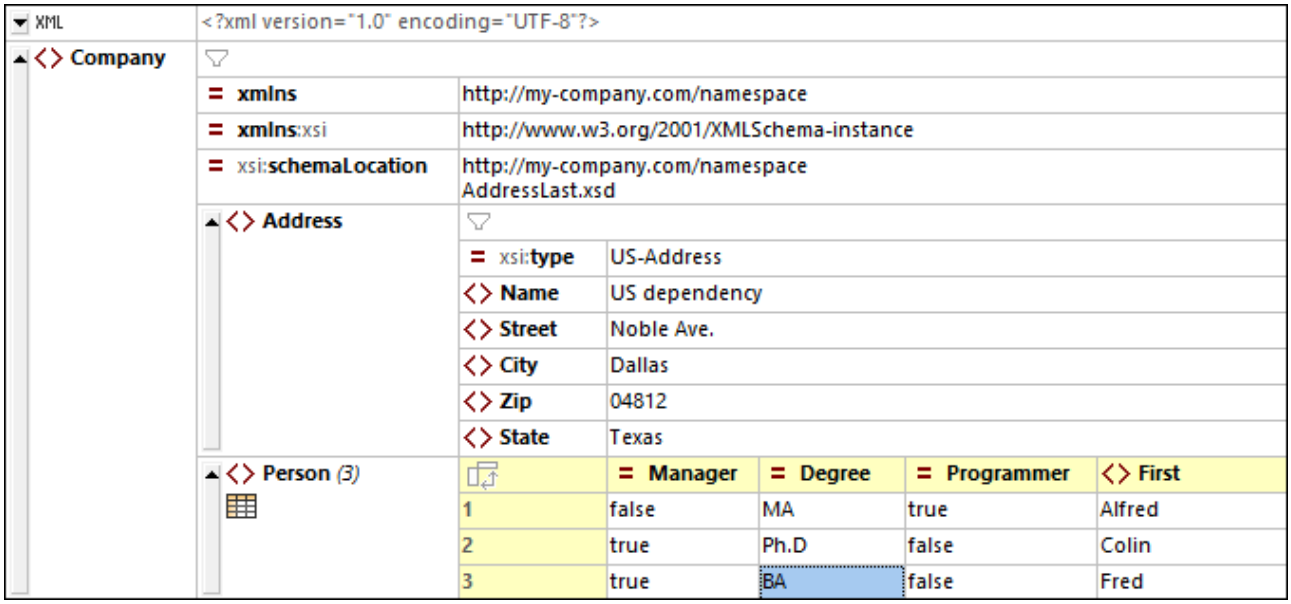
Speichern eines Base64-kodierten String als Bild

Um einen Base64-kodierten String in seinem Bildformat zu speichern, klicken Sie mit der rechten Maustaste auf das Bild oder dessen Zelle und wählen Sie den Befehl **Als Bild speichern**. Wählen Sie im daraufhin angezeigten Dialogfeld den Pfad, unter dem Sie das Bild speichern möchten und geben Sie einen Namen für die Bilddatei ein. Die Erweiterung der Bilddatei (`.png`, `.gif`, `.svg`, usw.) wird automatisch anhand der Base64-Kodierung ermittelt und im Speichern-Dialogfeld angezeigt. Klicken Sie anschließend auf **Speichern**.

Diese Aktion kann auch über den Menübefehl **Bearbeiten | Als Bild speichern** durchgeführt werden.

5.4 XML in der Grid-Ansicht

In der Grid-Ansicht[169] wird die hierarchische Struktur von **XML-Dokumenten** in Form einer Reihe von verschachtelten Behältern angezeigt, die erweitert und reduziert werden können. Dadurch wird die Struktur des Dokuments klar und übersichtlich dargestellt. In der Grid-Ansicht kann sowohl die Struktur als auch der Inhalt des Dokuments einfach bearbeitet werden.



Beachten Sie in der Abbildung oben, dass das Dokument als Hierarchie in Form eines Rasters angezeigt wird. Wenn ein Node Inhalt enthalten kann, enthält er diesen entweder direkt (wie im Fall von *Text*-Nodes) oder der Inhalt ist in zwei Felder unterteilt: den Node-Namen und den Node-Inhalt (wie im Fall von *Element*-Nodes). Node-Namen werden fett und Node-Inhalt in normaler Schrift angezeigt.

Symbolleiste der Grid-Ansicht

Befehle im Zusammenhang mit der Grid-Ansicht können schnell über die Grid-Ansichts-Symbolleiste *(Abbildung unten)* aufgerufen werden. Diese Symbolleiste befindet sich im Symbolleistenbereich am oberen Rand des Applikationsfensters. Wenn Sie die Maus über eine Symbolleisten-Schaltfläche platzieren, sehen Sie deren Namen und das dazugehörige Tastaturkürzel.



Anmerkung: Diese Befehle stehen auch im Menü "XML"[1348] zur Verfügung.

Übersicht über die Funktionalitäten der Grid-Ansicht

In den folgenden Abschnitten finden Sie eine Übersicht über die Funktionalitäten der Grid-Ansicht:

- Dokumentanzeige[298]: Hier werden die Anzeigefunktionen der Grid-Ansicht beschrieben.

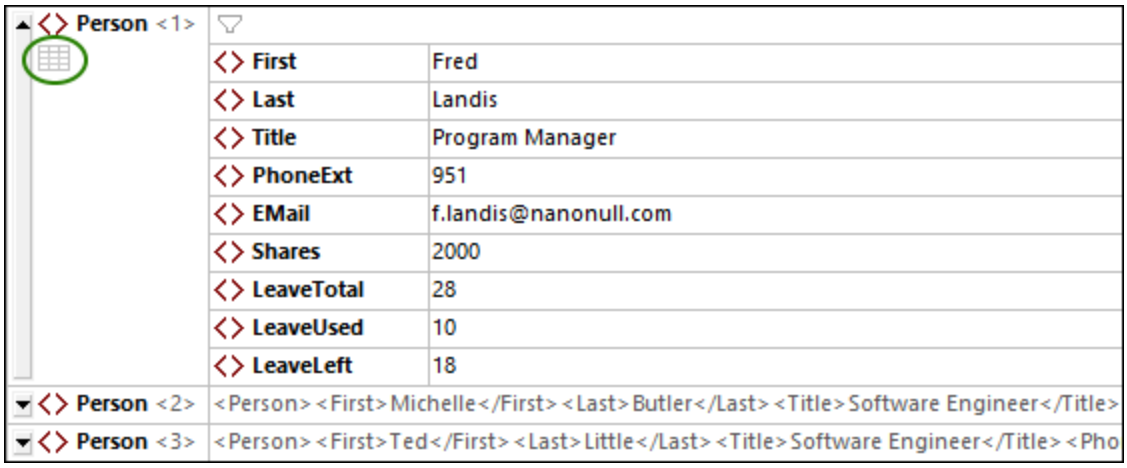
© 2016-2022 Altova GmbH

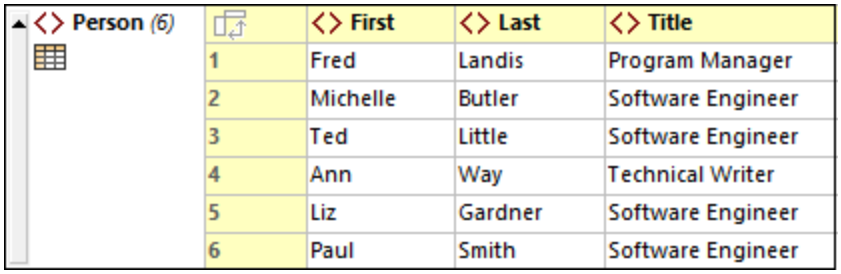
- Dokumentstruktur[302]: Hier wird beschrieben, wie die Dokumentstruktur in der Grid-Ansicht angezeigt wird und es wird erläutert, mit welchen Funktionalitäten der Grid-Ansicht Sie die Dokumentstruktur bearbeiten können.
- Dokumentinhalt[305]: Hier wird beschrieben, wie Sie Inhalt in der Grid-Ansicht bearbeiten und validieren können.
- Einstellungen der Grid-Ansicht[329]: Hier werden die Einstellungen für die Grid-Ansicht aufgelistet und beschrieben.

Als Tabelle anzeigen

Sich wiederholende Elemente werden in der *Standard*-Grid-Ansicht nacheinander von oben nach unten in der Dokumentreihenfolge angezeigt (*siehe Abbildung links unten*). Durch Anzeige sich wiederholender Elemente als *Zeilen einer Tabelle* stehen Ihnen jedoch zusätzliche Bearbeitungsfunktionen zur Verfügung. In der Abbildung unten ist das Element **Person** das sich wiederholende Element. In der Abbildung links sehen Sie die Standard-Grid-Ansicht; das erste **Person**-Element wird erweitert angezeigt, während die darauf folgenden Instanzen reduziert angezeigt werden. In der Abbildung rechts werden die sich wiederholenden **Person**-Elemente als Zeilen einer Tabelle angezeigt.

Um in die Tabellenansicht zu wechseln, klicken Sie auf das **Tabellenmodus**-Symbol *(in der Abbildung links unten grün umrandet)*. Wenn Sie sich in der Tabellenansicht befinden, wird das Symbol färbig angezeigt *(siehe Abbildung rechts unten)*.





Die Tabellenansicht bietet beim Bearbeiten den einzigartigen Vorteil, dass ganze Zeilen und Spalten relativ zu anderen Spalten und Zeilen in der Tabelle bearbeitet werden können. Auf diese Art können Tabellenzeilen nach den Werten einer Spalte sortiert werden. So können etwa in der Abbildung rechts oben die sechs **Person**-Elemente anhand ihres Child-Elements **Last** mit einem einzige Befehl aus der Grid-Ansichts-Symbolleiste[296] sortiert werden. Dies ist einfacher, als ein XML-Nodeset auf die übliche Weise mit Hilfe einer XSLT-Transformation zu sortieren.

Nähere Informationen finden Sie im Abschnitt Tabellenansicht[312].

Filter und Formeln

Die Grid-Ansicht bietet zwei einzigartige und äußerst nützliche Funktionalitäten:

- Mit Hilfe von Filtern[317] können Sie die in der Grid-Ansicht angezeigten Daten filtern. Es können für ein oder mehrere der Elemente im Dokumente Filter definiert werden.
- Mit Hilfe von Formeln[319] kann ein Ergebnis berechnet oder ein Nodeset generiert werden, das im Dokument gespeichert werden kann.

Nähere Informationen dazu finden Sie im jeweiligen Kapitel.

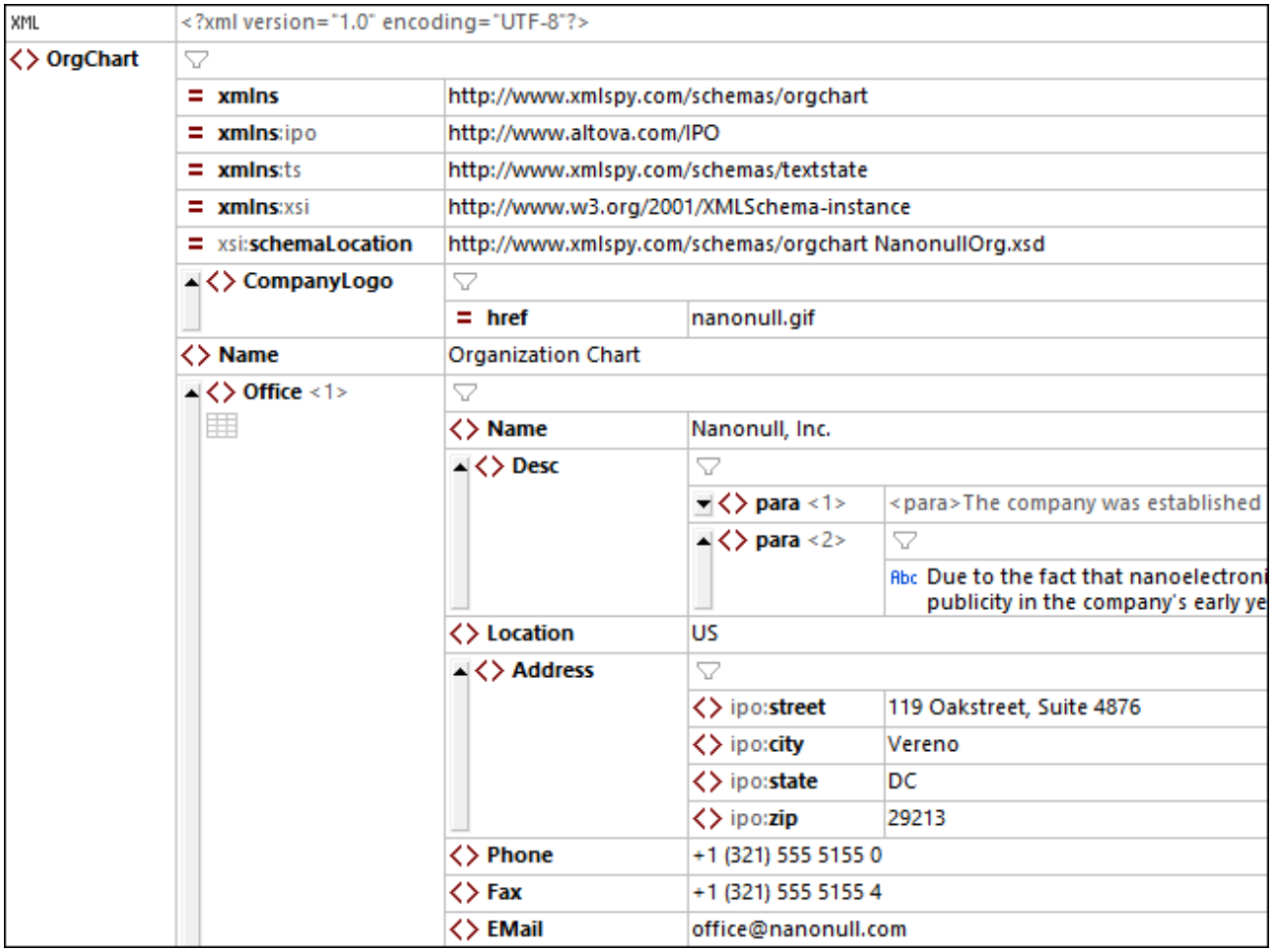
Bilder und Diagramme in der Grid-Ansicht

Bilder und Diagramme können in der Grid-Ansicht in grafischer Form angezeigt werden. Eine Anleitung dazu finden Sie unter Anzeige von Bildern in der Grid-Ansicht[322] und Diagramme in der Grid-Ansicht[323].

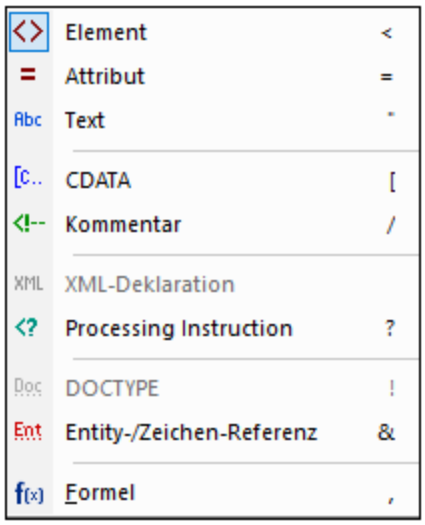
5.4.1 Dokumentanzeige

In der Grid-Ansicht wird das XML-Dokument hierarchisch innerhalb eines Rasters (*siehe Abbildung unten*) angezeigt.

© 2016-2022 Altova GmbH



In jeder Zeile des Rasters wird jeweils ein XML-Strukturelement (ein so genannter Node) wie z.B. ein Element, Attribut, Kommentar oder Text angezeigt. In der Abbildung unten werden die verschiedenen in XML-Dokumenten verfügbaren Node-Typen zusammen mit den dazugehörigen Symbolen angezeigt. Der Typ eines Node wird in der Grid-Ansicht durch ein Symbol auf der linke Seite der Rasterzelle gekennzeichnet. Beachten Sie, dass Nodes vom Typ *Element, Attribut* und *Processing Instruction* einen Schlüssel und einen Wert haben, während Nodes vom Typ *Text, CDATA* und *Comment* nur einen Wert haben. So hat etwa ein Element-Node einen Namen (seinen Schlüssel) sowie Inhalt (seinen Wert), während ein Text-Node nur einen Wert aufweist.



Funktionalitäten der Grid-Ansicht

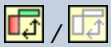
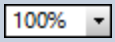
Beachten Sie die folgenden Funktionalitäten:

- Sie können mit Hilfe von **Strg + Mausrad** oder **Strg+[Plus]** und **Strg+[Minus]** in das Raster hinein- bzw. daraus hinauszoomen.
- Gleichrangige Elemente, die denselben Namen haben, werden beginnend mit 1 nummeriert. So ist z.B. das Element `office` mit der Nummer 1 in der Abbildung oben das erste in einer Sequenz von `office`-Elementen. Die anderen Elemente in der Abbildung oben sind nicht nummeriert, da sie keine gleichrangigen Elemente desselben Namens haben.
- Nodes können durch Klick auf den Pfeil links vom Node-Symbol (*siehe Abbildung oben*) erweitert und reduziert werden. Wenn ein Node reduziert wurde, wird sein Inhalt als Text in einer einzigen Zeile angezeigt.
- Wenn Sie mehrere Komponenten auf derselben Ebene auswählen, können Sie alle davon durch Drücken der **Umschalttaste** und Klick auf die Pfeilspitze einer der ausgewählten Komponenten erweitern/reduzieren.
- Wenn über die Grid-Ansichts-Symbolleiste[296] ein Zeilenumbruch aktiviert wurde, werden alle Zellen, die Text enthalten, der die Breite der Zelle übersteigt, umbrochen. Sie können den Zeilenumbruch durch Klick auf das Zeilenumbruch-Symbol in der Grid-Ansichts-Symbolleiste[296] deaktivieren.
- Formeln[709] sind XMLSpy-spezifisch.

Symbolleistenschaltflächen/Befehlsschaltflächen innerhalb von Zellen

In der XML-Grid-Ansicht stellen Schaltflächen in den Zellen des Rasters oder in der Grid-Ansichts-Symbolleiste[296] Funktionen bereit, die im Kontext der jeweiligen Auswahl (Zelle oder Spalte) zur Verfügung stehen. Häufig verwendete Operationen können dadurch schneller und einfacher aufgerufen werden. In der Tabelle unten sehen Sie eine Liste der verfügbaren Befehle. Befehle im Zusammenhang mit der Anzeige haben einen blauen Hintergrund. Befehle für die Bearbeitung des Dokuments (Struktur und Inhalt) haben einen beigen Hintergrund. Zusätzliche Befehle im Zusammenhang mit einer bestimmten Zelle können Sie über das Kontextmenü[718] der Zelle aufrufen. Anmerkung: Der Name eines Befehls wird als Tooltipp angezeigt, wenn Sie den Mauszeiger über die Zellen- oder Symbolleisten-Schaltfläche platzieren.

© 2016-2022 Altova GmbH

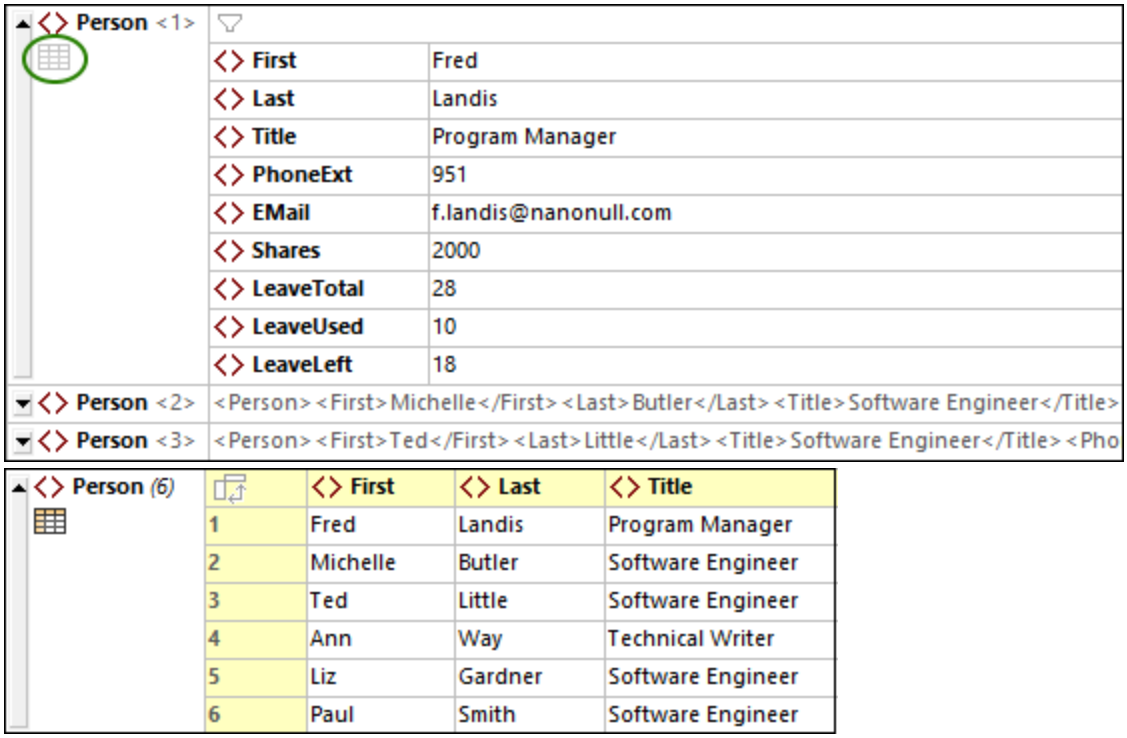
	Optimiert die Breite von Grid-Spalten entsprechend dem Zelleninhalt.
	Bei farbiger Anzeige ist die Tabellenansicht[704] aktiviert, andernfalls ist sie deaktiviert. Klicken Sie darauf, um die Anzeige zu wechseln.
	Ein Ein/Aus-Befehl in der linken oberen Zelle einer Tabelle. Zeigt Zeilen als Spalten an und umgekehrt.
	Wenn das Symbol farbig angezeigt wird, ist ein Filter[709] für die Tabelle, das Objekt oder Array aktiv. Andernfalls ist der Filter deaktiviert. Klicken Sie darauf, um den Filter zu deaktivieren/zu aktivieren. Um den Ausdruck zu bearbeiten, doppelklicken Sie darauf.
	Ein/Aus-Befehl für den Zeilenumbruch in allen Zellen. Wenn die Schaltfläche ausgewählt ist, ist der Zeilenumbruch aktiviert. Standardmäßig wird nur der Inhalt von Elementen umbrochen. Wenn auch die Namen von Elementen umbrochen werden sollen, wählen Sie diese Option aus der Dropdown-Liste des Symbols aus.
	Zoomfaktor der Grid-Ansicht.
	Ist aktiv, wenn eine Spaltenüberschrift in der Tabellenansicht ausgewählt ist. Mit diesen Schaltflächen werden die Zeilen der Tabelle nach dem Spalteninhalt in auf- bzw. absteigender Reihenfolge sortiert.

Anmerkung: Eine Beschreibung weiterer Symbolleistenbefehle finden Sie im Abschnitt Kontextmenü in der Grid-Ansicht[326].

Als Tabelle anzeigen

Sich wiederholende Elemente werden in der *Standard*-Grid-Ansicht nacheinander von oben nach unten in der Dokumentreihenfolge angezeigt (*siehe Abbildung links unten*). Durch Anzeige sich wiederholender Elemente als *Zeilen einer Tabelle* stehen Ihnen jedoch zusätzliche Bearbeitungsfunktionen zur Verfügung. In der Abbildung unten ist das Element **`Person`** das sich wiederholende Element. In der Abbildung links sehen Sie die Standard-Grid-Ansicht; das erste **`Person`**-Element wird erweitert angezeigt, während die darauf folgenden Instanzen reduziert angezeigt werden. In der Abbildung rechts werden die sich wiederholenden **`Person`**-Elemente als Zeilen einer Tabelle angezeigt.

Um in die Tabellenansicht zu wechseln, klicken Sie auf das **Tabellenmodus**-Symbol *(in der Abbildung links unten grün umrandet)*. Wenn Sie sich in der Tabellenansicht befinden, wird das Symbol färbig angezeigt *(siehe Abbildung rechts unten)*.



Die Tabellenansicht bietet beim Bearbeiten den einzigartigen Vorteil, dass ganze Zeilen und Spalten relativ zu anderen Spalten und Zeilen in der Tabelle bearbeitet werden können. Auf diese Art können Tabellenzeilen nach den Werten einer Spalte sortiert werden. So können etwa in der Abbildung rechts oben die sechs **`Person`**-Elemente anhand ihres Child-Elements **`Last`** mit einem einzige Befehl aus der Grid-Ansichts-Symbolleiste[296] sortiert werden. Dies ist einfacher, als ein XML-Nodeset auf die übliche Weise mit Hilfe einer XSLT-Transformation zu sortieren.

Nähere Informationen finden Sie im Abschnitt Tabellenansicht[312].

5.4.2 Dokumentstruktur

In der Grid-Ansicht kann die XML-Dokumentstruktur grafisch bearbeitet werden. So können Sie etwa einzelne Segmente der Dokumentstruktur reduzieren und erweitern, Nodes einfügen, anhängen und löschen, Nodes mit der Maus an eine andere Stelle ziehen und einen Node-Typ in einen anderen konvertieren.

Hinzufügen neuer Nodes

Es gibt zwei Methoden, um neue Nodes zum Dokument hinzuzufügen:

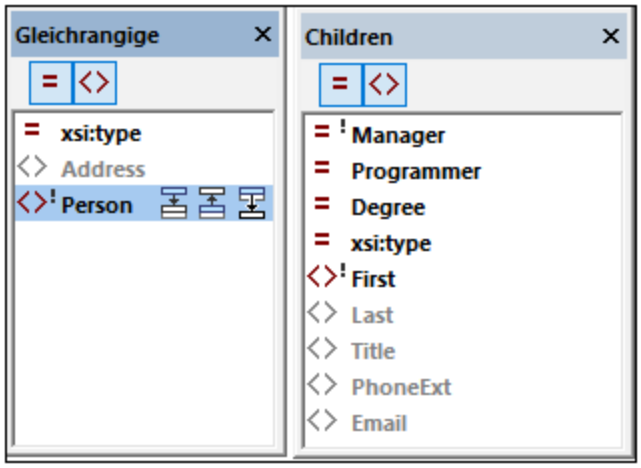
- Über die Eingabehilfen "Gleichrangige" und "Children" können bestimmte Nodes an der ausgewählten Stelle eingefügt werden. Dazu muss ein XML-Dokument mit einem Schema (DTD oder XML-Schema) verknüpft sein. Wenn Sie z.B. im Hauptfenster einen Node auswählen, werden in den Eingabehilfen "Gleichrangige" und "Children" die gleichrangigen Elemente bzw. Child-Elemente, die (dem Schema gemäß) an dieser Stelle zulässig sind, angezeigt. Sie können den gewünschten Node über die Eingabehilfen auswählen.

 © 2016-2022 Altova GmbH

- Sie können einen neuen, nicht in einem Schema definierten Node als gleichrangiges Element, Child oder Parent hinzufügen. Die Befehle dazu stehen in den Kontextmenüs von Zellen der Grid-Ansicht sowie im Menü "XML" 1348 und in der Grid-Ansichts-Symbolleiste 296 zur Verfügung.

Eingabehilfen "Gleichrangige" und "Children"

Wenn ein XML-Dokument mit einem Schema (DTD oder XML-Schema) verknüpft ist, werden die verfügbaren gleichrangigen Elemente und Children eines ausgewählten Nodes in der Eingabehilfe "Gleichrangige" bzw. "Children" angezeigt. Durch Klick auf die entsprechenden Symbolleisten-Schaltflächen können Sie die Sichtbarkeit von Elementen und Attributen in jeder dieser Eingabehilfen aktivieren und deaktivieren. Obligatorische Nodes werden mit einem Ausrufezeichen angezeigt. Grau angezeigte Nodes können nicht hinzugefügt werden. Dies liegt an einem der folgenden Gründe: (i) Der Node wurde bereits hinzugefügt und gemäß dem Schema sind keine weiteren Instanzen zulässig, wie z.B. im Fall des gleichrangigen Elements `Address` in der Abbildung unten; (ii) es muss ein weiterer Node hinzugefügt werden, bevor der ausgegraute Node hinzugefügt werden kann, wie z.B. im Fall des Child-Elements `Last` in der Abbildung unten, das erst hinzugefügt werden kann, nachdem das Child-Element `First` hinzugefügt wurde.



Fügen Sie Nodes auf folgende Art hinzu:

- *Gleichrangige:* Wählen Sie in der Eingabehilfe den Node aus, den Sie als gleichrangigen Node hinzufügen möchten. Klicken Sie anschließend, je nachdem, ob Sie den gleichrangigen Node vor oder nach dem im Raster ausgewählten Node hinzufügen oder als letzten Node an die gleichrangigen Nodes anhängen möchten, auf das entsprechende Symbol (*siehe Abbildung links oben*).
- *Children:* Doppelklicken sie auf den Node, den Sie als Child-Node hinzufügen möchten.

Einfügen neuer Nodes

Wenn ein Node im Dokument (d.h. eine Rasterzelle) ausgewählt wurde, können Sie einen neuen leeren Node als gleichrangigen, Child- oder Parent-Node hinzufügen. Die Befehle für diese Operationen sind in der Tabelle unten aufgelistet und stehen im (i) Kontextmenü der Zelle, (ii) dem Menü "XML" 1348 und (iii) der Grid-Ansichts-Symbolleiste 296 zur Verfügung.

Befehl	Tastaturkürzel
(Gleichrangiges Element) einfügen nach	Strg+Eingabetaste

(Gleichrangiges Element) einfügen vor	Strg+Umschalt+Eingabetaste
(Gleichrangiges) anhängen	Strg+Alt+A
Child hinzufügen	Strg+Alt+Eingabetaste
In Element verpacken	Strg+Alt+ W

Beachten Sie die folgenden Punkte:

- Der neue Node wird standardmäßig als leeres Element erstellt. Sie können den Node-Typ später ändern 305.
- Der Befehl **In Element verpacken** erstellt rund um den aktuellen Node herum einen Element-Node. Dieses Element wird zum neuen Parent des aktuellen Node.

Ändern der Struktur mit Hilfe von Standard-Windows-Methoden

Sie können die Dokumentstruktur in der Grid-Ansicht auch mit Hilfe der folgenden Windows-Methoden ändern:

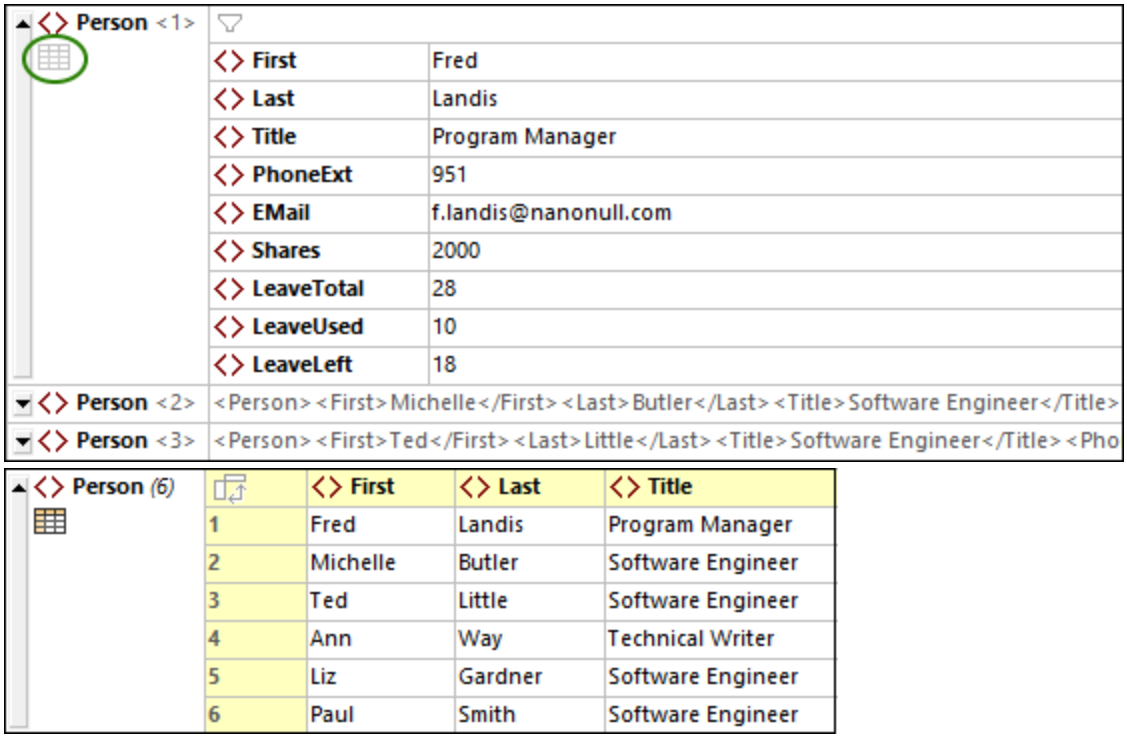
- *Löschen:* Wählen Sie eine Komponente aus und löschen Sie sie mit der **Entf**-Taste.
- *Verschieben:* Wählen Sie eine Komponente aus und ziehen Sie sie mit der Maus an eine andere Stelle.

Als Tabelle anzeigen

Sich wiederholende Elemente werden in der *Standard*-Grid-Ansicht nacheinander von oben nach unten in der Dokumentreihenfolge angezeigt (*siehe Abbildung links unten*). Durch Anzeige sich wiederholender Elemente als *Zeilen einer Tabelle* stehen Ihnen jedoch zusätzliche Bearbeitungsfunktionen zur Verfügung. In der Abbildung unten ist das Element **`Person`** das sich wiederholende Element. In der Abbildung links sehen Sie die Standard-Grid-Ansicht; das erste **`Person`**-Element wird erweitert angezeigt, während die darauf folgenden Instanzen reduziert angezeigt werden. In der Abbildung rechts werden die sich wiederholenden **`Person`**-Elemente als Zeilen einer Tabelle angezeigt.

Um in die Tabellenansicht zu wechseln, klicken Sie auf das **Tabellenmodus**-Symbol *(in der Abbildung links unten grün umrandet)*. Wenn Sie sich in der Tabellenansicht befinden, wird das Symbol färbig angezeigt *(siehe Abbildung rechts unten)*.

© 2016-2022 Altova GmbH



Person <1>	
First	Fred
Last	Landis
Title	Program Manager
PhoneExt	951
EMail	f.landis@nanonull.com
Shares	2000
LeaveTotal	28
LeaveUsed	10
LeaveLeft	18
Person <2>	<Person><First>Michelle</First><Last>Butler</Last><Title>Software Engineer</Title>
Person <3>	<Person><First>Ted</First><Last>Little</Last><Title>Software Engineer</Title><Pho

Person (6)	First	Last	Title
1	Fred	Landis	Program Manager
2	Michelle	Butler	Software Engineer
3	Ted	Little	Software Engineer
4	Ann	Way	Technical Writer
5	Liz	Gardner	Software Engineer
6	Paul	Smith	Software Engineer

Die Tabellenansicht bietet beim Bearbeiten den einzigartigen Vorteil, dass ganze Zeilen und Spalten relativ zu anderen Spalten und Zeilen in der Tabelle bearbeitet werden können. Auf diese Art können Tabellenzeilen nach den Werten einer Spalte sortiert werden. So können etwa in der Abbildung rechts oben die sechs **`Person`**-Elemente anhand ihres Child-Elements **`Last`** mit einem einzige Befehl aus der Grid-Ansichts-Symbolleiste 296 sortiert werden. Dies ist einfacher, als ein XML-Nodeset auf die übliche Weise mit Hilfe einer XSLT-Transformation zu sortieren.

Nähere Informationen finden Sie im Abschnitt Tabellenansicht 312.

5.4.3 Dokumentinhalt

XML-Inhalte lassen sich in der Grid-Ansicht ganz einfach bearbeiten: Doppelklicken Sie einfach in das Inhaltsfeld und bearbeiten Sie den Inhalt nach Bedarf. Der Node-Typ kann ganz einfach durch Klick auf das Symbol des Node-Typs und Auswahl eines anderen Typs aus dem daraufhin angezeigten Menü geändert werden. Entities können über die Eingabehilfe "Entities" eingefügt werden. Die Grid-Ansicht bietet zusätzlich Validierungs- und Such- und Ersetzungsfunktionen.

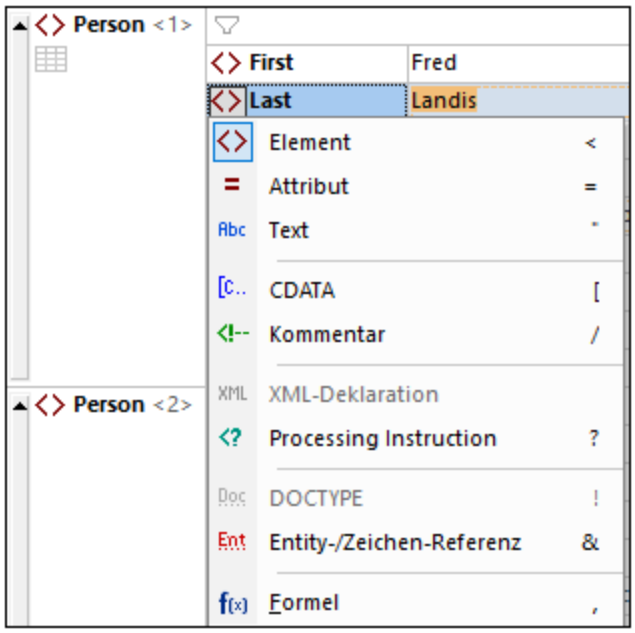
Auswahl und Konvertierung des Node-Typs

Die Zellen in der Grid-Ansicht enthalten Nodes des XML-Dokuments. Wir haben die Typen zum besseren Verständnis folgendermaßen in Gruppen eingeteilt:

- *Name-Wert-Typen:* Element, Attribut, Processing Instruction (PI). Nodes dieses Typs haben Namen und Werte.

- *Wert-Typen:* Text, CDATA, Comment (Kommentar), Entity/Zeichen-Referenzen. Nodes dieses Typs haben nur Werte.
- *Definitionstypen:* XML-Deklaration, DOCTYPE (interne oder externe DTD). Diese Nodes definieren Eigenschaften des XML-Dokuments.
- *Spezielle, XMLSpy-spezifische Container-Typen:* Formel

Der Typ eines Werts wird durch ein Symbol vor dem Wert angezeigt (*siehe Abbildung unten*). Um einen Typ zu ändern, klicken Sie auf sein Symbol und wählen Sie den gewünschten Typ aus dem angezeigten Menü aus. Klicken Sie alternativ dazu mit der rechten Maustaste in eine Zelle und wählen Sie aus dem daraufhin angezeigten Kontextmenü einen Typ aus dem Untermenü **Typ** aus. In der Abbildung unten sehen Sie die Symbole und Shortcuts von Typen:



Beachten Sie die folgenden Punkte:

- Nodes vom Typ *Element, Attribut* und *PI* erhalten einen Namen und einen Wert, während Nodes vom Typ *Text, CDATA* und *Comment* nur einen Wert haben. So hat etwa ein Element-Node einen Namen sowie Inhalt, während ein Text-Node nur einen Wert aufweist.
- Bei einer Typkonvertierung wird versucht, den ursprünglichen Schlüssel und Wert beizubehalten. Wenn Sie z.B. ein Element in ein Attribut konvertieren, erhält das Attribut (i) denselben Namen wie das Element und (ii) einen Wert, der so weit wie möglich mit dem Inhalt des Elements übereinstimmt.
- In der Reihenfolge der Child-Nodes eines Elements werden Attribute immer zuerst aufgelistet. Eine Typkonvertierung kann infolgedessen zu einer Umordnung von Nodes führen.
- Bei gemischtem Inhalt (Zeichendaten gemischt mit Element-Children, wie z.B. ein Absatzelement, das sowohl Text als auch fette und kursive Elemente enthält) sind aufeinander folgende Text-Nodes nicht zulässig und werden daher eventuell automatisch miteinander verbunden.
- Sie können eine Option definieren, um zu ermitteln, ob eine Änderung bei Auswahl mehrerer Nodes für die Konvertierung in einen einzigen Typ durchgeführt werden soll oder nicht, oder ob Sie gewarnt werden möchten.
- Reiner Text in einer Zelle oder Zeile kann durch Auswahl der übergeordneten Zelle oder einer übergeordneten Zeile und Klick auf die Symbolleisten-Schaltfläche **Als reinen Text bearbeiten** bearbeitet werden. Wenn Sie Text als reinen Text bearbeiten, werden Entities und Markup-Code in

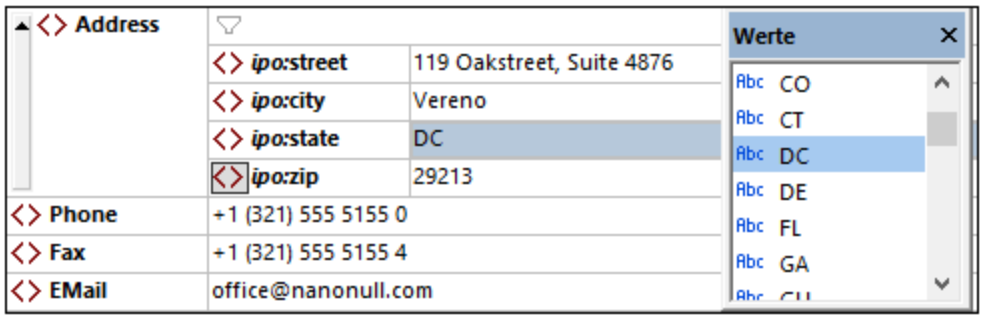
© 2016-2022 Altova GmbH

dieser Zelle oder Zeile nicht zu Glyphen und Komponenten der Grid-Ansicht aufgelöst (und können dadurch bearbeitet werden).

Autokomplettierung

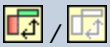
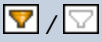
Die Autokomplettierung ist aktiv, wenn das bearbeitete XML-Dokument mit einem Schema verknüpft ist.

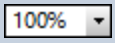
Bei der Autokomplettierung werden Ihnen an der Cursorposition Eingabeoptionen vorgeschlagen (*siehe Abbildung unten*). Diese auf den Definitionen im Schema basierenden Optionen werden (i) über Popups im Hauptfenster und (ii) über die Eingabehilfen vorgeschlagen. In den Pop-ups und Eingabehilfen wird jeweils eine Liste von Einträgen angezeigt, die an dieser Cursorposition gültig sind. Mit Hilfe der Pfeiltasten können Sie durch die Einträge in der Pop-up-Liste navigieren. Wählen Sie einen Eintrag aus dem Pop-up-Fenster aus oder doppelklicken Sie in der Eingabehilfe auf einen Eintrag, um diesen einzufügen.



Symbolleistenschaltflächen/Befehlsschaltflächen innerhalb von Zellen

In der XML-Grid-Ansicht stellen Schaltflächen in den Zellen des Rasters oder in der Grid-Ansichts-Symbolleiste[296] Funktionen bereit, die im Kontext der jeweiligen Auswahl (Zelle oder Spalte) zur Verfügung stehen. Häufig verwendete Operationen können dadurch schneller und einfacher aufgerufen werden. In der Tabelle unten sehen Sie eine Liste der verfügbaren Befehle. Befehle im Zusammenhang mit der Anzeige haben einen blauen Hintergrund. Befehle für die Bearbeitung des Dokuments (Struktur und Inhalt) haben einen beigen Hintergrund. Zusätzliche Befehle im Zusammenhang mit einer bestimmten Zelle können Sie über das Kontextmenü[718] der Zelle aufrufen. Anmerkung: Der Name eines Befehls wird als Tooltipp angezeigt, wenn Sie den Mauszeiger über die Zellen- oder Symbolleisten-Schaltfläche platzieren.

	Optimiert die Breite von Grid-Spalten entsprechend dem Zelleninhalt.
	Bei farbiger Anzeige ist die Tabellenansicht[704] aktiviert, andernfalls ist sie deaktiviert. Klicken Sie darauf, um die Anzeige zu wechseln.
	Ein Ein/Aus-Befehl in der linken oberen Zelle einer Tabelle. Zeigt Zeilen als Spalten an und umgekehrt.
	Wenn das Symbol farbig angezeigt wird, ist ein Filter[709] für die Tabelle, das Objekt oder Array aktiv. Andernfalls ist der Filter deaktiviert. Klicken Sie darauf, um den Filter zu deaktivieren/zu aktivieren. Um den Ausdruck zu bearbeiten, doppelklicken Sie darauf.
	Ein/Aus-Befehl für den Zeilenumbruch in allen Zellen. Wenn die Schaltfläche ausgewählt ist, ist der Zeilenumbruch aktiviert. Standardmäßig wird nur der Inhalt von Elementen umbrochen. Wenn auch die Namen von Elementen umbrochen werden sollen, wählen Sie diese Option aus der Dropdown-Liste des Symbols aus.

	Zoomfaktor der Grid-Ansicht.
	Ist aktiv, wenn eine Spaltenüberschrift in der Tabellenansicht ausgewählt ist. Mit diesen Schaltflächen werden die Zeilen der Tabelle nach dem Spalteninhalt in auf- bzw. absteigender Reihenfolge sortiert.

Anmerkung: Eine Beschreibung weiterer Symbolleistenbefehle finden Sie im Abschnitt Kontextmenü in der Grid-Ansicht 326.

Bei Änderung validieren

Der *Bei Bearbeitung validieren*-Modus ist standardmäßig deaktiviert. Wenn dieser Modus aktiv ist, werden, während Sie ein Dokument der-Grid-Ansicht bearbeiten, Wohlgeformtheits- und Validierungsüberprüfungen durchgeführt. Damit ein Dokument (nicht nur auf Wohlgeformtheit geprüft, sondern auch) validiert werden kann, muss ihm eine DTD oder ein XML-Schema zugewiesen sein. Fehler im Text werden rot angezeigt. Die Stelle im Dokument wird durch ein rotes Ausrufezeichen markiert. Falls es zu einem Fehler einen Smart Fix-Vorschlag gibt, wird in der Zeile mit dem Fehler ein Glühbirnensymbol angezeigt. Wenn Sie den Mauszeiger über das Symbol platzieren, erscheint ein Popup-Fenster, in dem Smart Fix-Vorschläge aufgelistet werden. Wählen Sie einen Vorschlag aus, um ihn sofort anzuwenden. Nähere Informationen dazu finden Sie unter Validieren eines XML-Dokuments 339.

Der *Bei Bearbeitung validieren*-Modus kann entweder (i) über den Menübefehl **XML | Bei Bearbeitung validieren** 1359, (ii) die Symbolleisten-Schaltfläche **Bei Bearbeitung validieren** oder (iii) die Option *Bei Bearbeitung* der Validierungseinstellungen des Dialogfelds "Optionen" 1603 aktiviert bzw. deaktiviert werden.

Suchen und Ersetzen

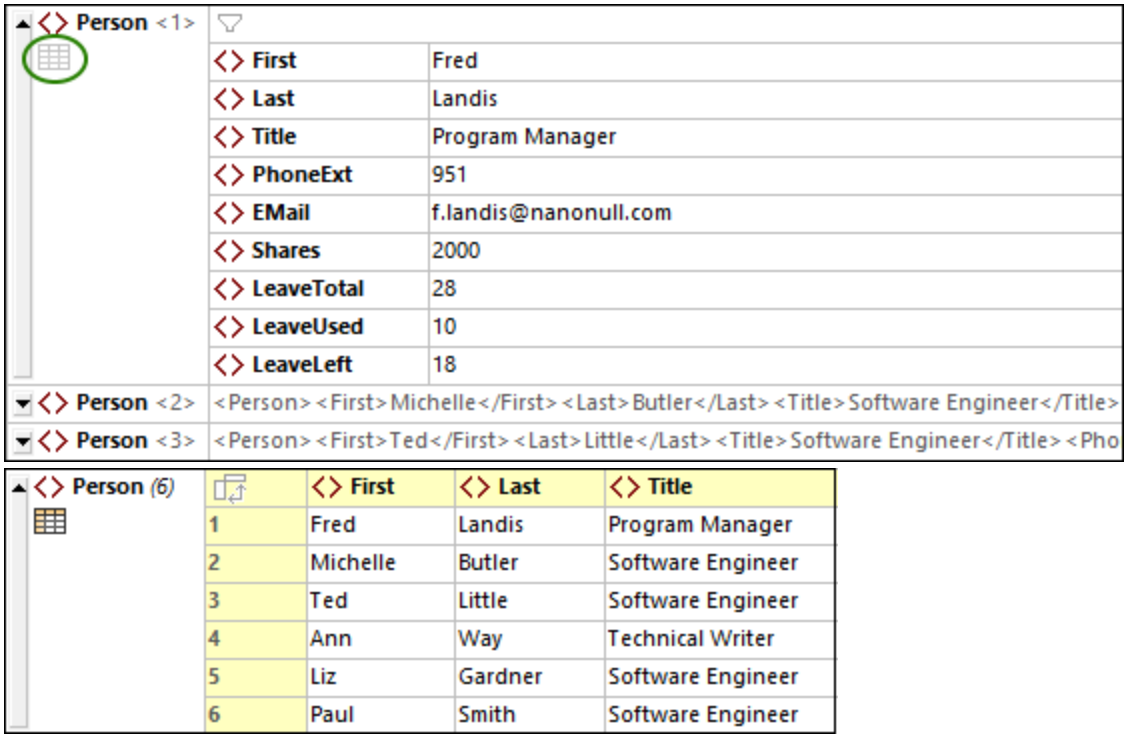
Mit Hilfe der Befehle **Suchen (Strg+F)** und **Ersetzen (Strg+H)** aus dem Menü **Bearbeiten** können Sie nach Text in der Grid-Ansicht suchen und diesen ersetzen. Die Suchergebnisse werden orange markiert, wobei auch Zellen, die diese enthalten, orange markiert werden.

Als Tabelle anzeigen

Sich wiederholende Elemente werden in der *Standard*-Grid-Ansicht nacheinander von oben nach unten in der Dokumentreihenfolge angezeigt (*siehe Abbildung links unten*). Durch Anzeige sich wiederholender Elemente als *Zeilen einer Tabelle* stehen Ihnen jedoch zusätzliche Bearbeitungsfunktionen zur Verfügung. In der Abbildung unten ist das Element **`Person`** das sich wiederholende Element. In der Abbildung links sehen Sie die Standard-Grid-Ansicht; das erste **`Person`**-Element wird erweitert angezeigt, während die darauf folgenden Instanzen reduziert angezeigt werden. In der Abbildung rechts werden die sich wiederholenden **`Person`**-Elemente als Zeilen einer Tabelle angezeigt.

Um in die Tabellenansicht zu wechseln, klicken Sie auf das **Tabellenmodus**-Symbol *(in der Abbildung links unten grün umrandet)*. Wenn Sie sich in der Tabellenansicht befinden, wird das Symbol färbig angezeigt *(siehe Abbildung rechts unten)*.

© 2016-2022 Altova GmbH



Die Tabellenansicht bietet beim Bearbeiten den einzigartigen Vorteil, dass ganze Zeilen und Spalten relativ zu anderen Spalten und Zeilen in der Tabelle bearbeitet werden können. Auf diese Art können Tabellenzeilen nach den Werten einer Spalte sortiert werden. So können etwa in der Abbildung rechts oben die sechs **`Person`**-Elemente anhand ihres Child-Elements **`Last`** mit einem einzige Befehl aus der Grid-Ansichts-Symbolleiste[296] sortiert werden. Dies ist einfacher, als ein XML-Nodeset auf die übliche Weise mit Hilfe einer XSLT-Transformation zu sortieren.

Nähere Informationen finden Sie im Abschnitt Tabellenansicht[312].

5.4.4 Drag-and-Drop in der Grid-Ansicht

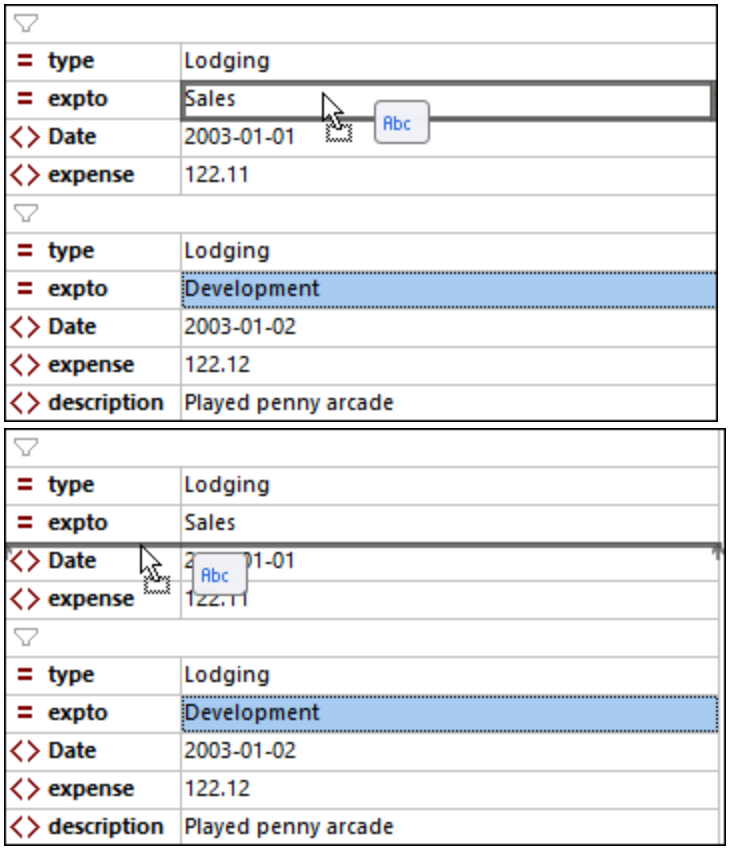
Die Grid-Ansicht bietet ein äußerst nützliches Drag Overlay, mit Hilfe dessen Sie ein XML-Dokumentfragment aus einem in XMLSpy, einer externen Applikation oder sogar einer Website geöffneten Dokument in die Grid-Ansicht ziehen können.

Wenn Sie den Cursor über den Ziel-Node platzieren, erscheint ein Drag Overlay, das nicht nur Informationen darüber enthält, was gerade mit der Maus gezogen wird (Typ und Anzahl der Elemente), sondern auch darüber, wie dieser Inhalt nach dem Loslassen der Maustaste erstellt wird. So wird etwa in der Abbildung unten der Wert **`Development`** mit der Maus gezogen (verschoben). Dass es sich dabei um einen String-Wert handelt, wird durch das Popup **`Abc`** angezeigt.

- Wenn Sie die Maus über ein Wertefeld platzieren (wie in der Abbildung links), wird das gesamte Feld markiert, wodurch angezeigt wird, dass der Wert des markierten Felds durch den Wert des dorthin gezogenen Felds ersetzt wird.

- Wenn Sie die Maus über einen Node-Namen platzieren (wie `Date` in der Abbildung rechts), erscheint eine Einfügezeile, wodurch angezeigt wird, dass der Wertestring hierhin gezogen wird, um in der Struktur einen Text-Node zu erstellen.

Anmerkung: Sie können jede Verschiebung immer mit **Strg+Z** oder über den Menübefehl **Bearbeiten | Rückgängig** rückgängig machen.



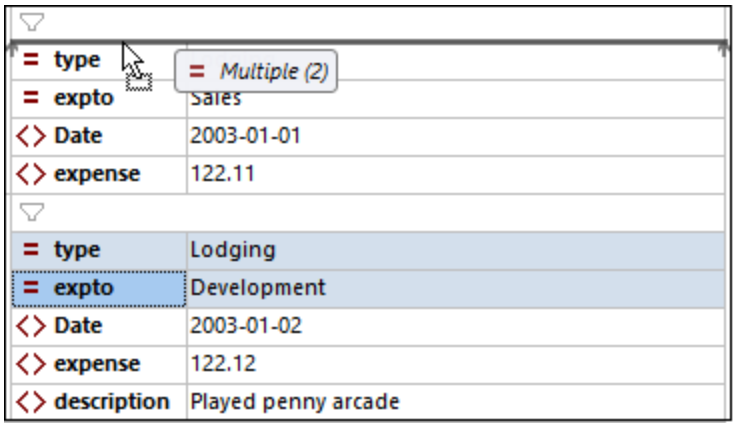
Im Drag Overlay angezeigte Informationen

Es werden die folgenden Arten von Drag Overlay-Informationen angezeigt.

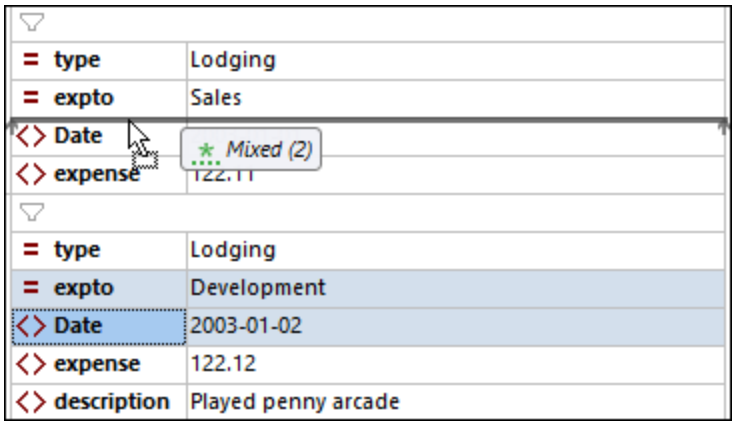
Normale Grid-Ansicht

- Wenn ein Wert in ein Wertefeld gezogen wird, wird der Wert im Zielfeld durch den Quellwert überschrieben (*siehe Abbildung links oben*).
- Wenn ein Wert als Node in die Struktur gezogen wird, wird ein Text-Node erstellt (*siehe Abbildung rechts oben*).
- Im Fall eines Overlay mehrerer Nodes eines einzigen Typs werden der Typ des gezogenen Nodes und die Anzahl dieser Nodes angezeigt. Beachten Sie, dass Sie auf den Namen des Node klicken müssen, um einen Node (und nicht seinen Wert) auszuwählen. In der Abbildung unten sehen Sie anhand der Informationen im Drag Overlay, dass zwei Attribute eingefügt werden.

© 2016-2022 Altova GmbH

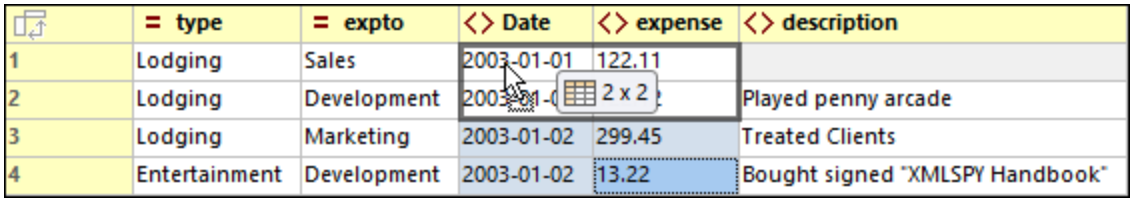


- Beim Overlay mehrerer Nodes unterschiedlichen Typs wird der Text *Mixed* und die Anzahl der Nodes in der Zwischenablage angezeigt (*siehe Abbildung unten*). Beachten Sie, dass Sie auf den Namen des Node klicken müssen, um einen Node (und nicht seinen Wert) auszuwählen.

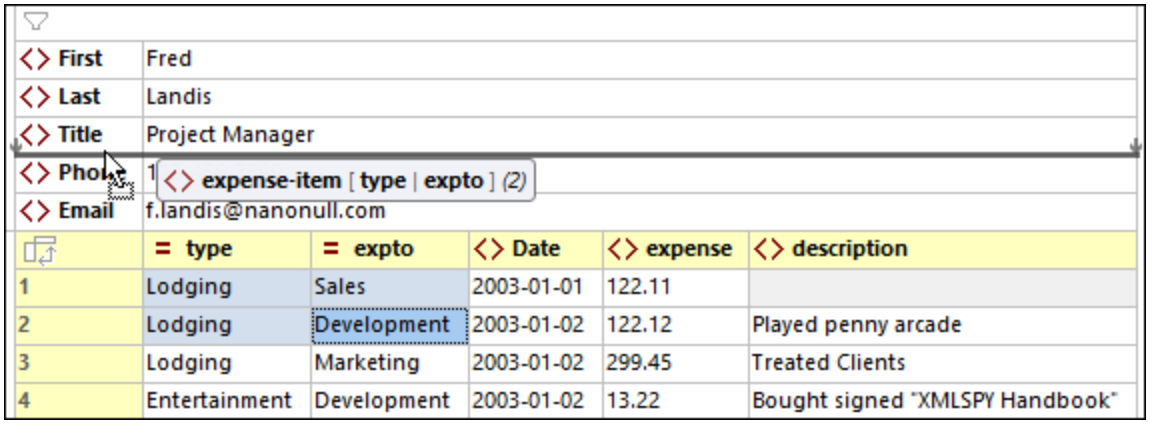


Tabellenansicht

- Wenn Werte mit der Maus gezogen werden, wird die Anzahl der ausgewählten Zellen durch **`Spalten x Zeilen`** (*siehe Abbildung unten*) angegeben. Die entsprechende Matrix wird durch die Matrix der mit der Maus gezogenen Zellen ersetzt. Die Zielzelle (d.h. die Zelle auf die Sie die Matrix ziehen) erhält den Wert der obersten linken Zelle der Matrix. Die Zellen rechts und unterhalb dieser Zielzelle werden durch die restlichen mit der Maus gezogenen Zellen ausgefüllt. So werden etwa in der Abbildung unten die blauen Zellen auf die erste**`Date`**-Zelle gezogen. Dadurch werden die markierten Zellen durch die neuen Zellen ersetzt. Wenn die Matrix der mit der Maus gezogenen Zellen die Tabellengröße übersteigt, werden entsprechend viele Spalten und/oder Zeilen hinzugefügt, um Platz für alle mit der Maus gezogenen Zellen zu schaffen. In diesem Fall werden die erweiterten Grenzen durch gestrichelte Linien angezeigt.



- Wenn das Ziel die Node-Struktur und nicht eine Zelle ist, wird der hinzugefügte Node zusammen mit (i) den Spalten, die hinzugefügt werden (innerhalb von eckigen Klammern) und (ii) der Anzahl der für das neue Element hinzugefügten Zeilen angezeigt (*siehe Abbildung unten*).



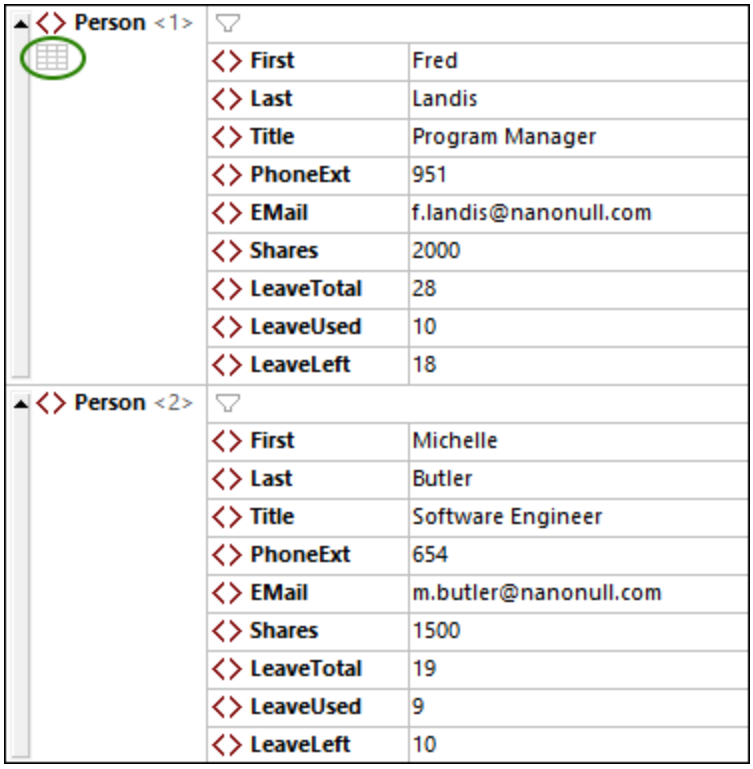
- Um eine Tabellenspalte zu verschieben, wählen Sie diese durch Klick auf ihre Kopfzeile aus und ziehen Sie sie auf die Spaltenüberschrift neben derjenigen, in die sie die Spalte verschieben möchten. Daraufhin wird mittels einer Einfügelinie angezeigt, auf welcher Seite der Zielspalte die verschobene Spalte eingefügt wird.

5.4.5 Tabellenansicht

Informationen zur Tabellenansicht

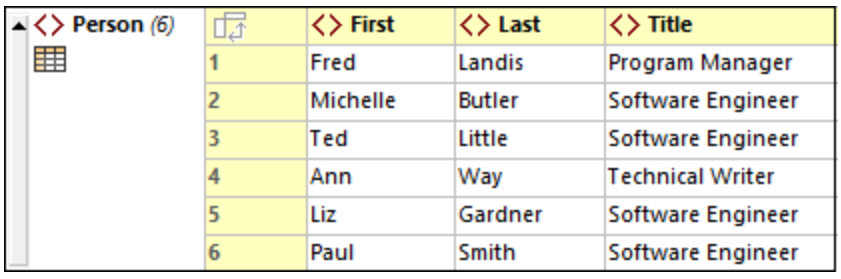
Wenn sich ein Element in der Standard-Grid-Ansicht wiederholt, werden die einzelnen Elementinstanzen nacheinander in Dokumentreihenfolge angezeigt. In der Abbildung unten ist z.B. das Element **`Person`** das sich wiederholende Element.

© 2016-2022 Altova GmbH



Person <1>	
First	Fred
Last	Landis
Title	Program Manager
PhoneExt	951
EMail	f.landis@nanonull.com
Shares	2000
LeaveTotal	28
LeaveUsed	10
LeaveLeft	18
Person <2>	
First	Michelle
Last	Butler
Title	Software Engineer
PhoneExt	654
EMail	m.butler@nanonull.com
Shares	1500
LeaveTotal	19
LeaveUsed	9
LeaveLeft	10

Ein sich wiederholendes Element wie das oben gezeigte Element **Person** kann auch in Form einer Tabelle angezeigt werden (*Abbildung unten*). In der Tabellendarstellung bilden die Child-Nodes des sich wiederholenden Elements die Spalten der Tabelle, während die Instanzen des sich wiederholenden Elements die Zeilen der Tabelle darstellen. Um in die Tabellenansicht zu wechseln, klicken Sie auf die Schaltfläche **Tabellenmodus**, die in der ersten Instanz des sich wiederholenden Elements angezeigt wird *(in der Abbildung oben grün umrandet)*. Wenn Sie sich in der Tabellenansicht befinden, wird das **Tabellenmodus**-Symbol färbig angezeigt *(siehe Abbildung unten)*.



Person *(6)*		First	Last	Title
	1	Fred	Landis	Program Manager
	2	Michelle	Butler	Software Engineer
	3	Ted	Little	Software Engineer
	4	Ann	Way	Technical Writer
	5	Liz	Gardner	Software Engineer
	6	Paul	Smith	Software Engineer

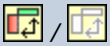
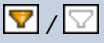
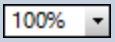
Die Tabellenansicht bietet beim Bearbeiten insofern einen Vorteil, als ganze Zeilen und Spalten relativ zu anderen Spalten und Zeilen in der Tabelle bearbeitet werden können. Auf diese Art können z.B. Zeilen nach den Werten einer Spalte sortiert werden. So können etwa in der Abbildung oben die sechs `Person`-Elemente anhand ihres Child-Elements **Last** mit einer einfachen GUI-Operation sortiert werden. Eine solche Operation *(Näheres dazu siehe unten)* lässt sich einfacher durchführen, als ein XML-Nodeset auf die übliche Weise mit Hilfe einer XSLT-Transformation zu sortieren.

Anmerkung

- Die Tabellenansicht kann nur auf eine Sequenz von Elementen desselben Namens angewendet werden.
- Die Tabellenansicht steht auch für ein einziges Element zur Verfügung. Wenn Sie auf das Element klicken, wird das **Tabellenmodus**-Symbol des Elements sichtbar.
- Die Farben der Tabellenansicht können im Dialogfeld "Optionen" definiert werden: Schriftarten und Farben | Grid-Ansicht | Zellenfarben[1626].

Symbole zur Anzeige und Bearbeitung in der Tabellenansicht

Über die in der Tabelle unten aufgelisteten Symbole stehen Anzeige- und Bearbeitungsfunktionen zur Verfügung. Sie stehen in Tabellenzellen und/oder der Symbolleiste der Grid-Ansicht zur Verfügung.

	Optimiert die Breite von Grid-Spalten entsprechend dem Zelleninhalt.
	Bei farbiger Anzeige ist die Tabellenansicht[704] aktiviert, andernfalls ist sie deaktiviert. Klicken Sie darauf, um die Anzeige zu wechseln.
	Ein Ein/Aus-Befehl in der linken oberen Zelle einer Tabelle. Zeigt Zeilen als Spalten an und umgekehrt.
	Wenn das Symbol farbig angezeigt wird, ist ein Filter[709] für die Tabelle, das Objekt oder Array aktiv. Andernfalls ist der Filter deaktiviert. Klicken Sie darauf, um den Filter zu deaktivieren/zu aktivieren. Um den Ausdruck zu bearbeiten, doppelklicken Sie darauf.
	Ein/Aus-Befehl für den Zeilenumbruch in allen Zellen. Wenn die Schaltfläche ausgewählt ist, ist der Zeilenumbruch aktiviert. Standardmäßig wird nur der Inhalt von Elementen umbrochen. Wenn auch die Namen von Elementen umbrochen werden sollen, wählen Sie diese Option aus der Dropdown-Liste des Symbols aus.
	Zoomfaktor der Grid-Ansicht.
	Ist aktiv, wenn eine Spaltenüberschrift in der Tabellenansicht ausgewählt ist. Mit diesen Schaltflächen werden die Zeilen der Tabelle nach dem Spalteninhalt in auf- bzw. absteigender Reihenfolge sortiert.

Bearbeiten in der Tabellenansicht

In der Tabellenansicht können die folgenden unten beschriebenen Bearbeitungsvorgänge durchgeführt werden.

Hinzufügen einer Tabellenzeile (neue Instanz des sich wiederholenden Elements der Tabelle)
Um eine neue Zeile, d.h. eine weitere Instanz des sich wiederholenden Elements der Tabelle, hinzuzufügen, gehen Sie folgendermaßen vor:

1. Klicken Sie mit der rechten Maustaste oberhalb oder unterhalb der gewünschten neuen Zeile auf die *Zelle mit der Zellennummer*.
2. Wählen Sie den Befehl **Einfügen nach (Strg+Eingabe)**, um eine Zeile unterhalb der ausgewählten Zeile oder **Einfügen vor (Strg+Umschalt+Eingabe)**, um eine Zeile oberhalb der ausgewählten Zeile hinzuzufügen. Diese Befehle stehen auch im Menü "XML"[1348] und in der Grid-Ansichts-Symbolleiste[296] zur Verfügung.

Die neue Zeile wird als Element-Node erstellt. Wenn Sie möchten, können Sie ihren Node-Typ ändern *(nähere Informationen siehe unten)*.

© 2016-2022 Altova GmbH

Hinzufügen eines gleichrangigen Elements oder eines Child-Elements zu einer Tabellenzelle
Wenn eine Tabellenzelle ein Child-Element einer Tabellenzeile darstellt, können Sie diesem Child-Element einen darauf folgenden gleichrangigen Node oder einen Child-Node geben. Klicken Sie mit der rechten Maustaste auf die Tabellenzelle und wählen Sie den Befehl **Anhängen (Strg+Alt+A)** bzw. **Child hinzufügen (Strg+Alt+Eingabe)**. Diese Befehle stehen auch im Menü "XML" 1348 und in der Grid-Ansichts-Symbolleiste 296 zur Verfügung. Die neue Zeile wird als Element-Node erstellt. Wenn Sie möchten, können Sie ihren Node-Typ ändern *(nähere Informationen siehe unten)*.

Verpacken einer Zelle in ein Element
Sie können rund um eine Tabellezelle ein Element erstellen. Das neue Element wird auf Ebene zwischen der der Zelle und der des Parent der Zelle erstellt. Klicken Sie dazu mit der rechten Maustaste auf die Tabellenzelle und wählen Sie den Befehl **In Element verpacken (Strg+Alt+W)**. Dieser Befehl steht auch im Menü "XML" 1348 und in der Grid-Ansichts-Symbolleiste 296 zur Verfügung.

Hinzufügen einer Tabellenspalte (neuer Child-Node aller Instanzen des sich wiederholenden Elements der Tabelle)
Um eine neue Spalte, d.h. einen neuen Child-Node aller Instanzen des sich wiederholenden Elements der Tabelle hinzuzufügen, gehen Sie folgendermaßen vor:

1. Klicken Sie mit der rechten Maustaste auf eine Spaltenüberschrift oder eine nicht leere Zelle einer Spalte.
2. Wählen Sie den Befehl **Einfügen nach (Strg+Eingabe)**, um eine Spalte rechts von der ausgewählten Spalte oder **Einfügen vor (Strg+Umschalt+Eingabe)**, um eine Spalte links von der ausgewählten Spalte hinzuzufügen. Diese Befehle stehen auch im Menü "XML" 1348 und in der Grid-Ansichts-Symbolleiste 296 zur Verfügung.

Die neue Spalte wird als Element-Node erstellt. Wenn Sie möchten, können Sie ihren Node-Typ ändern *(nähere Informationen siehe unten)*.

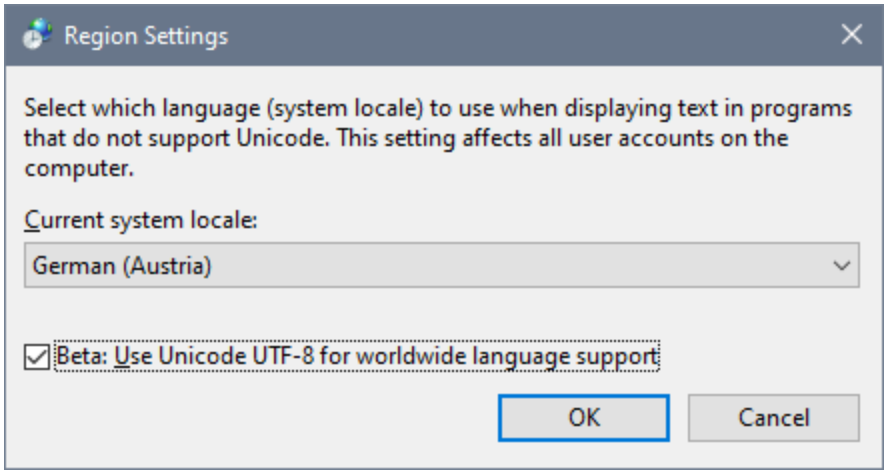
Ändern von Node-Typen und Spaltennamen
Um den Node-Typ einer Spalte zu ändern, klicken Sie auf das Node-Typ-Symbol der Spalte und wählen Sie den neuen Node-Typ im daraufhin angezeigten Menü aus. Der Typ dieses Node wird für alle Instanzen des sich wiederholenden Elements geändert.

Um den Namen einer Spaltenüberschrift zu ändern, doppelklicken Sie auf den Namen und bearbeiten Sie ihn. Daraufhin wird der Name dieses Node in allen Instanzen des sich wiederholenden Elements geändert.

Sortieren von Tabellenzeilen nach den Werten einer ausgewählten Spalte
Sie können die Zeilen einer Tabelle nach den relativen Werten eines ihrer Child-Nodes (als Spalte) sortieren. So können Sie etwa Zeilen auf Basis der Spalte *LastName* sortieren, damit die sich wiederholenden Elemente der Tabelle alphabetisch sortiert werden. Um eine Spalte zu sortieren, wählen Sie die Spaltenüberschrift aus und klicken Sie anschließend auf die Schaltfläche **Aufsteigend sortieren** oder **Absteigend sortieren** der Grid-Ansichts-Symbolleiste 296. Diese Befehle stehen auch im Menü "XML" 1348 zur Verfügung.

Für die Sortierreihenfolge ist es in manchen Sprachen, v.a. für solche mit einem nicht lateinischen Alphabet, von Vorteil, im Dialogfeld "Sprache Region" von Versionen ab Windows 10 die beta Unicode UTF-8-Unterstützung zu aktivieren. Gehen Sie dazu folgendermaßen vor: Gehen Sie zu Ihrem Windows-

Einstellungsdialogfeld und suchen Sie nach *Spracheinstellungen*. Klicken Sie unter *Verwandte Einstellungen* auf *Verwaltung Spracheinstellungen*. Gehen Sie im daraufhin angezeigten Dialogfeld "Region" zum Register *Verwaltung* und klicken Sie unter *Sprache für Unicode-inkompatible Programme* auf *Gebietsschema ändern*. Aktivieren Sie im daraufhin angezeigten Dialogfeld "Region" (*Abbildung unten*) die Option*Beta: Unicode UTF-8 für weltweite Sprachunterstützung* und klicken Sie auf **OK**.



Tabellenansicht und externe Applikationen

Mit Hilfe der Tabellenstruktur können Sie Daten zwischen der Tabellenansicht und einem Tabellenkalkulationsprogramm wie z.B. MS Excel austauschen. Um Daten aus der Tabellenansicht zu verschieben, gehen Sie folgendermaßen vor:

1. Wählen Sie in die zu kopierenen Nodes in der Tabellenansicht aus. Klicken Sie dazu auf die Zellen selbst, auf die Spalten- oder Zeilenüberschriften oder die gesamte Tabelle. Wenn Sie die gesamte Tabelle oder Spaltenüberschriften auswählen, wird auch der Text der Spaltenüberschriften kopiert; ansonsten nicht. (In der Abbildung unten sind die Zeilen 2 und 3 ausgewählt).

expense-item (3)	type	expto	Date	expense	description
1	Lodging	Sales	2003-01-01	122.11	
2	Lodging	Development	2003-01-02	122.12	Played penny arcade
3	Lodging	Marketing	2003-01-02	299.45	Treated Clients

2. Wählen Sie den Kontextmenübefehl Kopieren | Als tabulatorgetrennten Text kopieren[1300] aus.
3. Fügen Sie die Daten direkt in das Tabellenkalkulationsprogramm ein.

	A	B	C	D	E
1	Lodging	Development	1/2/2003	122.12	Played penny arcade
2	Lodging	Marketing	1/2/2003	299.45	Treated Clients

Der Datenaustausch funktioniert in beide Richtungen. Sie können auch Daten aus einem Arbeitsblatt in die Tabellenansicht kopieren. Gehen Sie dazu folgendermaßen vor:

1. Wählen Sie in der externen Applikation einen Bereich aus und kopieren Sie ihn (mit **Strg+C**) in die Zwischenablage.

© 2016-2022 Altova GmbH

2. Wählen Sie in der Tabellenansicht eine einzelne Zelle aus.
3. Fügen Sie die kopierten Daten mit **Strg+V** ein.

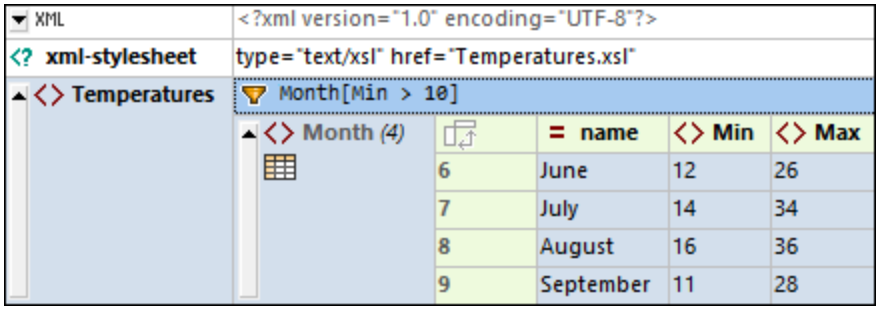
Die Daten werden mit der gleichen Struktur wie in der Originalstruktur ab der in der Tabellenansicht ausgewählten Zelle in die Tabelle kopiert. Beachten Sie bitte Folgendes:

- Wenn diese Zellen in der Tabellenansicht bereits Daten enthalten, werden die alten Daten durch die neuen überschrieben.
- Wenn für die neuen Daten mehr Zeilen und/oder Spalten benötigt werden, werden diese erstellt.
- Die Daten werden zum Inhalt der Elemente, für die die entsprechenden Zellen stehen.

Verwenden Sie für komplexere Datenaustauschvorgänge die einzigartigen Import/Export[1470]-Funktionen von XMLSpy.

5.4.6 Filter

Ein Filter in der XML-Grid-Ansicht kann auf einen Element-Node angewendet werden, um die Nachfahren dieses Node zu filtern. Überall dort, wo ein Filter angewendet werden kann, wird ein ausgegrautes Filtersymbol angezeigt. Sobald ein Filter definiert wurde, wird dieses Symbol farbig angezeigt *(siehe Abbildung unten)*. Ein Filter wird mit Hilfe eines XQuery 3.1-Ausdrucks definiert. So wurde z.B. in der Abbildung unten für den Node **`Temperatures`** ein Filter definiert, um nur diejenigen **`Month`**-Elemente anzuzeigen, die ein **`Min`**-Child-Element haben, das einen Wert, der größer als `10` ist, anzuzeigen. (Beachten Sie, dass in der Abbildung die Indexzahl der entsprechenden Elementinstanz entspricht).



Um einen Filter einzurichten, klicken Sie mit der rechten Maustaste auf das Element, das Sie filtern möchten, wählen Sie im angezeigten Kontextmenü den **Filter** aus, geben Sie den XQuery-Ausdruck ein und drücken Sie die **Eingabetaste**. Die *Filter*-Zelle ist durch das Symbol  *(siehe Abbildung oben)* gekennzeichnet. Durch Klick auf das *Filter*-Symbol können Sie den Filter ein- und ausschalten.

Codefragment des XML-Dokuments aus der Abbildung oben

```
<?xml version="1.0" encoding="UTF-8"?>
<Temperatures>
   <Month name="January">
      <Min>-5</Min>
      <Max>3</Max>
   </Month>
   <Month name="February">
      <Min>-16</Min>
```

```
      <Max>1</Max>
   </Month>
   <Month name="March">
      <Min>-9</Min>
      <Max>7</Max>
   </Month>
   <Month name="April">
      <Min>2</Min>
      <Max>16</Max>
   </Month>
   <Month name="May">
      <Min>8</Min>
      <Max>21</Max>
   </Month>
   <Month name="June">
      <Min>12</Min>
      <Max>26</Max>
   </Month>
   <Month name="July">
      <Min>14</Min>
      <Max>34</Max>
   </Month>
   <Month name="August">
      <Min>16</Min>
      <Max>36</Max>
   </Month>
   <Month name="September">
      <Min>11</Min>
      <Max>28</Max>
   </Month>
   <Month name="October">
      <Min>10</Min>
      <Max>26</Max>
   </Month>
   <Month name="November">
      <Min>-1</Min>
      <Max>14</Max>
   </Month>
   <Month name="December">
      <Min>-3</Min>
      <Max>9</Max>
   </Month>
</Temperatures>
```

Beachten Sie zu Filtern die folgenden Punkte:

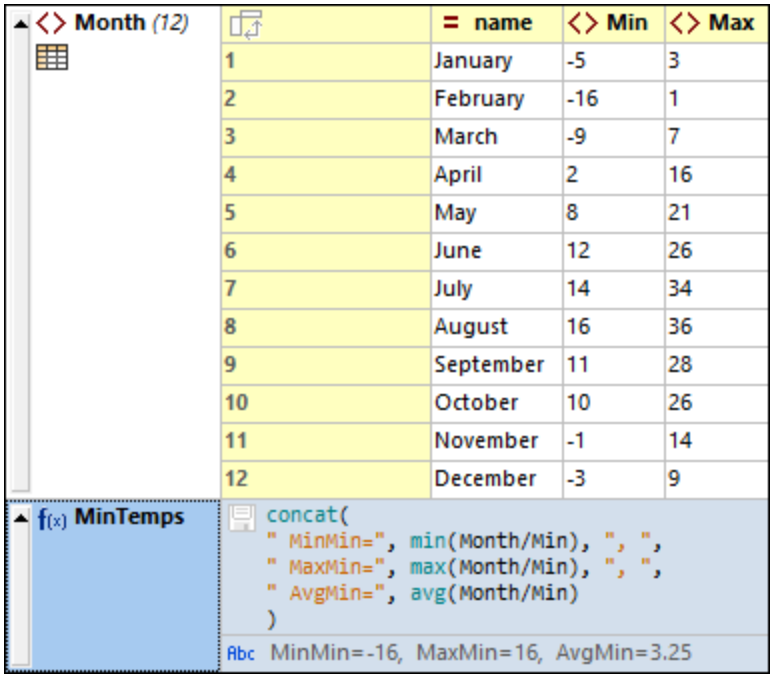
- Filter können nur auf Element-Nodes angewendet werden.
- Der Kontext-Node des XQuery-Ausdrucks des Filters ist der aktuelle Node. So ist z.B. in der Abbildung oben der Kontext-Node der Node **Temperatures**.
- Filter können verschachtelt sein. Verschachtelte Filter werden auf den gefilterten Inhalt des übergeordneten Filters angewendet.

© 2016-2022 Altova GmbH

- Um in einem Ausdruck eine neue Zeile hinzuzufügen, drücken Sie **Strg+Eingabetaste**. Dies ist nützlich, wenn Sie einen Ausdruck mehrzeilig anzeigen möchten, um eine bessere Übersicht zu haben.
- Der gefilterte Inhalt ist nur eine visuelle Anzeige. Der eigentliche Inhalt bleibt dabei unverändert.
- Jeder Filter wird separat ausgeführt. Andere Filter oder Formeln[319] haben keine Auswirkung darauf.
- Filter werden **nicht** im XML-Dokument gespeichert, können aber für die spätere Verwendung in einer speziellen Metadatendatei aufbewahrt werden. Wenn Filter gespeichert werden, werden sie automatisch angewendet, wenn das Dokument wieder in der-Grid-Ansicht geöffnet wird. Gehen Sie zu **Extras | Optionen | Ansicht | Einstellungen für Grid-Ansicht**[1617], um festzulegen, ob Filter immer automatisch, auf Anfrage oder nicht gespeichert werden sollen.

5.4.7 Formeln

A Mit Hilfe von Formeln kann in der XML-Grid-Ansicht mit Hilfe eines XQuery 3.1-Ausdrucks ein Ergebnis berechnet oder ein Nodeset generiert werden, das im Dokument gespeichert werden kann. Eine Formel wird mit Hilfe eines XQuery 3.1-Ausdrucks definiert. So wurde etwa in der Abbildung unten eine Formel namens **`MinTemps`** erstellt, die für alle Temperaturminima die Minimal-, Maximal- und Durchschnittstemperatur generiert.



Um eine Formel zu erstellen, gehen Sie folgendermaßen vor:

1. Fügen Sie an der Stelle, an der die Formel angezeigt werden soll, einen neuen Node hinzu.
2. Der Node wird standardmäßig als Element erstellt. Ändern Sie den Typ des Node[305] in "Formel".
3. Doppelklicken Sie in die Zelle, die das Symbol **`f(x)`** enthält, und geben Sie den Namen der Formel ein *(siehe Abbildung oben)*.
4. Sie können auf dieses Symbol klicken, um die Ausgabe der Formel im Dokument zu speichern.
5. Doppelklicken Sie in die Zelle des Ausdrucks, geben Sie den gewünschten Ausdruck ein und drücken Sie die **Eingabetaste**.

Codefragment des XML-Dokuments aus der Abbildung oben

```
<?xml version="1.0" encoding="UTF-8"?>
<Temperatures>
   <Month name="January">
      <Min>-5</Min>
      <Max>3</Max>
   </Month>
   <Month name="February">
      <Min>-16</Min>
      <Max>1</Max>
   </Month>
   <Month name="March">
      <Min>-9</Min>
      <Max>7</Max>
   </Month>
   <Month name="April">
      <Min>2</Min>
      <Max>16</Max>
   </Month>
   <Month name="May">
      <Min>8</Min>
      <Max>21</Max>
   </Month>
   <Month name="June">
      <Min>12</Min>
      <Max>26</Max>
   </Month>
   <Month name="July">
      <Min>14</Min>
      <Max>34</Max>
   </Month>
   <Month name="August">
      <Min>16</Min>
      <Max>36</Max>
   </Month>
   <Month name="September">
      <Min>11</Min>
      <Max>28</Max>
   </Month>
   <Month name="October">
      <Min>10</Min>
      <Max>26</Max>
   </Month>
   <Month name="November">
      <Min>-1</Min>
      <Max>14</Max>
   </Month>
   <Month name="December">
      <Min>-3</Min>
      <Max>9</Max>
   </Month>
```


© 2016-2022 Altova GmbH

```
</Temperatures>
```

Codefragment des in der Abbildung oben verwendeten Formelausdrucks

```
concat
" MinMin=", min(//Month/Min), ", ",
" MaxMin=", max(//Month/Min), ", ",
" AvgMin=", avg(//Month/Min)
)
```

Beachten Sie die folgenden Punkte:

- Der Kontext-Node des XQuery-Ausdrucks der Formel ist der Parent-Node des Formel-Node.
- Um in einem Ausdruck eine neue Zeile hinzuzufügen, drücken Sie **Strg+Eingabetaste**. Dies ist nützlich, wenn Sie einen Ausdruck mehrzeilig anzeigen möchten, um eine bessere Übersicht zu haben.
- Formeln (d.h. die Ausdrücke, mit denen Formeln definiert werden) werden in den Metadaten des Dokuments (in einer applikationsweiten Metadatendatei) gespeichert. Formeln werden in der XML-Datei auch als Processing Instructions gespeichert.
- Das von einer Formel generierte Ergebnis wird in der Zelle unterhalb des XQuery-Ausdrucks der Formel angezeigt und in den Metadaten des Dokuments gespeichert. Das Ergebnis kann auch im Dokument gespeichert werden. Klicken Sie dazu auf das Diskettensymbol neben dem XQuery-Ausdruck. Unabhängig davon, ob Sie auf das Diskettensymbol klicken oder nicht, wird das Ergebnis der Formel berechnet und in den Metadaten das Dokuments gespeichert.
- Wenn es sich beim Ergebnis der Formel um eine Berechnung handelt, wird diese als Inhalt eines Elements mit dem Namen, den Sie der Formel zugewiesen haben, gespeichert. So wird etwa das Ergebnis der **`MinTemps`**-Formel aus der Abbildung oben in einem Element namens **`MinTemps`** gespeichert.
- Gehen Sie zu **Extras | Optionen | Ansicht | Einstellungen für Grid-Ansicht**[1617], um (i) anzugeben, ob Formeln immer, nur auf Anfrage oder nie in den Metadaten gespeichert werden sollen und um anzugeben, (ii) ob Formeln im Dokument gespeichert werden sollen.

Formeln in Tabellen

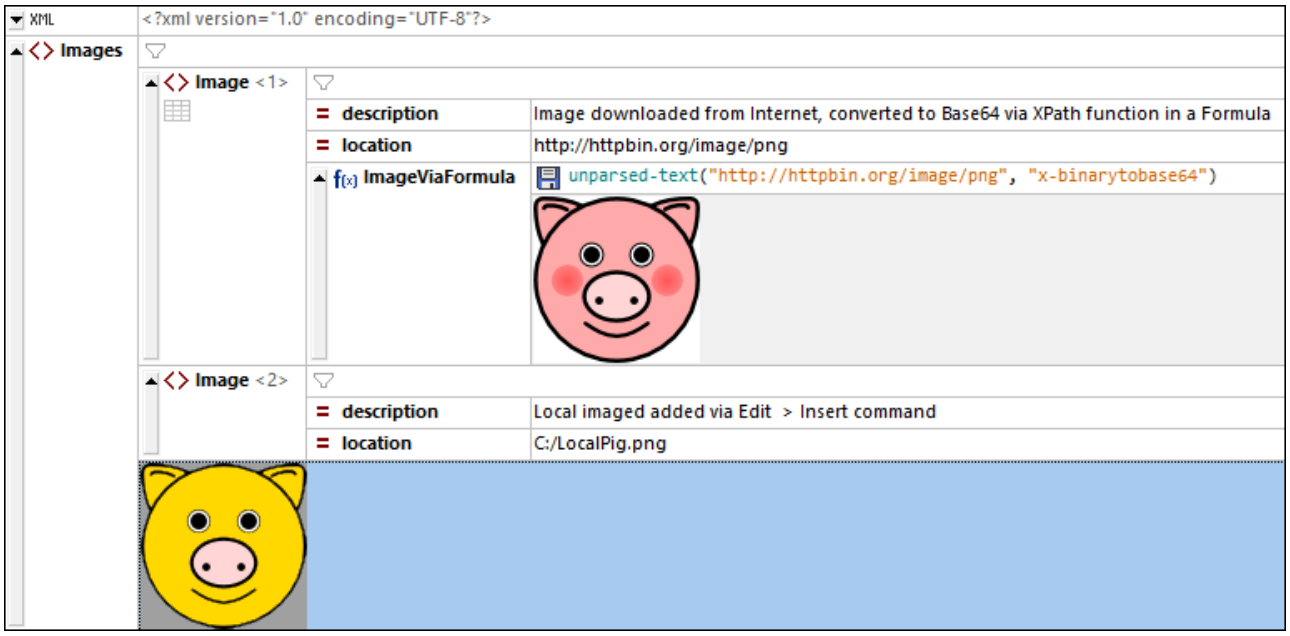
Wenn alle Zellen einer Tabellenspalte (in der Tabellenansicht[704]) dieselbe Formel enthalten, so wird die Formel nur einmal angezeigt, und zwar in der Spaltenüberschrift (*siehe Abbildung unten*). Die *Berechnungsergebnisse* der Formel werden jedoch in den jeweiligen Zellen angezeigt. Die Formel in der Spaltenüberschrift ist eine Darstellung in der Grid-Ansicht. Im XML-Dokument (siehe Textansicht) wird die Formel für jedes Tabellenzeilenelement wiederholt.

Month (12)	name	Min	Max	MidRange (Min + Max) div 2
1	January	-5	3	# -1
2	February	-16	1	# -7.5
3	March	-9	7	# -1
4	April	2	16	# 9
5	May	8	21	# 14.5
6	June	12	26	# 19
7	July	14	34	# 24
8	August	16	36	# 26
9	September	11	28	# 19.5
10	October	10	26	# 18
11	November	-1	14	# 6.5
12	December	-3	9	# 3

Wenn sich auch nur eine einzige Formel einer Zelle von den anderen unterscheidet, so wird jede Formel in ihrer dazugehörigen Zelle angezeigt. Wenn alle Formeln einer Tabellenspalte identisch sind, sodass die Formel in der Überschrift angezeigt wird und Sie für eine einzelne Zelle eine andere Formel erstellen möchten, deaktivieren Sie die Tabellenansicht und bearbeiten Sie die Formel dieser Zelle. Wenn Sie in die Tabellenansicht zurückwechseln, werden die Formeln (für alle Zellen) in den einzelnen Zellen angezeigt.

5.4.8 Bilder

Bilder können direkt in der Grid-Ansicht angezeigt werden (*siehe Abbildung unten*). Dazu muss das Bild in seiner Base64-Kodierung eingefügt werden.



© 2016-2022 Altova GmbH

Es gibt zwei Methoden, um ein Base64-kodiertes Bild in eine Zelle der Grid-Ansicht einzufügen:

- Erstellen Sie einen Node vom Typ *Formel*. Geben Sie in die Formelzelle *(siehe Abbildung oben)* den folgenden XPath-Ausdruck ein, um ein Bild in seine Base64-Kodierung zu konvertieren: `unparsed-text("<Image-URL>", "x-binarytobase64")`. Die XPath-Funktion `unparsed-text` konvertiert das Bild in die Base64-Kodierung. Das Bild wird unterhalb der Zelle, die den Formelausdruck enthält, dargestellt.
- Setzen Sie den Cursor in die Zelle, in der das Bild hinzugefügt werden soll. So wird etwa in der Abbildung oben das Bild als Inhalt des zweiten `Image`-Elements hinzugefügt, indem Sie das `Image`-Element auswählen. Wählen Sie anschließend den Befehl **Bearbeiten | Einfügen | Externe kodierte Datei**. Geben Sie im daraufhin angezeigten Dialogfeld (i) den Pfad zum gewünschten Bild (lokal oder Internet) ein, (ii) wählen Sie *Base 64*, (ii) wählen Sie *Text erstellen*. Daraufhin wird das Bild in seine Base64-Kodierung konvertiert und der kodierte Text wird in die Zelle eingegeben. In der Textansicht sehen Sie den kodierten Text. In der Grid-Ansicht hingegen wird nicht die Kodierung, sondern eine Darstellung des Bilds in der Zelle angezeigt *(siehe Abbildung oben)*.

Bild-URLs können einschließlich relativer Pfade in der folgenden Form angegeben werden:

- `http://httpbin.org/image/png`
- `file:///c:/LocalPig.png`
- `C:/LocalPig.png`
- `C:/LocalPig.png`

Die meisten gebräuchlichen Bildformate werden unterstützt. Dazu gehören PNG, JPEG, BMP und animierte GIFs. SVG wird als XML-Dokument gelesen; in der Grid-Ansicht wird das Bild als das letzte Child-Element des `<svg>`-Elements gelesen.

Speichern eines Base64-kodierten String als Bild

Um einen Base64-kodierten String in seinem Bildformat zu speichern, klicken Sie mit der rechten Maustaste auf das Bild oder dessen Zelle und wählen Sie den Befehl **Als Bild speichern**. (Beachten Sie, dass das Bild in der aktiven Datei als Base64-String kodiert ist, in der Grid-Ansicht aber als Bild angezeigt wird). Wählen Sie im daraufhin angezeigten Dialogfeld den Pfad, unter dem Sie das Bild speichern möchten und geben Sie einen Namen für die Bilddatei ein. Die Erweiterung der Bilddatei (`.png`, `.gif`, `.svg`, usw.) wird automatisch anhand der Base64-Kodierung ermittelt und im Speichern-Dialogfeld angezeigt. Klicken Sie anschließend auf **Speichern**.

Diese Aktion kann auch über den Menübefehl **Bearbeiten | Als Bild speichern** durchgeführt werden.

5.4.9 Diagramme

In der XML-Grid-Ansicht können mit Hilfe der Altova XPath/XQuery-Erweiterung `altovaext:chart` Diagramme erstellt werden (*siehe Abbildung unten*). Diese Erweiterung wird weiter unten beschrieben. Eine Beschreibung dazu finden Sie zusammen mit anderen Diagrammerweiterungen im Abschnitt Diagrammfunktionen 2223. Die in der Abbildung unten gezeigte `altovaext:chart`-Erweiterungsfunktion wird in einem XQuery-`Let`-Ausdruck, der innerhalb einer Formel in der Grid-Ansicht 319 definiert wird, verwendet. Das Diagramm wird unterhalb der Formel als Bild angezeigt. (Sie können die Diagrammfunktion unter Verwendung des XML-Codefragments und des XQuery-Ausdrucks weiter unten ausprobieren.)

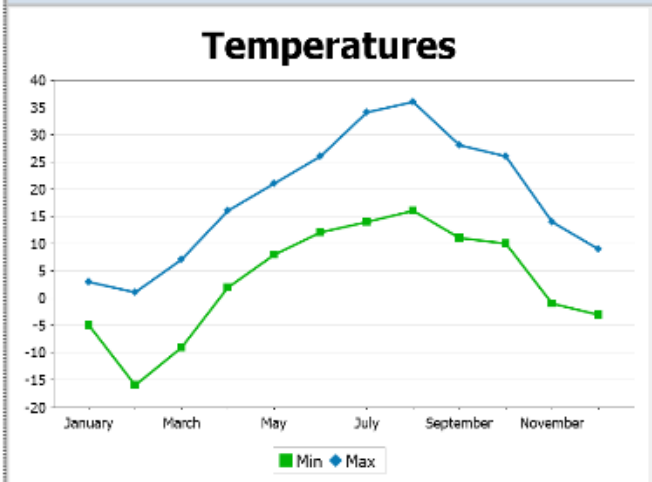
XML <?xml version="1.0" encoding="UTF-8"?>
xml-stylesheet type="text/xsl" href="Temperatures.xsl"
Temperatures

Month (12)

	name	Min	Max
1	January	-5	3
2	February	-16	1
3	March	-9	7
4	April	2	16
5	May	8	21
6	June	12	26
7	July	14	34
8	August	16	36
9	September	11	28
10	October	10	26
11	November	-1	14
12	December	-3	9

f(x) new_elem

```
let $months := //Month return
altovaext:chart(map{ "title": "Temperatures", "kind": "LineChart" },
(
    (:name, X-axis,        Y-axis :)
    ['Min', $months/@name, $months/Min],
    ['Max', $months/@name, $months/Max]
))
```



Codefragment des XML-Dokuments aus der Abbildung oben

```
<?xml version="1.0" encoding="UTF-8"?>
<Temperatures>
   <Month name="January">
      <Min>-5</Min>
      <Max>3</Max>
   </Month>
   <Month name="February">
      <Min>-16</Min>
```


© 2016-2022 Altova GmbH

```
      <Max>1</Max>
   </Month>
   <Month name="March">
      <Min>-9</Min>
      <Max>7</Max>
   </Month>
   <Month name="April">
      <Min>2</Min>
      <Max>16</Max>
   </Month>
   <Month name="May">
      <Min>8</Min>
      <Max>21</Max>
   </Month>
   <Month name="June">
      <Min>12</Min>
      <Max>26</Max>
   </Month>
   <Month name="July">
      <Min>14</Min>
      <Max>34</Max>
   </Month>
   <Month name="August">
      <Min>16</Min>
      <Max>36</Max>
   </Month>
   <Month name="September">
      <Min>11</Min>
      <Max>28</Max>
   </Month>
   <Month name="October">
      <Min>10</Min>
      <Max>26</Max>
   </Month>
   <Month name="November">
      <Min>-1</Min>
      <Max>14</Max>
   </Month>
   <Month name="December">
      <Min>-3</Min>
      <Max>9</Max>
   </Month>
</Temperatures>
```

Codefragment eines XQuery-Ausdrucks zur Generierung des Diagramms in der Abbildung oben

```
let $months := //Month return
altovaext:chart(map{ "title":"Temperatures", "kind":"LineChart" },
(
    (:name, X-axis,       Y-axis   :)
    ['Min', $months/@name, $months/Min],
```

```
        ['Max', $months/@name, $months/Max]
    ))
```

Verwendung der Altova Diagramm-Erweiterung

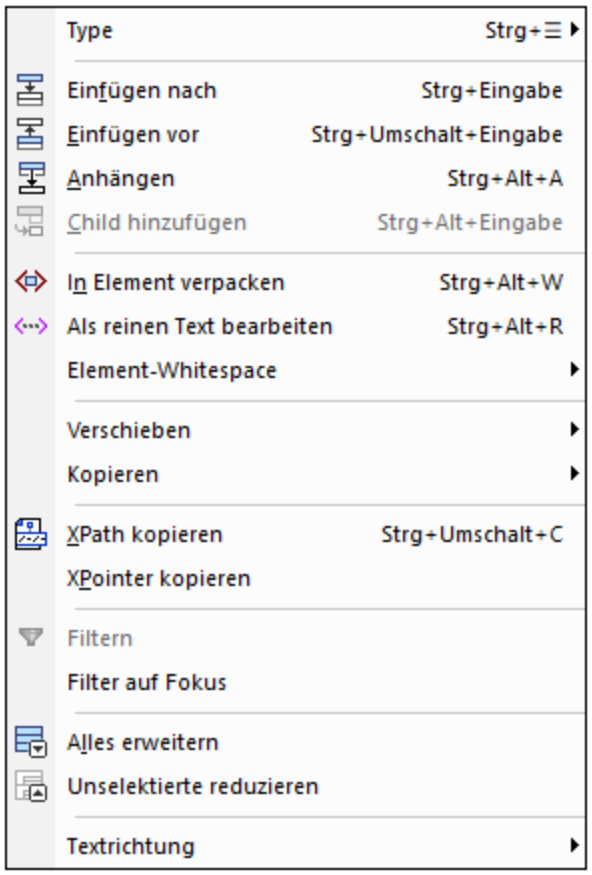
- In der Diagramm-Erweiterungsfunktion `altovaext:chart` muss das Namespace-Präfix `altovaext:` verwendet werden.
- Die Funktion `altovaext:chart` erhält zwei Argumente: (i) Diagrammkonfigurationsinformationen und (ii) Informationen über die Diagrammdatenreihe.
- Die Diagrammkonfigurationsinformationen bilden das erste Argument von `altovaext:chart`. Es handelt sich hierbei um eine nicht geordnete Reihe von vier Schlüssel-Wert-Paaren. Diese Paare sind für (i) den Titel des Diagramms (der Schlüssel ist `title`), (ii) die Art des Diagramms, wie z.B. Kreisdiagramm, Liniendiagramm, usw. (der Schlüssel ist `kind`; verfügbare Arten finden Sie unter Diagrammfunktionen 2223), (iii) die Diagrammbreite in Pixel (nur Ganzzahlen; der Schlüssel ist `width`) und (iv) die Diagrammhöhe in Pixel (nur Ganzzahlen; der Schlüssel ist `height`). Wenn entweder der Wert "width" oder der Wert "height" oder beide Werte fehlen, werden der fehlende Wert bzw. die fehlenden Werte anhand der Daten automatisch berechnet.
- Die Diagrammdatenreihe bildet das zweite Argument von `altovaext:chart`. Jede Datenreihe ist ein Array der Größe 3: (i) der Name der Datenreihe, (ii) die Werte der x-Achse, (iii) die Werte der y-Achse. Wenn Sie mehrere Datenreihen erstellen möchten (z.B. solche, bei denen jede Datenreihe eine Linie bildet, wie im Beispiel oben), erstellen Sie eine Sequenz mehrerer Arrays. Das Beispiel oben enthält drei Datenreihen; für die Temperaturminima, -maxima und die Durchschnittstemperaturen. Die Daten für die x- und y-Achse werden über die Sequenz aller `Month`-Elemente referenziert.

Nähere Informationen dazu finden Sie im Abschnitt Diagrammfunktionen 2223.

5.4.10 Kontextmenü in der Grid-Ansicht

Wenn Sie in der-Grid-Ansicht mit der rechten Maustaste auf eine Zelle klicken, wird ein Kontextmenü (*Abbildung unten*) mit Befehlen zum Bearbeiten von Inhalt im Zusammenhang mit der Zelle und zum Ändern der Anzeige angezeigt. Das Kontextmenü kann auch durch Drücken der **Menütaste** der Tastatur aufgerufen werden. Eine Beschreibung dieser Befehle finden Sie weiter unten.

© 2016-2022 Altova GmbH

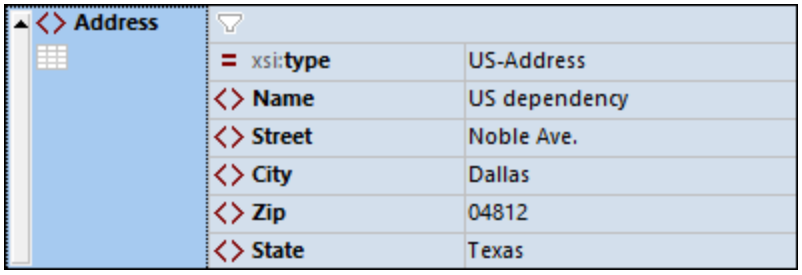


Typ
Wenn Sie den Mauszeiger über den Befehl **Typ** platzieren, wird ein Untermenü angezeigt, in dem Sie den Typ der Komponente 305 auswählen können. Sie können das **Typ**-Untermenü auch direkt über das Tastenkürzel **Strg+Menü** aufrufen.

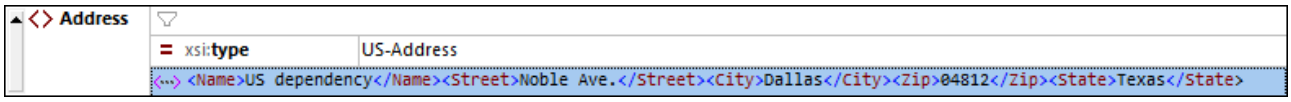
Einfügen nach/vor, Anhängen, Child hinzufügen
Mit den Befehlen **Einfügen** und **Anhängen** fügen Sie einen Eintrag (standardmäßig ein Element) auf derselben Ebene hinzu. Das neue Element wird im Fall von **Einfügen vor** und **Einfüge nach** vor bzw. nach dem ausgewählten Element und im Fall von **Anhängen** als letztes gleichrangiges Element des ausgewählten Elements hinzugefügt. Mit dem Befehl **Child hinzufügen** wird ein neuer Eintrag (standardmäßig ein Element) als Child-Element angehängt. Durch Doppelklick auf die Namenszelle und Bearbeitung dieser können Sie den Namen des neu hinzugefügten Elements ändern. Durch Klick auf das Symbol des Elements (links von seinem Namen) und Auswahl des gewünschten Node-Typs können Sie den Node-Typ ändern.

In Element verpacken
Der ausgewählte Eintrag erhält ein Parent-Element mit einem Standardelement, das Sie (durch Doppelklick auf den Namen und Bearbeitung des Namens) umbenennen können.

Als reinen Text bearbeiten
Damit können Sie Textinhalt des ausgewählten Elements als reinen Text bearbeiten. Dies ist z.B. dann nützlich, wenn Sie komplexen Inhalt wie z.B. HTML-Code bearbeiten. So sehen Sie z.B. in der Abbildung unten das Element `Address` in der Grid-Ansicht.



Wenn Sie für die Anzeige des Elements **Address** auf reinen Text (*Abbildung unten*) wechseln, wird die Rasterstruktur in ein einziges Element, das reinen Text enthält, der zur Gänze in der Textansicht bearbeitet werden kann, konvertiert.



Element-Whitespace
Steht für Element-Nodes zur Verfügung. Der Befehl fügt das **xml:space**-Attribut zum Markup-Code des Elements hinzu und gibt diesem Attribut den Wert, den Sie aus dem Untermenü des Befehls auswählen.

- Wenn Sie jedoch den Wert *Weglassen* auswählen, wird das **xml:space**-Attribut nicht hinzugefügt, sodass Whitespaces normalisiert werden.
- Mit *Beibehalten* werden alle Whitespaces unverändert beibehalten und die Pretty-Print[145]-Anzeige für dieses Element wird deaktiviert. Wenn Whitespace beibehalten wurden, wird in der Grid-Ansicht ein Auslassungssymbol angezeigt.
- Bei Auswahl des Werts *Standard* wird der für dieses Element im Schema definierte Wert verwendet. Damit können Sie einen aus einem Vorfahren-Element im XML-Dokument geerbten **xml:space**-Wert außer Kraft setzen.

Nach oben/nach unten/höher stufen/tiefer stufen
Eine Komponente kann im Raster nach oben, unten, links oder rechts verschoben werden. In diesem Fall sind die entsprechende Befehle aktiv. Wählen Sie den entsprechenden Befehl aus, um die Komponente zu verschieben.

Kopieren
Mit Hilfe dieser Befehle kann die aktuelle Auswahl kopiert werden. Es sind nur diejenige Optionen aktiv, die für diese Komponente zulässig sind.

Befehl	Beschreibung
Als XML-Text kopieren	Die aktuelle Auswahl wird als XML-Markup-Code serialisiert.
Als tabulatorgetrennten Text kopieren	Die aktuelle Tabellenauswahl wird als TSV (Tab Separated Values = tabulatorgetrennte Werte) serialisiert.
Als Bild kopieren	Die aktuelle Bildzelle wird als Bild kopiert

Beachten Sie die folgenden Punkte:

© 2016-2022 Altova GmbH

- *Als Bild kopieren* kopiert den als Base64-kodierten String des ausgewählten Bilds. Wenn der String in ein Dokument eingefügt wird, in dem der Base-64-kodierte String als Bild dargestellt werden kann (z.B. in eine andere Tabellenzelle in der Tabellenansicht), so wird er als Bild dargestellt. Anderfalls wird er als String eingefügt.
- Um Text in eine Zelle einzufügen, kopieren Sie den Text und fügen Sie ihn in die Zelle ein.
- Um etwas, z.B. ein Bild, aus einer Datei zu importieren, verwenden Sie den Befehl **Bearbeiten | Einfügen | Kodierte externe Datei**[1301]. Dadurch wird ein Bild als Base64-kodierter String in das Dokument eingefügt und in der JSON-Grid-Ansicht als Bild angezeigt. Für den Import werden die folgenden Bilddateiformate unterstützt: `PNG`, `JPEG`, `BMP`, `GIF`, `TIFF`.

XPath kopieren
Dieser Befehl kopiert einen XPath 3,1 Locator-Ausdruck in die Zwischenablage. Der Ausdruck beginnt an der Dokument-Root, unter der sich der ausgewählte Node befindet.

XPointer kopieren
Dieser Befehl kopiert einen XPointer-Ausdruck zum Aufrufen des ausgewählten Nodes in die Zwischenablage. Nähere Informationen dazu finden Sie unter **XPointer/JSON-Pointer kopieren**[1301].

Filter, Filter auf Fokus
Der **Filter**-Befehl ist ein Ein/Aus-Befehl. Dieser Befehl fügt einen Filter zum ausgewählten Element hinzu oder deaktivert einen Filter. Beachten Sie, dass Sie, nachdem Sie mit diesem Befehl einen Filter hinzugefügt haben, einen Filterausdruck eingeben müssen, da der Filter sonst automatisch entfernt wird. Nachdem ein Filter hinzugefügt wurde, können Sie ihn durch erneute Auswahl des Befehls oder Klick auf den Filter deaktivieren. Um den Filter zu entfernen, klicken Sie in der Symbolleiste auf *Alle Filter entfernen*.

Der Befehl **Filter auf Fokus** (i) erstellt am Root-Element, das einen XPath-Ausdruck zur Suche des ausgewählten Node enthält, einen Filter und (ii) filtert die Anzeige der Grid-Ansicht, um den ausgewählten Node (und dessen Nachfahren) direkt unterhalb des Root-Elements anzuzeigen. Dies ist nützlich, wenn nur der ausgewählte Node angezeigt werden soll. Um den Filter zu entfernen, klicken Sie in der Inhaltszelle des Root-Elements auf das Filtersymbol.

Alles erweitern
Dieser Befehl ist aktiviert, wenn die ausgewählte Komponente oder eine ihrer untergeordneten Komponenten reduziert angezeigt wird. Die Komponente und alle ihre untergeordneten Komponenten werden dadurch erweitert angezeigt.

Unselektierte reduzieren
Mit diesem Befehl werden alle Komponenten mit Ausnahme der ausgewählten Komponente und ihrer Vorfahren reduziert.

Textrichtung
Dieser Befehl steht für Zellen, die Text enthalten, zur Verfügung. Sie wechseln damit die Leserichtung, damit in der Zelle beginnend entweder von links oder von rechts gelesen wird. Dies ist bei Verwendung von Sprachen wie Arabisch oder Hebräisch nützlich.

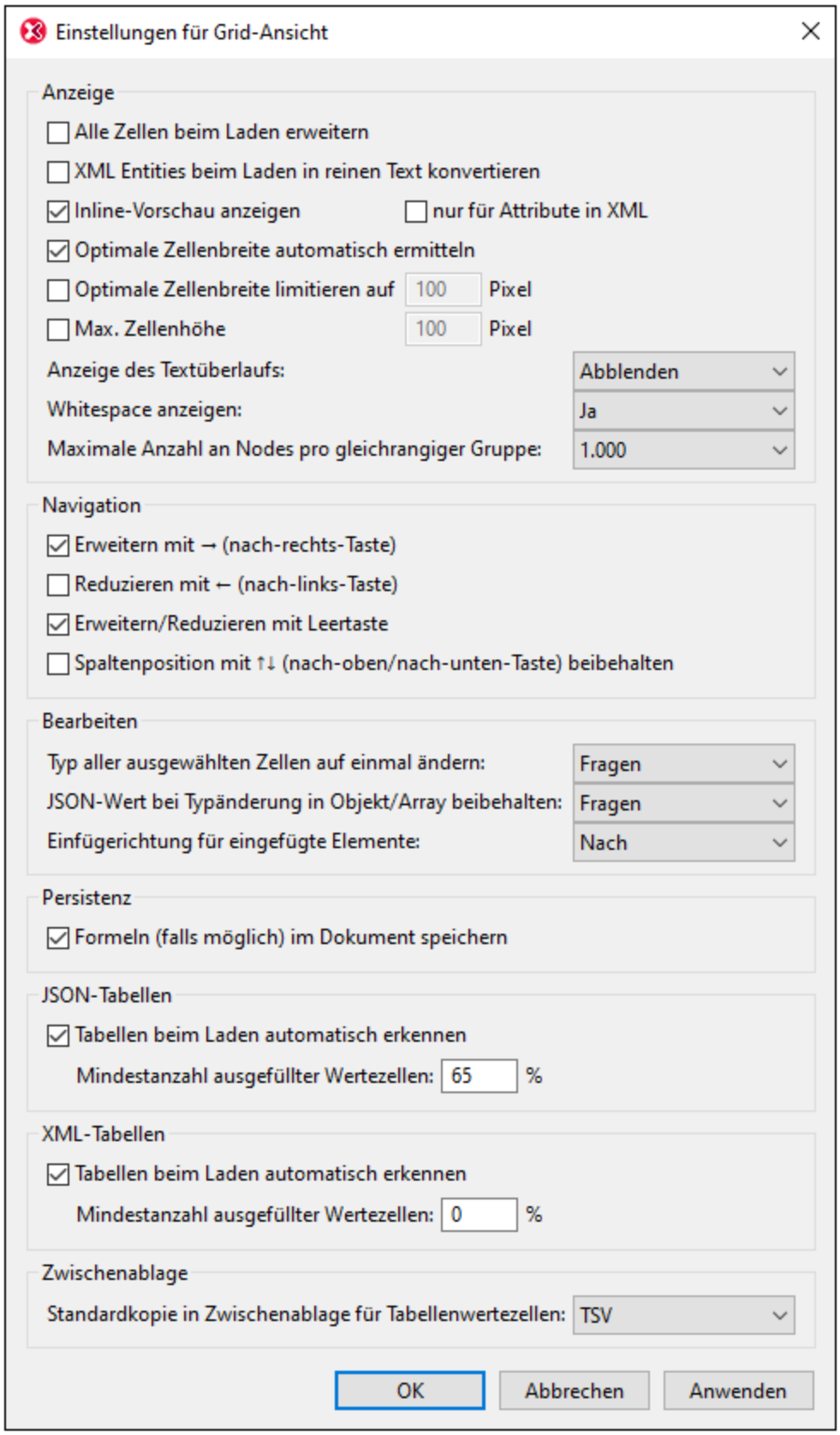
5.4.11 Einstellungen der Grid-Ansicht

Die Einstellungen der Grid-Ansicht werden im Dialogfeld "Einstellungen für Grid-Ansicht" *(Abbildung unten)* definiert, welches in der Grid-Ansichts-Symbolleiste[296] über die Schaltfläche "Einstellungen" aufgerufen wird..

	Schaltfläche "Einstellungen für Grid-Ansicht) in der Grid-Ansichts-Symbolleiste 296.

Weiter unten finden Sie eine Beschreibung der Einstellungen der Grid-Ansicht. Beachten Sie, dass diese Einstellungen für die Grid-Ansicht aller Dokumente (XML, JSON, DTD) gelten.

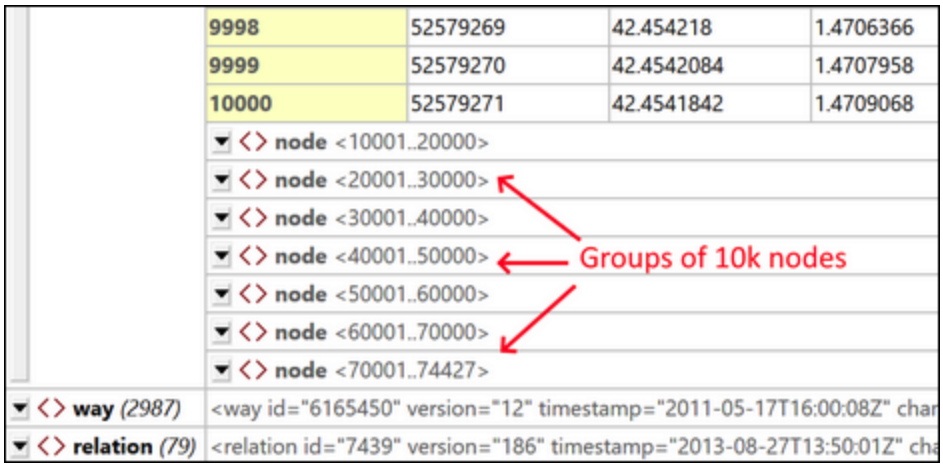
© 2016-2022 Altova GmbH


Einstellungen für Grid-Ansicht
Anzeige
Alle Zellen beim Laden erweitern
XML Entities beim Laden in reinen Text konvertieren
Inline-Vorschau anzeigen
nur für Attribute in XML
Optimale Zellenbreite automatisch ermitteln
Optimale Zellenbreite limitieren auf 100 Pixel
Max. Zellenhöhe 100 Pixel
Anzeige des Textüberlaufs: Abblenden
Whitespace anzeigen: Ja
Maximale Anzahl an Nodes pro gleichrangiger Gruppe: 1.000
Navigation
Erweitern mit → (nach-rechts-Taste)
Reduzieren mit ← (nach-links-Taste)
Erweitern/Reduzieren mit Leertaste
Spaltenposition mit ↑↓ (nach-oben/nach-unten-Taste) beibehalten
Bearbeiten
Typ aller ausgewählten Zellen auf einmal ändern: Fragen
JSON-Wert bei Typänderung in Objekt/Array beibehalten: Fragen
Einfügerichtung für eingefügte Elemente: Nach
Persistenz
Formeln (falls möglich) im Dokument speichern
JSON-Tabellen
Tabellen beim Laden automatisch erkennen
Mindestanzahl ausgefüllter Wertezellen: 65 %
XML-Tabellen
Tabellen beim Laden automatisch erkennen
Mindestanzahl ausgefüllter Wertezellen: 0 %
Zwischenablage
Standardkopie in Zwischenablage für Tabellenwertezellen: TSV
OK
Abbrechen
Anwenden

end

Anzeige
Die Kontrollkästchen im Abschnitt *Anzeige* sind selbsterklärend. Im Folgenden finden Sie einige klarstellende Anmerkungen.

- Wenn beim Laden nicht alle Zelle erweitert werden, werden der Root-Node und alle seine Nachfahren reduziert angezeigt. Sie müssen dann jeden Node erweitern, wenn Sie tiefer in die Hierarchie eintauchen möchten.
- Wenn die Option *XML-Entities beim Laden in reinen Text konvertieren* aktiv ist, werden XML-Entities in der Grid-Ansicht als reiner Text der jeweiligen Entity geladen und nicht zu ihrer jeweiligen Darstellung als Glyph aufgelöst.
- Wenn *Inline-Vorschau anzeigen* nicht aktiviert ist, wird anstelle der Vorschau der Zelle nur die Indexnummer des Elements in der Zelle angezeigt. Wenn die Inline-Vorschau aktiviert ist, haben Sie die Wahl zwischen einer Vorschau, die (i) sowohl den Elementinhalt als auch Attribute enthält oder (ii) einer, die nur Attribute enthält. Wenn Sie sich für zweiteres entscheiden, aktivieren Sie *nur für Attribute...*; wenn Sie ersteres möchten, deaktivieren Sie *nur für Attribute*. Beachten Sie, dass nur der erste Teil des Inline-Inhalts einer Zelle angezeigt wird; um den gesamten Inhalt zu sehen, platzieren Sie den Mauszeiger über den Start-Tag eines Elements.
- Wenn die optimale Breite aktiviert wurde, wird die gesamte Breite des Rasters angezeigt. Zu diesem Zweck wird Text in einigen Zellen umbrochen.
- Wenn Text über die Breite einer Zelle hinausragt, kann dies entweder durch heller werdenden Text oder Auslassungspunkte gekennzeichnet werden.
- Sie können die Anzeige von Whitespaces in Rasterzellen ein- oder ausschalten. Ein Leerzeichen wird in Form eines vertikal zentrierten Punkts und ein Tabulatorzeichen als Pfeil angezeigt. Ein Zeilenendezeichen wird als Wagenrücklauf in der Zelle angezeigt.
- Gleichrangige Nodes können in gleichrangigen Gruppen von 100, 1.000 oder 10.000 Nodes *(siehe Abbildung unten)* angezeigt werden. Dies ist aus zwei Gründen nützlich: (i) Wenn Sie in Dateien mit vielen gleichrangigen Nodes Platz für die Anzeige sparen möchten, können Sie gleichrangige Nodes in großen Gruppen zusammenfassen; (ii) Wenn Sie nur eine Teilmenge der gleichrangigen Nodes verarbeiten möchte (und somit Verarbeitungszeit sparen wollen), können Sie die gewünschten gleichrangigen Gruppen, die verarbeitet werden sollen, erweitern und die Verarbeitungsaktion nur an diesen Nodes ausführen; z.B. wenn Sie eine Suchen-und-Ersetzen-Aktion durchführen möchten oder den Typ eines Node konvertieren möchten (siehe *Bearbeiten* weiter unten).



Navigation
Sie können mit Hilfe der Pfeiltasten durch das Raster navigieren. Diese Einstellungen bieten clevere Optionen für die Verwendung der Tasten.

 © 2016-2022 Altova GmbH

- *Erweitern mit nach-rechts-Taste:* Wenn eine Zelle reduziert angezeigt wird, können Sie den Eintrag in der Zelle mit Hilfe der *nach-rechts*-Taste erweitern. Falls der Zelleneintrag nicht reduziert angezeigt wird, gelangen Sie mit Hilfe der *nach-rechts*-Taste zur nächsten rechts davon gelegenen Zelle (dies kann auch eine Child-Zelle sein, wenn die nächste rechts gelegene Zelle ein Child ist). Wenn diese Option nicht aktiviert ist, halten Sie mit der *nach-rechts*-Taste bei einer reduziert angezeigten Zelle an. Beachten Sie, dass die Funktion *Erweitern mit nach-rechts-Taste* nicht auf Zellen innerhalb von Tabellen angewendet wird; in Tabellenzellen gelangen Sie damit zur nächsten rechts davon gelegenen Zelle.
- *Reduzieren mit nach-links-Taste:* Wenn Sie mit Hilfe der *nach-links*-Taste nach links gehen, müssen Sie sich in der Dokumenthierarchie irgendwann nach oben bewegen. Wenn diese Option aktiviert ist, werden Elemente, die reduziert werden können, bei Drücken der *nach-links*-Taste reduziert; andernfalls werden solche Elemente nicht reduziert, obwohl sich der Fokus auf das übergeordnete Element verschiebt. Beachten Sie, dass die Funktion *Reduzieren mit nach-links-Taste* nicht auf Zellen innerhalb von Tabellen angewendet wird; in Tabellenzellen gelangen Sie damit zur nächsten links davon gelegenen Zelle.
- *Erweitern/Reduzieren mit Leertaste:* Die Leertaste fungiert als Ein/Aus-Befehl zum Erweitern/Reduzieren eines Elements. Sie kann somit als zusätzliche Hilfstaste beim Navigieren im Raster verwendet werden.
- *Spaltenposition mit nach-oben/nach-unten-Taste beibehalten:* Mit Hilfe der *nach-oben*- und *nach-unten*-Taste können Sie sich durch die Zellen des Rasters nach oben bzw. nach unten bewegen - auch durch übergeordnete und untergeordnete Elemente, die sich auf verschiedenen Ebenen und daher in unterschiedlichen Spalten befinden. Wenn diese Option aktiviert ist, werden Ebenen, die in anderen als der aktuellen Spalte dargestellt werden, übersprungen. Dies funktioniert, z.B. folgendermaßen: Angenommen, der Cursor befindet sich in der Spalte für das Element **`subject/course/books/book/title`**. Wenn die Option *Spaltenposition beibehalten* aktiv ist, können Sie mit Hilfe der *nach-oben*- und *nach-unten*-Pfeiltaste nur durch die Titel der Bücher navigieren (ohne in die Spalten **`book`**, **`books`**, **`course`** oder **`subject`** oder Spalten für untergeordnete Elemente von **`Title`** zu gelangen).

Bearbeiten

Die Kontrollkästchen im Abschnitt *Anzeige* sind selbsterklärend. Im Folgenden finden Sie einige klarstellende Anmerkungen.

- Wenn Sie den Typ mehrerer ausgewählter Zellen ändern, stehen die folgenden Optionen dazu, wie Sie damit fortfahren möchten, zur Verfügung: *Immer, Nie* oder *Fragen* (für die Entscheidung durch den Benutzer).
- Wenn Sie einen JSON-Typ von einem atomaren Typ in ein Objekt oder Array oder umgekehrt ändern, stehen die folgenden Optionen dazu, wie Sie damit fortfahren möchten, zur Verfügung: (i) *Fragen* (ob der Wert des atomaren Typs als Wert eines unbenannten Child-**`Schlüssel:Wert`**-Paars beibehalten werden soll oder ob er verworfen werden soll); (ii) *Immer*: (Der Wert in einem nicht benannten Child-**`Schlüssel:Wert`**-Paar soll immer beibehalten werden); (iii) *Nie*: (Der Wert soll nie beibehalten werden, d.h. der Wert soll verworfen werden).
- Mit der Option *Einfügerichtung* wird festgelegt, ob eine Auswahl in der Zwischenlage oberhalb oder unterhalb der ausgewählten Zelle eingefügt werden soll.

Persistenz

Filter und Formeln können in den Metadaten von XML- und JSON-Dokumenten gespeichert werden. Formeln werden in XML-Dokumenten als Processing Instructions und in einigen Arten von JSON-Dokumenten als Kommentare gespeichert. Mit diesen Einstellungen wird festgelegt, wie beim Speichern von Filtern und Formeln verfahren werden soll. Im Fall von Formeln bezieht sich "*wenn möglich*" darauf, dass Formeln nicht in allen

JSON-Dokumenten (sondern nur in JSON5- und JSONC-Dokumenten) als Kommentare gespeichert werden können.

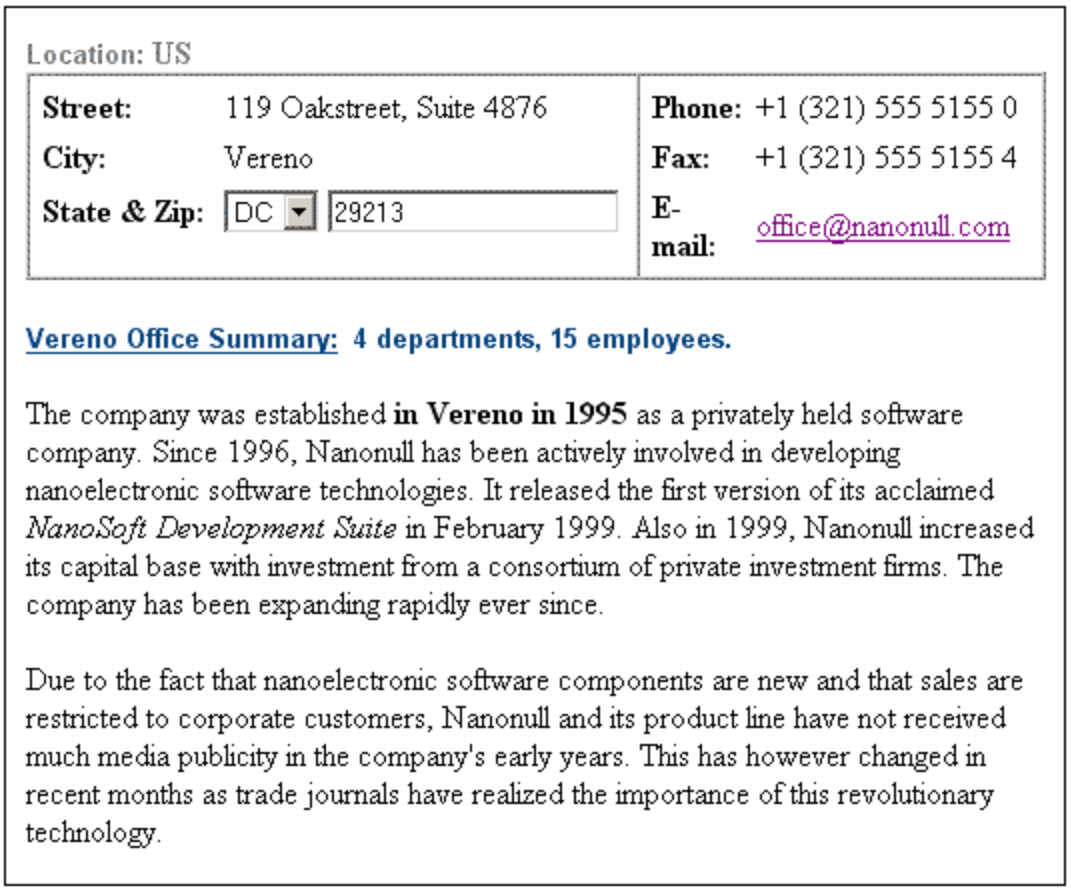
JSON-Tabellen, XML-Tabellen
Wenn die Einstellung zum automatischen Erkennen von Tabellen der-Grid-Ansicht beim Laden aktiviert wurde, können Sie auswählen, wie viele Tabellenzellen mindestens ausgefüllt sein müssen, damit Tabellen als solche erkannt werden. Wenn die Anzahl der ausgefüllten Tabellenzellen diesen Wert nicht übersteigt, wird die Struktur als normales Raster angezeigt, wobei sich wiederholende Elemente nacheinander aufgelistet werden.

Zwischenablage
Sie können auch auswählen, ob der Inhalt der Zwischenablage als tabellengetrennte Werte (TSV = tab-separated values) oder (je nach Dokumenttyp) als XML/JSON gespeichert werden soll. Dies ist eine sehr nützliche Funktion. Wenn Sie eine Tabelle aus der Zwischenablage in ein anderes Dokument kopieren möchten, können Sie mit dieser Einstellung auswählen, ob die kopierte Tabelle als TSV oder mit Markup-Code gespeichert werden soll. (Um den Unterschied zu sehen, fügen Sie eine Tabelle in einen Text-Editor ein, nachdem Sie diese in jedem dieser Formate in die Zwischenablage kopiert haben.)

© 2016-2022 Altova GmbH

5.5 XML in der Authentic-Ansicht

In der Authentic-Ansicht kann der Benutzer ein XML-Dokument wie ein normales Textdokument bearbeiten (*Abbildung unten*). Der XML Markup-Code und der gesamte restliche Text, bei dem es sich nicht um Inhalt handelt, kann für denjenigen, der das Dokument bearbeitet, ausgeblendet werden. Dies ist nützlich, wenn jemand das Dokument bearbeitet, der nicht mit XML vertraut ist. Er kann dadurch ein gültiges XML-Dokument erstellen und sich dabei ausschließlich auf den Inhalt des Dokuments konzentrieren.



Die Authentic-Ansicht eines Dokuments ist aktiviert, wenn dem XML-Dokument ein StyleVision Power Stylesheet (SPS) zugewiesen wurde. Ein SPS basiert auf derselben Schemaquelle wie der, auf der das XML-Dokument basiert, und definiert die Struktur des XML-Dokuments. Im SPS sind außerdem das Layout und die Formatierung des Dokuments in der Authentic-Ansicht definiert. So werden in dem Dokument in der Abbildung oben z.B. die folgenden Authentic-Formatierungs- und Bearbeitungsfunktionen verwendet:

- Formatierung von Absätzen und anderen Blöcken
- Tabellenstrukturen
- Textformatierung wie z.B. Farbe und Schriftart
- Mit Hilfe von Auswahllisten (siehe Felder "State & Zip") kann der Benutzer einen Eintrag aus einer Gruppe von gültigen Auswahlmöglichkeiten auswählen. Diese Listeneinträge können wie in diesem Fall oben aus Schema-Enumerationen übernommen werden.

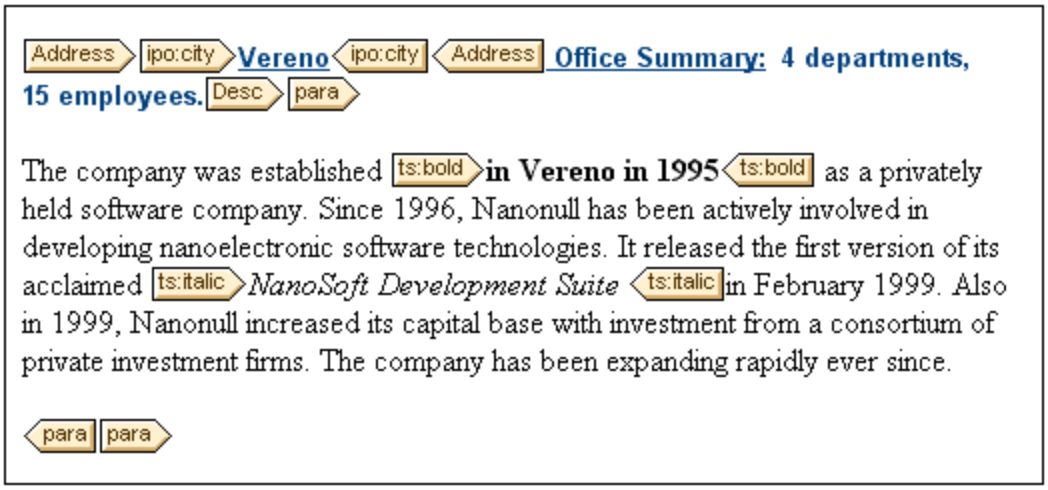
- Zusätzliche Daten können anhand der Daten im Dokument berechnet und dargestellt werden (Im Beispiel oben wurden z.B. die Information in der Office Summary nicht vom Benutzer eingegeben sondern anhand anderer Daten im Dokument berechnet)

SPSs werden speziell zum Anzeigen und Bearbeiten von XML-Dokumenten in der Authentic-Ansicht und zum Generieren von Standardausgabedateien (wie z.B. HTML-, PDF-, RTF- und Word 2007-Dokumenten) anhand von XML-Dateien erstellt. SPSs werden mit Altova StyleVision erstellt.

Bearbeiten der Dokumentstruktur

Sie können jederzeit gültige Nodes zum Dokument hinzufügen. Wählen Sie dazu die Stelle im Dokument aus und fügen Sie den gewünschten Node über die Eingabehilfen (Elemente und Attribute) oder das Kontextmenü hinzu. Die Nodes, die an einer bestimmten Stelle verfügbar sind, beschränken sich auf die Nodes, die an dieser Stelle laut Schema gültig sind und als gleichrangige oder untergeordnete Nodes hinzugefügt werden können. Wenn sich der Cursor z.B. innerhalb eines Absatzes befindet, können Sie einen weiteren Absatz anhängen, wenn dies laut Schema zulässig ist.

Bei der Bearbeitung der Struktur eines XML-Dokuments in der Authentic-Ansicht ist es manchmal nützlich, den Markup-Code des Dokuments zu sehen. Sie können daher den Markup-Code in Form von Tags (*Abbildung unten*) mit dem Befehl **Authentic | Große Markup-Symbole einblenden** (oder über die entsprechende Symbolleisten-Schaltfläche) einblenden.



Bearbeiten von Inhalt

Inhalt wird durch Eingabe des Inhalts in die Nodes des Dokuments erstellt und bearbeitet. Entities und CDATA-Abschnitte können über das Kontextmenü hinzugefügt werden (Entities auch über die Eingabehilfe "Entities).

Mehr über die Bearbeitung in der Authentic-Ansicht

Nähere Informationen über die Bearbeitung in der Authentic-Ansicht finden Sie im Abschnitt "Authentic-Ansicht".

© 2016-2022 Altova GmbH

5.6 Eingabehilfen (Textansicht, Authentic-Ansicht)

Es gibt für XML-Dokumente in der Text- und der Authentic-Ansicht drei Eingabehilfen: Elemente, Attribute und Entities. Wenn ein Element über die Element-Eingabehilfe hinzugefügt wird, können Sie es je nach der im Dialogfeld "Optionen" im Abschnitt "Bearbeiten" 1609 gewählten Einstellung mit seinen obligatorischen Child-Elementen, Attributen, allen Child-Elementen oder keinem Child-Element oder Attribut einfügen. Wenn leere Attribute hinzugefügt werden, werden sie mit Anführungszeichen hinzugefügt.

Beachten Sie, dass die Eingabehilfen in den verschiedenen Ansichten unterschiedlich aussehen, je nachdem, welche Funktionalitäten in der entsprechenden Ansicht verfügbar sind.

Element-Eingabehilfe

Beachten Sie die folgenden Punkte:

- *Textansicht:* Elemente werden an der Cursorposition eingefügt. Nicht verwendete Elemente werden rot, verwendete Elemente grau angezeigt. Obligatorische Elemente sind mit einem Ausrufezeichen "!" vor dem Namen des Elements markiert.
- *Authentic-Ansicht:* Elemente können vor oder hinter dem ausgewählten Element oder innerhalb dieses Elements eingefügt werden. Des Weiteren gibt es eine Dokumentstruktur, in der Sie sehen, an welcher Stelle in der Dokumentstruktur sich das gerade ausgewählte Element befindet. Nähere Informationen zum Bearbeiten in der Authentic-Ansicht finden Sie im Abschnitt "Authentic-Ansicht".

Attribut-Eingabehilfe

Beachten Sie die folgenden Punkte:

- *Textansicht:* Wenn Sie den Cursor in der Textansicht innerhalb des Start-Tags eines Elements und hinter einem Leerzeichen positionieren, werden die für dieses Element deklarierten Attribute angezeigt. Nicht verwendete Attribute werden rot angezeigt, verwendete grau. Obligatorische Attribute sind durch ein Ausrufezeichen "!" vor dem Namen des Attributs gekennzeichnet.

  Um ein Attribut einzufügen, doppelklicken Sie auf das gewünschte Attribut. Das Attribut wird nun an der Cursorposition zusammen mit einem Ist-gleich-Zeichen und Anführungszeichen, innerhalb derer der Attributwert gesetzt wird, angezeigt. Der Cursor wird zwischen die Anführungszeichen gesetzt, sodass Sie den Attributwert direkt eingeben können.
- *Authentic-Ansicht:* Wenn ein Element ausgewählt ist, sehen Sie die für dieses Element deklarierten Attribute. Geben Sie den Wert des Attributs in die Eingabehilfe ein.

Entities-Eingabehilfe

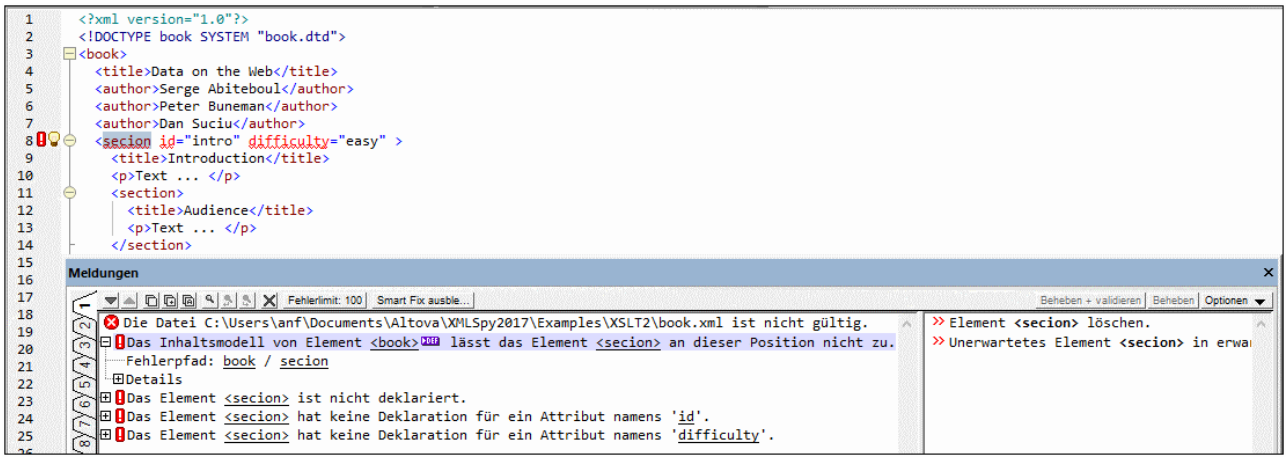
Alle geparsten oder ungeparsten Entities, die innerhalb einer Zeile (innerhalb des XML-Dokuments) oder in einer externen DTD deklariert sind, werden in der Entities-Eingabehilfe angezeigt. In allen drei Ansichten (Text, Grid und Authentic) wird eine Entity durch Doppelklick auf die Entity an der Cursorposition eingefügt. In der Grid-Ansicht werden Entities auf den Registern "Anhängen", "Einfügen" und "Child hinzufügen" angezeigt.

Hinweis: Wenn Sie eine interne Entity einfügen, müssen Sie Ihr Dokument speichern und erneut öffnen, damit die Entity in der Entities-Eingabehilfe angezeigt wird.

© 2016-2022 Altova GmbH

5.7 Validieren eines XML-Dokuments

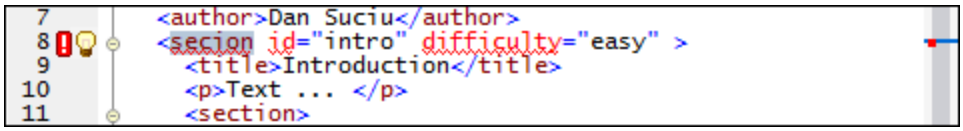
Mit dem Befehl **XML** | **XML validieren (F8)** wird ein XML-Dokument anhand einer damit verknüpften DTD, eines XML-Schemas oder eines anderen Schemas validiert. Bei erfolgreicher Validierung wird im Fenster "Meldungen" eine entsprechende Meldung angezeigt: Andernfalls werden die Fehlerursachen im Bereich auf der linken Seite (*siehe Abbildung unten*) angezeigt. Wenn im linken Bereich eine Fehlerursache ausgewählt wird, so werden im rechten Bereich Smart Fix-Vorschläge dazu angezeigt, falls vorhanden. Die Smart Fix-Vorschläge basieren auf den Informationen im verknüpften Schema. Um einen Smart Fix-Vorschlag anzuwenden, (i) doppelklicken Sie entweder darauf oder (ii) wählen Sie ihn aus und klicken Sie entweder auf die Option **Beheben** oder **Beheben + validieren** (*siehe Abbildung unten*).



Fehlermarkierungen und Smart Fix-Vorschläge

Fehlermarkierungen

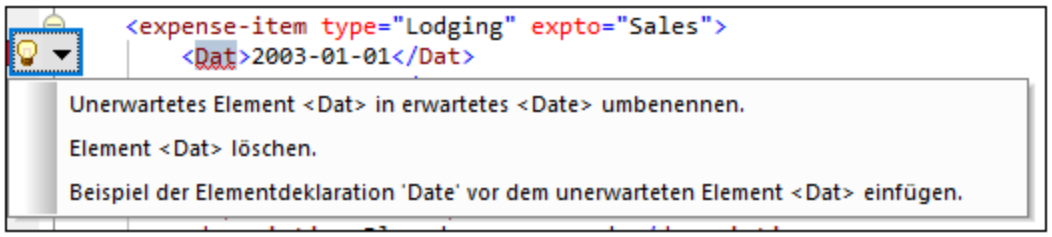
In der Textansicht finden Sie zwei weitere Markierungen für einen Validierungsfehler (*siehe Abbildung unten*): (i) ein rotes Ausrufezeichen in der Zeilennummernleiste und (ii) ein rotes Markierungsquadrat in der Bildlaufleiste (auf der rechten Seite des Fensters).



Beachten Sie, dass das rote Markierungsquadrat auf der linken Seite der (auf der rechten Seite des Fensters angezeigten) Bildlaufleiste (*siehe Abbildung oben*) angezeigt wird. Dies wird hier erwähnt, weil in der Bildlaufleiste zwei weitere Arten von Markierungsquadraten angezeigt werden: (i) eines für markierte Textstellen (braun, linke Seite der Bildlaufleiste; *siehe* Navigation im Dokument 154) und (ii) eines für gefundene Textinstanzen (braun, rechte Seite der Bildlaufleiste; *siehe* Suchen und Ersetzen 157).

Smart Fix-Vorschläge

Falls es zu einem Fehler einen Smart Fix-Vorschlag gibt, wird in der Zeile mit dem Fehler ein Glühbirnensymbol angezeigt (*siehe Abbildung unten*). Wenn Sie den Mauszeiger über das Symbol platzieren, erscheint ein Popup-Fenster, in dem Smart Fix-Vorschläge aufgelistet werden (*siehe Abbildung*). Wählen Sie einen Vorschlag aus, um ihn sofort anzuwenden.



Beachten Sie die folgenden Punkte:

- Validierungsfehlermarkierungen und Smart Fix-Vorschläge stehen für Dokumenttypen zur Verfügung, die in XMLSpy validiert werden können, z.B. JSON-Dokumente.
- Die oben beschriebenen Validierungsfehlermarkierungen und Smart Fix-Vorschläge, werden erst bei Ausführung des Befehls **XML | Validieren (F8)** aktualisiert; sie werden nicht im Hintergrund aktualisiert. Sie müssen den Befehl **Validieren (F8)** erneut ausführen, um zu überprüfen, ob der Fehler wirklich behoben wurde.

Nähere Informationen zum Validieren eines XML-Dokuments finden Sie unter der Beschreibung des Befehls **Validieren**[1353].

Bei der Bearbeitung validieren

Wenn der *Bei Bearbeitung validieren*-Modus aktiv ist, werden, werden Wohlgeformtheits- und Validierungsüberprüfungen durchgeführt, während Sie ein Dokument in der Textansicht (oder der JSON-Grid-Ansicht) bearbeiten. Damit das Dokument (nicht nur auf Wohlgeformtheit geprüft, sondern auch) validiert werden kann, muss dem XML-Dokument eine DTD oder ein XML-Schema zugewiesen sein (einem JSON-Dokument muss ein JSON-Schema zugewiesen worden sein). Fehler werden durch entsprechende Markierungen (siehe oben) am linken Rand und in den Zeilen, die die Fehler enthalten, angezeigt.

Der *Bei Bearbeitung validieren*-Modus kann entweder (i) über den Menübefehl **XML | Bei Bearbeitung validieren**[1359], (ii) die Symbolleisten-Schaltfläche **Bei Bearbeitung validieren** oder (iii) die Option *Bei Bearbeitung* der Validierungseinstellungen des Dialogfelds "Optionen"[1603] aktiviert bzw. deaktiviert werden.

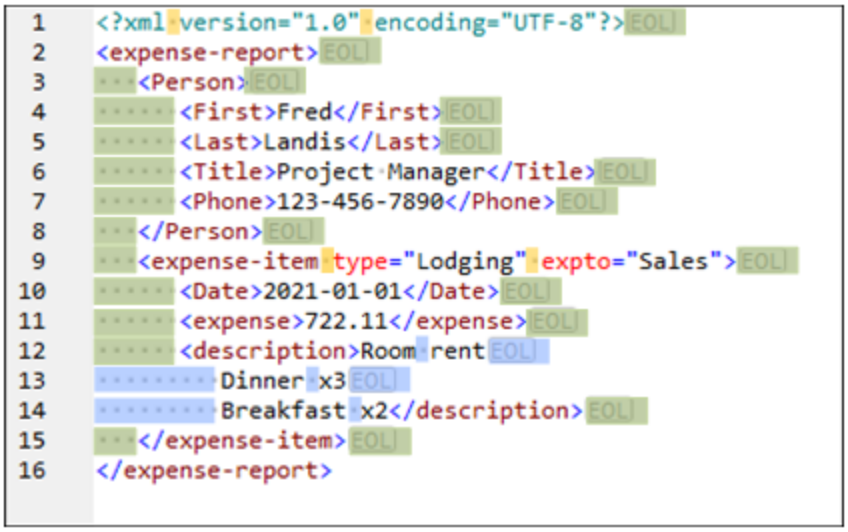
© 2016-2022 Altova GmbH

5.8 Whitespace

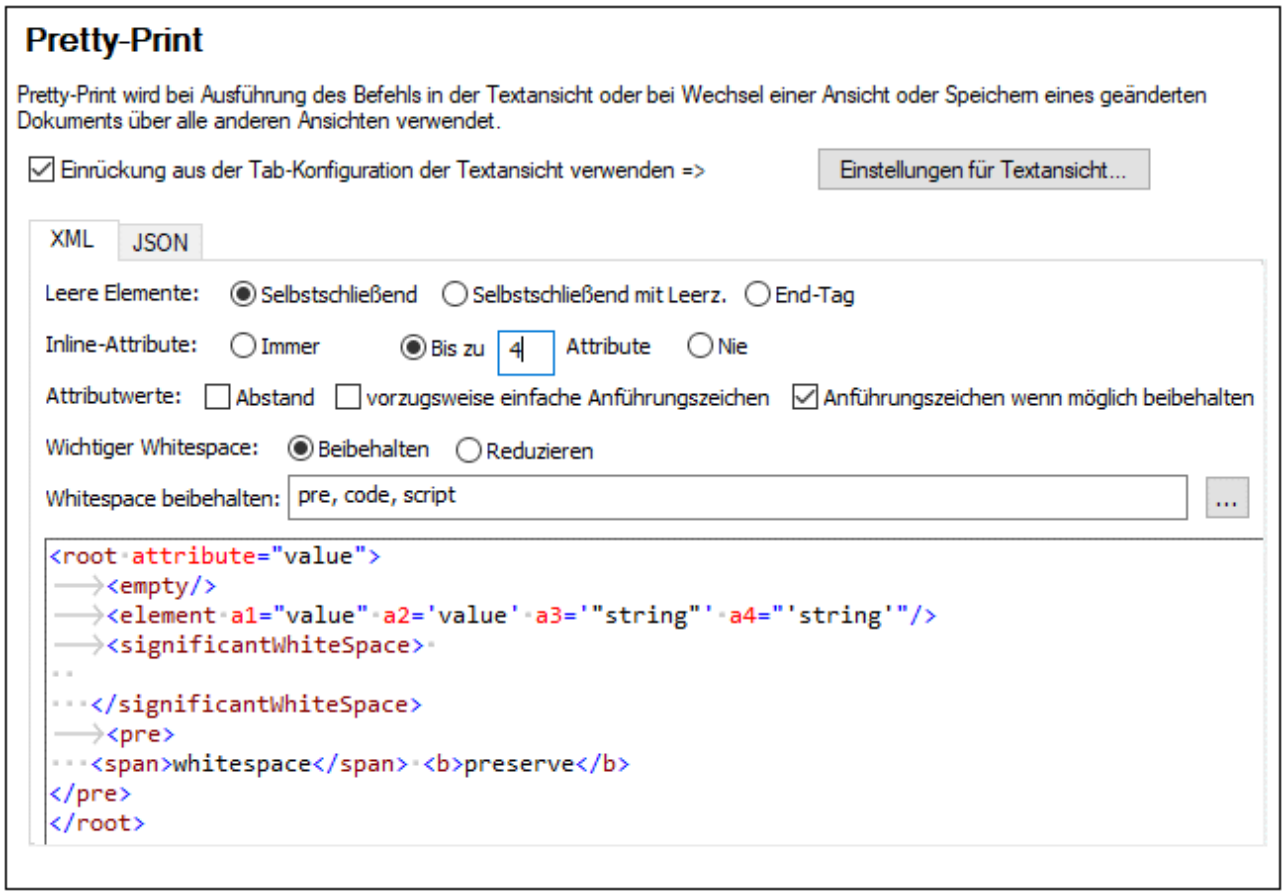
Whitespace-Zeichen sind Leerzeichen, Tabulator, Wagenrücklauf (Carriage Return) und Zeilenvorschub (Line Feed). Sie können Whitespace-Markierungen (Leerzeichen, Tabulator und Zeilenendemarkierungen über das Dialogfeld Einstellungen für Textansicht (**Ansicht | Einstellungen für Textansicht**[1506]) einblenden.

Whitespace-Zeichen kommen aus folgenden Gründen in einem XML-Dokument vor:

- Aus Gründen der XML-Syntax, normalerweise als Trennzeichen zwischen XML-Konstrukten. Solche Whitespace-Zeichen sind in der Abbildung unten gelb markiert.
- Wichtige Whitespace-Zeichen, die zwischen einem Element, Attribut oder einer Processing Instruction vorhanden sind, sollten nicht ignoriert werden, da sie von Bedeutung sind. Diese Zeichen sind in der Abbildung unten blau markiert.
- Unwichtige Whitespace-Zeichen, die zwischen zwei Elementen ohne gleichrangige Text-Nodes vorkommen. Unwichtige Whitespace-Zeichen kommen somit nur in Elementen vor, die nicht vom Typ "mixed content" sind. Sie dienen zu Formatierungszwecken und haben keine Bedeutung. Unwichtige Whitespace-Zeichen sind in der Abbildung unten grün markiert.



Whitespace-Zeichen werden in XMLSpy hinzugefügt, wenn Sie ein Dokument mit Pretty-Print (**Bearbeiten Pretty-Print**[1306]) formatieren. Dabei werden unwichtige Whitespace-Zeichen hinzugefügt, um das Dokument so zu formatieren, dass seine Struktur klar ersichtlich wird. Bei Anwendung von Pretty-Print werden wichtige Whitespace-Zeichen, je nachdem, welche Optionen gerade für Pretty-Print eingestellt sind, eventuell auch reduziert (*siehe Abbildung unten*).



Die folgenden Einstellungen wirken sich in den Pretty-Print-Optionen[1611] auf die Behandlung von Whitespace-Zeichen aus.

- *Wichtiger Whitespace* kann beibehalten oder reduziert werden. Wenn diese Option auf *Reduzieren* gesetzt wird, können Sie Whitespace-Zeichen dennoch in bestimmten Elementen beibehalten, indem Sie diese Elemente zur Liste *Whitespace beibehalten* hinzufügen.
- Mit Hilfe der Option *Whitespace beibehalten* können Sie eine Liste von Elemente erstellen, in denen alle Whitespace-Zeichen (wichtige und unwichtige) beibehalten werden.
- Wenn in einem leeren Element ein wichtiges Whitespace-Zeichen vorhanden ist, wird dieses entfernt, wenn *Wichtiger Whitespace* auf *Reduzieren* gesetzt wurde. Mit der Einstellung der Option *Leere Elemente* wird festgelegt, wie leere Elemente angezeigt werden, wenn ihre wichtigen Whitespace-Zeichen entfernt wurden.

Anmerkung: Bei Wechsel zwischen Text- und Grid-Ansicht werden alle Änderungen, die sich aus dem Pretty-Printing ergeben, beibehalten. Bei einem Wechsel aus einer dieser Ansichten in eine andere Ansicht und umgekehrt werden die Änderungen nicht angewendet.

© 2016-2022 Altova GmbH

5.9 Einfügen von XML-Fragmenten

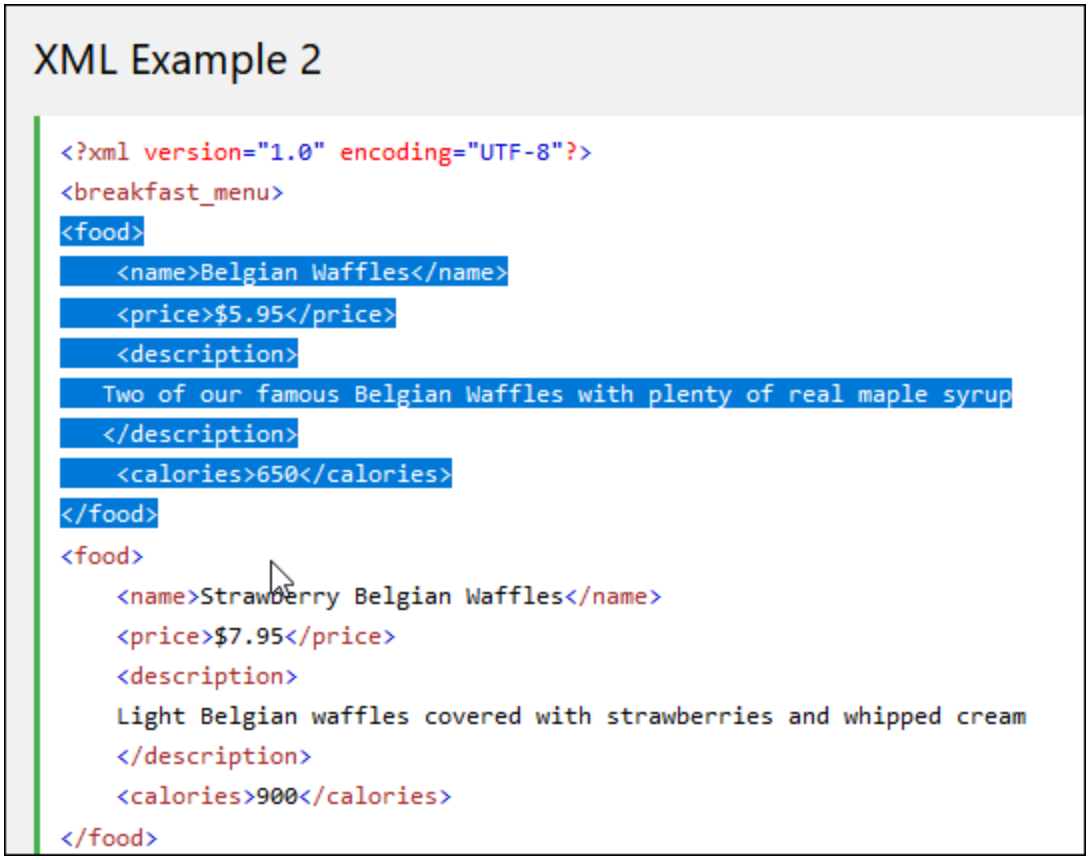
Sie können XML-Fragmente aus anderen Applikationen oder Webseiten einfügen. Diese Fragmente können auf zwei verschiedene Arten eingefügt werden:

- Sie können diese mit der Maus (Drag-and-Drop) in die Text- oder Grid-Ansicht ziehen. Wenn Sie ein Fragment mittels Drag-and-Drop in die Grid-Ansicht ziehen, helfen Ihnen die intelligenten Informationen in den Drag Overlays 309 bei der Entscheidung, an welcher Stelle das Fragment eingefügt werden soll.
- Sie können diese in die Text- oder Grid-Ansicht kopieren (Kopieren und Einfügen).

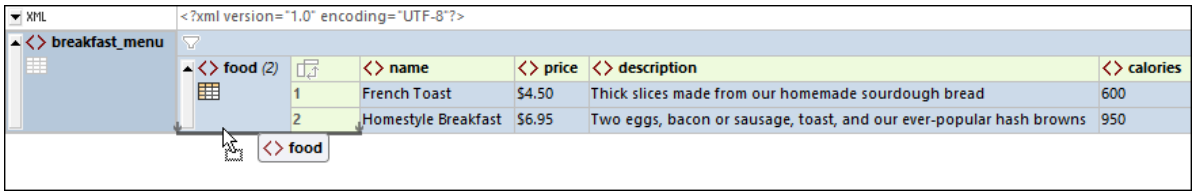
Beispiel

Im folgenden Beispiel wird gezeigt, wie ein Fragment einfach und schnell an der richtigen Stelle in einem XML-Dokument hinzugefügt werden kann.

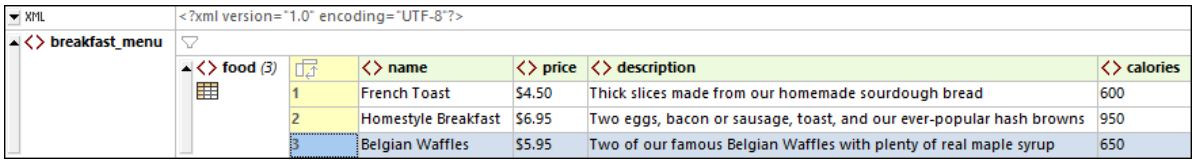
1. Das unten markierte Fragment (aus einem XML-Tutorial unter w3schools.com) wird ausgewählt. Es handelt sich hierbei um ein Element namens `food`, das eine Reihe von Child-Elementen enthält.



2. In der Abbildung unten sehen Sie die Grid-Ansicht eines XML-Dokuments, in der zwei `food`-Elemente als Zeilen einer Tabelle angezeigt werden. Wenn das Fragment aus der Webseite auf die `food`-Tabelle gezogen wird, erscheint ein Drag Overlay 702 mit der Information, dass das mit der Maus gezogene XML-Fragment als `food`-Element als letzte Zeile in die Tabelle `food` eingefügt wird.



3. Bei Loslassen der Maustaste wird das Element genau an der benötigten Stelle eingefügt - nämlich als das letzte **`food`**-Element-Child von **`breakfast_menu`** (*siehe Abbildung unten*).



© 2016-2022 Altova GmbH

5.10 Verarbeitung mit XSLT und XQuery

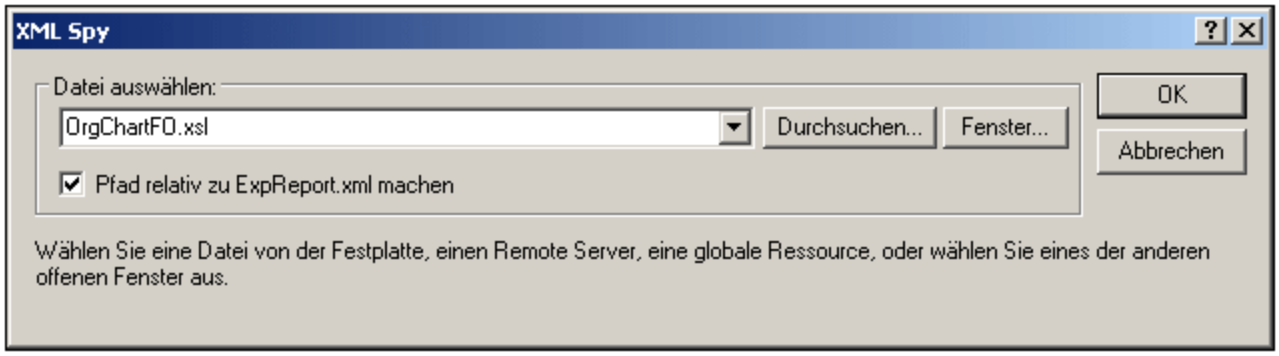
XML-Dokumente können mit XSLT- oder XQuery-Dokumenten verarbeitet werden um damit Ausgabedokumente zu erzeugen. XMLSpy hat einen integrierten XSLT 1.0-, XSLT 2.0-, XSLT 3.0-, XQuery 1.0- und XQuery 3.0-Prozessor. Über die Benutzeroberfläche stehen Ihnen die folgenden verarbeitungsbezogenen Funktionen zur Verfügung:

- Zuweisen von XSLT Stylesheets 345
- Gehe zu XSLT 346
- XSLT-Parameter und XQuery-Variablen 346
- XSLT-Transformationen 346
- XQuery-Ausführungen 346
- Automatisieren von XML-Aufgaben mit RaptorXML 346

Zuweisen von XSLT Stylesheets

Mit dem Befehl **XSL/XQuery | XSL zuweisen** können Sie einem XML-Dokument ein XSLT Stylesheet zuweisen (wählen Sie die Datei in dem Dialogfeld, das daraufhin angezeigt wird mit "Durchsuchen" aus, *Abbildung unten*). Die Zuweisung wird als Processing Instruction (PI) (Verarbeitungsanweisung) mit dem vom W3C definierten XSLT-Standardziel `xml-stylesheet` in das XML-Dokument eingegeben. Diese Zuweisung wird verwendet, wenn eine XSLT-Transformation aufgerufen wird (**XSL/XQuery | XSL-Transformation**).

Zusätzlich dazu kann ein XSLT-für-FO Stylesheet mit dem Befehl **XSL/XQuery | XSL:FO zuweisen** (wählen Sie die Datei in dem Dialogfeld, das daraufhin angezeigt wird mit "Durchsuchen" aus, *Abbildung unten*) zugewiesen werden. Die Zuweisung wird als Processing Instruction (PI) (Verarbeitungsanweisung) mit der Altova-Zieldatei `altova_xslfo` eingefügt. Diese Zuweisung wird verwendet, wenn eine XSLT-für-FO-Transformation aufgerufen wird (**XSL/XQuery | XS:FO-Transformation**).



Sie können zur Definition der XSLT-Datei auch eine globale Ressource auswählen. Eine globale Ressource ist ein Alias für eine Datei oder einen Ordner. Die Zieldatei bzw. der Zielordner kann durch Wechseln der aktiven Konfiguration der globalen Ressource über die Benutzeroberfläche gewechselt werden (Menübefehl **Extras | Aktive Konfiguration**). Mit Hilfe globaler Ressourcen können Sie also zwischen mehreren XSLT-Dateien wechseln, was für Testzwecke sehr nützlich ist. Eine Anleitung zur Verwendung globaler Ressourcen finden Sie unter Globale Altova Ressourcen 1061.

Wenn bereits eine Zuweisung für eine dieser PI-Ziele vorhanden ist, werden Sie gefragt, ob Sie die vorhandene Zuweisung überschreiben möchten.

Gehe zu XSLT

Mit dem Befehl **XSL/XQuery | Gehe zu XSL** wird die dem XML-Dokument zugewiesene XSLT-Datei geöffnet.

XSLT-Parameter und XQuery-Variablen

XSLT-Parameter und XQuery-Variablen können in dem Dialogfeld definiert, bearbeitet und gelöscht werden, das mit dem Befehl **XSL/XQuery | XSLT-Parameter / XQuery-Variablen** aufgerufen wird. Die hier definierten Parameter-/Variablenwerte werden für alle XSLT-Transformationen und XQuery-Ausführungen in XMLSpy verwendet. Diese Werte werden jedoch nicht an externe Prozessoren wie z.B. MSXML übergeben. Nähere Informationen zur Verwendung dieser Funktion finden Sie im Abschnitt "Menübefehle"[1414].

XSLT-Transformationen

Es stehen zwei Arten von XSLT-Transformation zur Verfügung:

- XSLT-Standardtransformation (**XSL/XQuery | XSL-Transformation**): Die Ausgabe der Transformation wird in einem neuen Fenster angezeigt, oder, wenn im Stylesheet so definiert, unter einem Dateipfad gespeichert. Der für die Transformation verwendete Prozessor wird im Dialogfeld "Optionen" im Abschnitt XSL[1633] definiert (**Extras | Optionen**[1633]).
- XSL-für-FO-Transformation (**XSL/XQuery | XSL-FO-Transformation**): Das XML-Dokument wird in zwei Schritten in ein PDF-Dokument transformiert. Im ersten Schritt wird das XML-Dokument mit Hilfe des XSLT-Prozessors, der im Dialogfeld "Optionen" (**Extras | Optionen**[1633]) im Abschnitt XSL[1633] definiert wurde, in ein FO-Dokument transformiert; beachten Sie, dass Sie (am unteren Rand des Registers) auch den XSLT-Prozessor auswählen können, der mit einigen FO-Prozessoren mitgeliefert wird, wie z.B. FOP. Im zweiten Schritt wird das FO-Dokument von dem FO-Prozessor, der im Dialogfeld "Optionen" (**Extras | Optionen**[1633]) im Abschnitt XSL[1633] definiert wurde, verarbeitet, um ein PDF-Dokument zu erzeugen.

Anmerkung: Ein FO-Dokument (dies ist eine bestimmte Art von XML-Dokument) kann mit dem Befehl "XSL:FO-Transformation" in PDF transformiert werden. Wenn es sich beim Quelldokument um ein FO-Dokument handelt, wird der zweite Befehlsschritt dieses zweistufigen Vorgangs direkt ausgeführt.

XQuery-Ausführung

Durch Klicken auf den Befehl **XSL/XQuery | XQuery-Ausführung** können Sie am aktiven XML-Dokument ein XQuery-Dokument ausführen. Sie werden aufgefordert, die XQuery-Datei anzugeben. Das Ergebnisdokument wird auf der GUI in einem neuen Fenster angezeigt.

Automatisieren von XML-Aufgaben mit RaptorXML

Altova RaptorXML ist eine Applikation zur XML-Validierung, XSLT-Transformation und XQuery-Ausführung. Sie kann über die Befehlszeile, über eine COM-Schnittstelle, in Java-Programmen und in .NET-Applikationen aufgerufen werden. So können mit RaptorXML z.B. Aufgaben wie die XSLT-Transformation automatisiert werden. Sie können z.B. eine Batch-Datei erstellen, die RaptorXML aufruft, um damit eine Reihe von Dokumenten zu transformieren. Nähere Informationen dazu finden Sie in der Dokumentation zu RaptorXML.

© 2016-2022 Altova GmbH

5.11 PDF-Schriftarten

Verarbeitung von Schriftarten durch das Formatierungsprogramm und den PDF Viewer

Das Formatierungsprogramm (z.B. FOP) erstellt das PDF-Dokument und der PDF Viewer (in der Regel der Reader von Adobe) liest es.

Für die Erstellung des PDF-Layout muss das Formatierungsprogramm Details über die im Dokument verwendeten Schriftarten wissen, v.a. muss es die Breitenangaben und alle im Dokument verwendeten Schriftzeichen kennen. Diese Informationen werden zur Berechnung von Zeilenlängen, der Abteilung der Wörter, Blocksatz usw. benötigt. Diese Informationen werden als Bemaßungen bezeichnet und sind zusammen mit der Schriftart gespeichert. Einige Formatierungsprogramme können die Bemaßungen direkt aus dem Ordner für die Schriftart auslesen. Andere (wie z.B. FOP) benötigen die Bemaßungen in einem speziellen Format, um sie lesen zu können. Wenn das Formatierungsprogramm die Bemaßungen einer Schriftart zur Verfügung hat, kann es ein Layout vom PDF erstellen. Stellen Sie daher sicher, dass dem von Ihnen verwendeten Formatierungsprogramm die Bemaßungsdateien (font metrics) aller in Ihrem Dokument verwendeten Schriftarten zur Verfügung stehen.

Das Formatierungsprogramm kann eine Schriftart entweder referenzieren oder sie in die PDF-Datei einbetten. Wenn die Schriftart referenziert wird, sucht der PDF Viewer (z.B Adobe Reader) diese Schriftart normalerweise zuerst in seinem eigenen Schriftart-Ressourcenordner (der die 14 Base Fonts enthält) und anschließend im Schriftartordner des Systems. Wenn die Schriftart verfügbar ist, wird diese bei der Anzeige des PDF verwendet. Andernfalls verwendet das Programm eine andere Schriftart aus seinem Ressourcenordner oder generiert einen Fehler. Diese andere Schriftart hat unter Umständen andere Bemaßungen und könnte daher Anzeigefehler verursachen.

Wenn das Formatierungsprogramm eine Schriftart in die PDF-Datei einbettet, verwendet der PDF Viewer die eingebettete Schriftart. Dabei kann der gesamte Zeichensatz oder nur ein Unterzeichensatz mit den im Dokument verwendeten Zeichen eingebettet werden. Dieser Faktor hat Auswirkungen auf die Größe der PDF-Datei und unter Umständen Copyright-Probleme im Zusammenhang mit der Verwendung der Schriftart (siehe Anmerkung unten). Eventuell können Sie in den Optionen für das Formatierungsprogramm festlegen, welche dieser beiden Optionen verwendet werden soll.

XMLSpy und PDF-Schriftarten

Ein PDF-Dokument wird in XMLSpy im Allgemeinen anhand eines XSL-FO-Dokuments (auch FO-Dokument genannt) durch Verarbeitung des XSL-FO-Dokuments mit einem externen FO-Prozessor wie z.B. FOP generiert. (Der Pfad zum FO-Prozessor wird im Dialogfeld "Optionen" definiert. Auf diese Art kann die FO-Verarbeitung von der Benutzeroberfläche von XMLSpy aus gestartet werden.)

Das XSL-FO-Dokument selbst wird durch Verarbeitung eines XML-Dokuments mit einem XSLT-Stylesheet generiert. (Sie können dazu entweder den (in XMLSpy integrierten) XSLT-Prozessor von Altova oder einen externen XSLT-Prozessor verwenden.)

Die Formatierung des PDF-Dokuments einschließlich der Schriftarteigenschaften des gesamten Texts ist im XSLT-FO-Dokument definiert. Wenn das von Ihnen verwendete Formatierungsprogramm die Bemaßungen der gewünschten Schriftart direkt aus der Schriftart auslesen kann, müssen Sie im Formatierungsprogramm nur den Pfad zur Schriftart konfigurieren. Wenn Sie allerdings FOP als Formatierungsprogramm verwenden, müssen Sie dafür die richtigen Font Metrics-Dateien für alle Schriftarten mit Ausnahme der Base-14-Schriftarten zur Verfügung stellen.

Bereitstellung von Schriftarten für das Formatierungsprogramm

Den meisten Formatierungsprogrammen (einschließlich FOP) stehen die Base 14 Fonts bereits zur Verfügung. Um dem Formatierungsprogramm die Schriftarten korrekt angeben zu können, müssen Sie die Namen kennen, an denen das Formatierungsprogramm diese erkennt. Dies sind die von den Formatierungsprogrammen bereitgestellten Basisschriftarten. Sie können dem Formatierungsprogramm jedoch mehr Schriftarten zur Verfügung stellen, indem Sie einige einfache Formatierungssprogramm-spezifische Schritte durchführen. Im Folgenden werden die Schritte für FOP beschrieben.

Allgemeines Verfahren zum Einrichten zusätzlicher Schriftarten in FOP

Um zusätzliche Schriftarten für FOP zur Verfügung zu stellen, müssen Sie folgende Schritte durchführen:

1. Generieren Sie die Schriftartbemaßungsdatei (Font Metrics-Datei) anhand der PostScript- und TrueType-Schriftartdateien. FOP bietet PFM Reader- und TTF Reader-Dienstprogramme zum Konvertieren der PostScript- bzw. TrueType-Schriftarten in eine XML-Schriftartbemaßungsdatei. Eine Anleitung dazu finden Sie auf der Seite FOP: Schriftarten.
2. Konfigurieren Sie die FOP-Konfigurationsdatei so, das die benötigten Schriftartbemaßungsdateien verwendet werden. Geben Sie dazu Informationen über die Schriftartdateien in eine FOP-Konfigurationsdatei ein. Siehe FOP: Schriftarten.
3. Bearbeiten Sie in der Datei `fop.bat` die letzte Zeile:

   ```
   "%JAVACMD%" […] org.apache.fop.cli.Main %FOP_CMD_LINE_ARGS%
   ```

   und fügen Sie den Pfad der Konfigurationsdatei ein:

   ```
   "%JAVACMD%" […] org.apache.fop.cli.Main %FOP_CMD_LINE_ARGS% -c conf\fop.xconf
   ```

Nachdem die Schriftartbemaßungsdateien bei FOP (in einer FOP-Konfigurationsdatei) registriert wurden und die ausführbare FOP-Datei so konfiguriert wurde, dass sie die Konfigurationsdatei liest, stehen die zusätzlichen Schriftarten bei der Erstellung von PDF-Dokumenten zur Verfügung.

Einrichten der FOP-Konfigurationsdatei

Die FOP-Konfigurationsdatei hat den Namen `fop.xconf` und befindet sich im FOP-Installationsordner im Ordner `conf`. Diese Datei, bei der es sich um ein XML-Dokument handelt, muss bearbeitet werden, damit FOP die Font Metrics-Dateien korrekt liest. Fügen Sie für jede Datei, die von FOP dargestellt werden soll, ein `font` Element an der vom `font`-Elementplatzhalter im Dokument angegebenen Stelle ein:

```
<font metrics-url="arial.xml" kerning="yes" embed-url="arial.ttf">
          <font-triplet name="Arial" style="normal" weight="normal"/>
          <font-triplet name="ArialMT" style="normal" weight="normal"/>
</font>
```

Im Beispiel oben ist

`arial.xml`	die URL der Bemaßungsdatei; am besten sollte ein absoluter Pfad verwendet werden.
`arial.ttf`	der Name der TTF-Datei (befindet sich normalerweise in `%WINDIR%\Fonts`).

© 2016-2022 Altova GmbH

`Arial`	definiert, dass die oben angeführte Bemaßungsdatei und die TTF-Datei verwendet werden sollen, wenn die Schriftartfamilie als `Arial` definiert ist.
`style="normal"`	legt fest, dass die oben angeführte Bemaßungsdatei und die TTF-Datei verwendet werden sollen, wenn der Schriftstil als `normal` (und nicht beispielsweise `kursiv`) definiert ist.
`weight="normal"`	legt fest, dass die oben angeführte Bemaßungsdatei und die TTF-Datei verwendet werden sollen, wenn die Schriftstärke als `normal` (und nicht beispielsweise `fett`) definiert ist.

Anmerkung zu Copyrights zu Schriftarten: Die Verwendung von Schriftarten unterliegt kopierrechtlichen Bestimmungen, die unterschiedliche Nutzungsbedingungen enthalten. Stellen Sie also sicher, dass die Lizenz, die Sie für diese Schriftart erworben haben, die Einbettung einer Schriftart - v.a. wenn es sich um die gesamte Schriftart handelt - gestattet.

Zeichensätze

Beachten Sie, dass sich die Zeichensätze von Schriftarten voneinander unterscheiden. Die Base 14 Fonts enthalten die ISO-8859-1 Zeichen plus die Glyphen in den Schriften Symbol und Zapf Dingbats. Wenn Ihr Dokument ein Zeichen enthält, das nicht in den Base 14 Fonts enthalten ist, müssen Sie eine Schriftart verwenden, die dieses Zeichen in seinem Zeichensatz enthält.

5.12 Diagramme

Wenn ein XML-Dokument in der Text- oder Grid-Ansicht geöffnet ist, kann im Diagrammfenster[130] (das eines der Ausgabefenster[115] ist) ein Diagramm (Kreisdiagramm, Balkendiagramm, usw.) generiert werden, in dem ausgewählte Daten aus dem XML-Dokument dargestellt werden. Das Diagramm kann anschließend als Bilddatei oder als XSLT- oder XQuery-Fragment in die Zwischenablage exportiert werden. Die Diagrammfunktion eignet sich dazu, ausgewählte numerische Daten aus einem XML-Dokument schnell grafisch darzustellen.

Die folgenden Diagrammtypen stehen zur Verfügung:

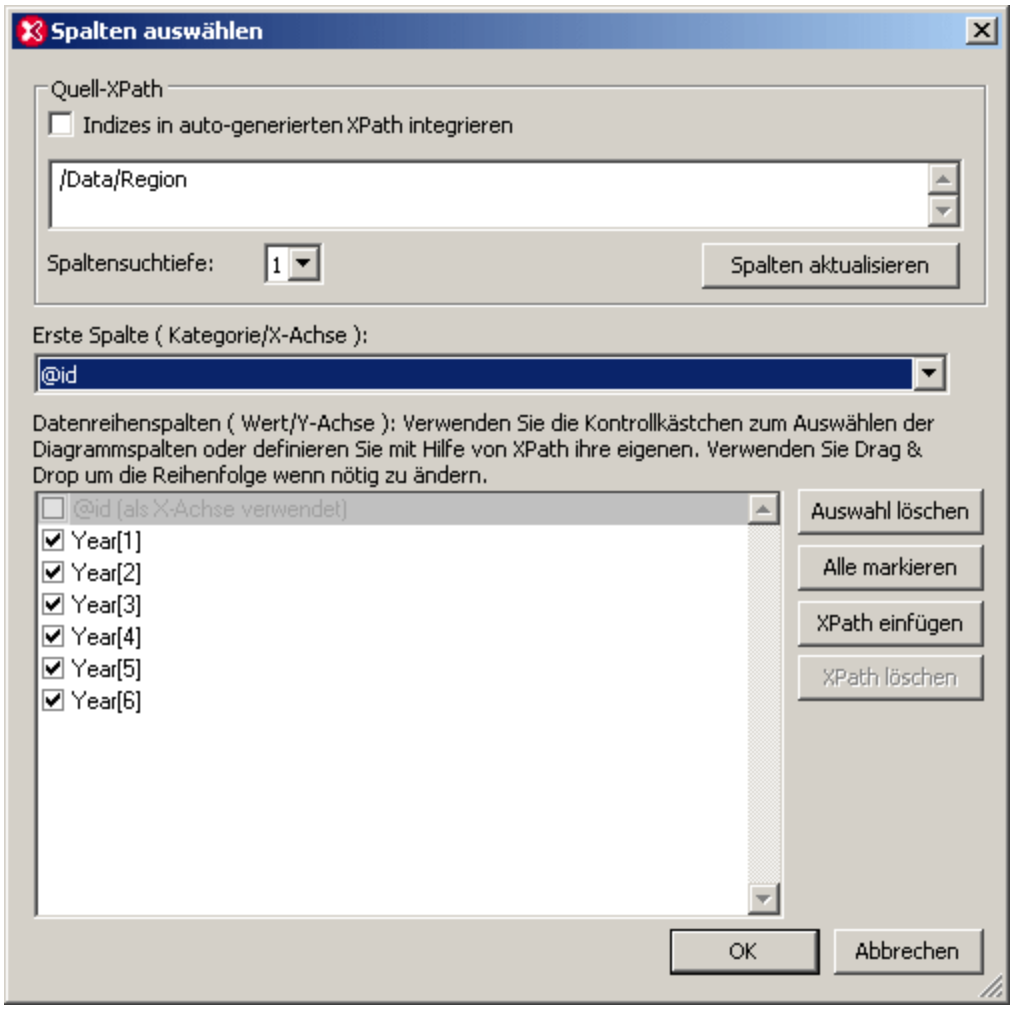
- Kreisdiagramme (2-D, 3-D)
- Balkendiagramme, Einzelbalken (2-D, 3-D)
- Balkendiagramme, gruppierte Balken (2-D, 3-D)
- Gestapelte Balkendiagramme
- Kategorie-Liniendiagramme
- Wert-Liniendiagramme
- Flächendiagramme und gestapelte Flächendiagramme
- Kerzendiagramme
- Einzelwertdiagramme (Tachometer und Balken)
- Overlay-Diagramme

Übersicht: von der Erstellung bis zum Export

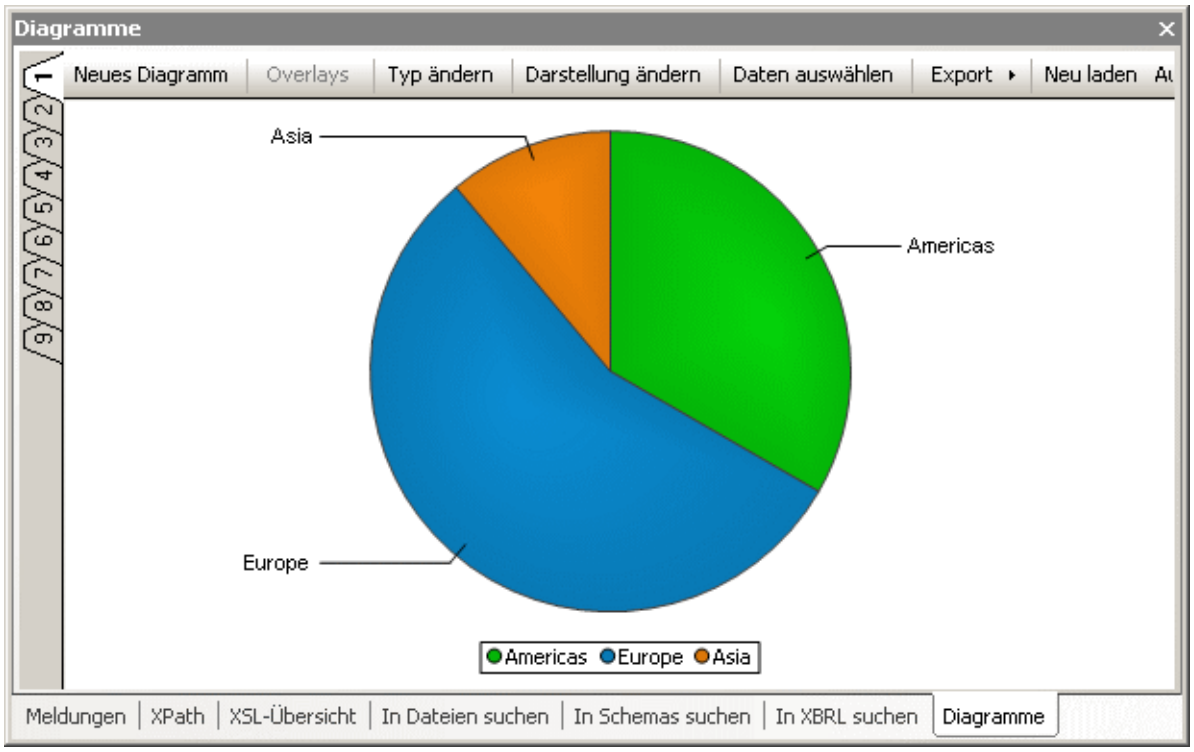
Im Folgenden sind die Schritte zur Erstellung eines Diagramms im Überblick beschrieben. Genauere Anleitungen dazu finden Sie in die Unterabschnitten dieses Abschnitts.

1. Wählen Sie in der Text- oder Grid-Ansicht den Node aus, der als Kontext-Node für die Datenauswahl verwendet werden soll. Sie können auch einen Node-Bereich auswählen. Die Auswirkungen der verschiedenen Auswahlmethoden sind im Abschnitt Quell-XPath[357] beschrieben.
2. Klicken Sie mit der rechten Maustaste darauf und wählen Sie im Kontextmenü den Befehl **Neues Diagramm** aus. Alternativ dazu können Sie im Diagrammausgabefenster auf die Schaltfläche "Neues Diagramm" klicken. Daraufhin wird das Dialogfeld "Spalten auswählen" (*Abbildung unten*) angezeigt, in dem die Daten für die X[361]- und die Y-Achse[366] ausgewählt werden und der Quell-XPath[357] geändert wird.

© 2016-2022 Altova GmbH



3. Wenn Sie auf **OK** klicken, wird das Diagramm im Diagrammfenster erstellt. (*siehe Abbildung unten*)



4. Die Auswahl der Diagrammdaten und andere Einstellungen können später bearbeitet werden. Es können nicht nur der Quell-XPath-Ausdruck zur Datenauswahl bzw. die Spaltenauswahl, sondern auch der Diagrammtyp und das Aussehen des Diagramms bearbeitet werden. Die Datenauswahl für die Diagrammachsen kann durch Klicken auf die Schaltfläche *Daten auswählen* bearbeitet werden. Um den Typ und das Aussehen des Diagramms zu ändern, klicken Sie auf die Schaltfläche **Typ ändern** bzw. **Darstellung ändern**.
5. Das Diagramm kann als Bilddatei oder als XSLT- oder XQuery-Fragment in die Zwischenablage exportiert werden.

Andere Funktionen

Die folgenden Funktionen erleichtern Ihnen die Arbeit mit Diagrammen:

- ***Mehrere Register:*** Wenn Sie ein neues Diagramm erstellen möchten, ohne das aktuelle zu löschen, dann erstellen Sie das neue Diagramm auf einem der anderen von eins bis neun durchnummerierten Register (*siehe Abbildung oben*). Beachten Sie: Selbst wenn ein XML-Dokument geschlossen wurde, bleiben die Diagramme, die anhand dieses Dokuments generiert wurden, auf den entsprechenden Registern im Diagrammfenster geöffnet.
- ***Automatisches Neuladen:*** Wenn die Schaltfläche **Auto** aktiv ist (*siehe Abbildung oben*), wird das Diagramm jedes Mal, wenn Daten im XML-Dokument geändert werden, automatisch neu geladen. Andernfalls muss das Diagramm manuell durch Klicken auf die Schaltfläche **Neu laden** aktualisiert werden.

Beispieldatei

In den Erklärungen zur Funktionsweise von Diagrammen in diesem Abschnitt und seinen Unterabschnitten wird auf eine XML-Datei mit dem Namen `YearlySales.xml` Bezug genommen. Diese Datei befindet sich im Ordner

© 2016-2022 Altova GmbH

```
C:\Dokumente und Einstellungen\<Benutzer>\Eigene
Dokumente\Altova\XMLSpy2023\Examples\Tutorial.
```

```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
       <Region id="Americas">
              <Year id="2005">30000</Year>
              <Year id="2006">90000</Year>
              <Year id="2007">120000</Year>
              <Year id="2008">180000</Year>
              <Year id="2009">140000</Year>
              <Year id="2010">100000</Year>
       </Region>
       <Region id="Europe">
              <Year id="2005">50000</Year>
              <Year id="2006">60000</Year>
              <Year id="2007">80000</Year>
              <Year id="2008">100000</Year>
              <Year id="2009">95000</Year>
              <Year id="2010">80000</Year>
       </Region>
       <Region id="Asia">
              <Year id="2005">10000</Year>
              <Year id="2006">25000</Year>
              <Year id="2007">70000</Year>
              <Year id="2008">110000</Year>
              <Year id="2009">125000</Year>
              <Year id="2010">150000</Year>
       </Region>
</Data>
```

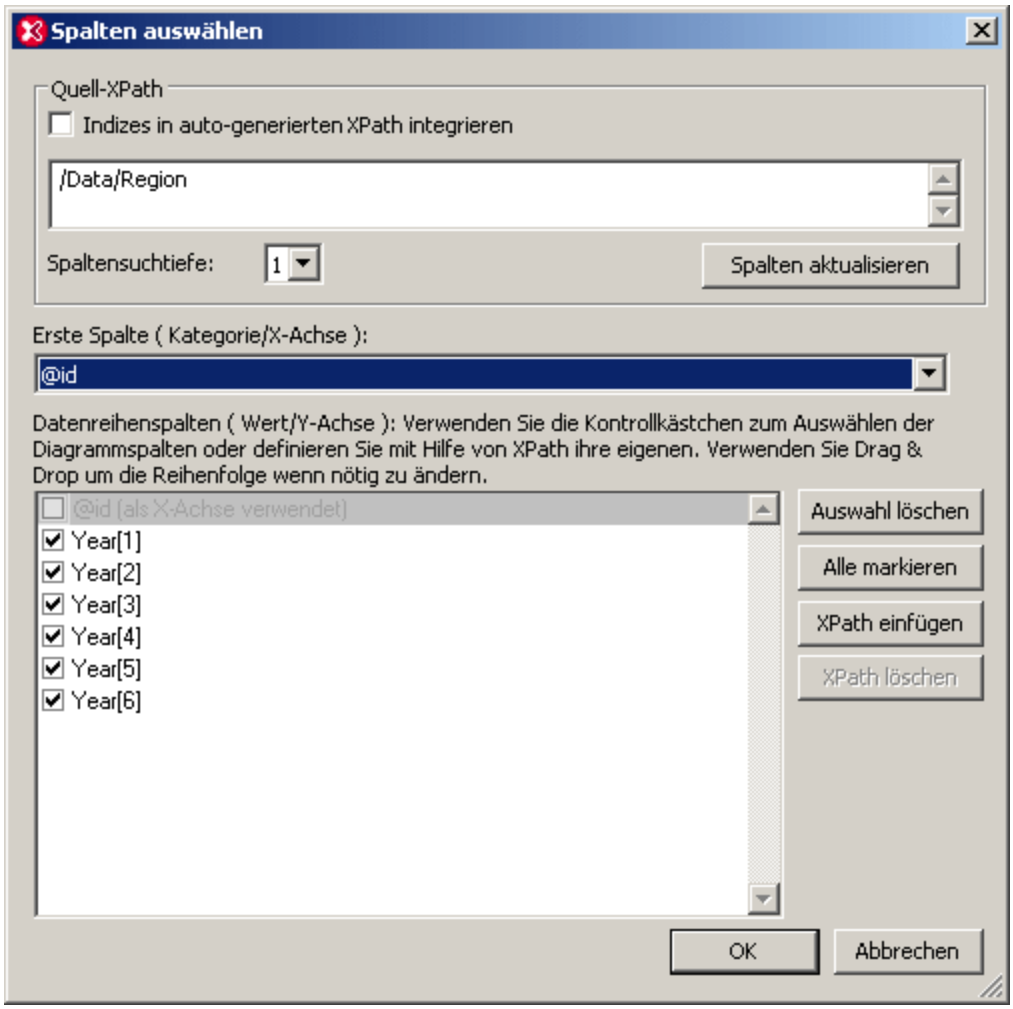
5.12.1 Erstellen eines Diagramms

Wenn Sie auf die Schaltfläche **Neues Diagramm** klicken, erscheint das Dialogfeld "Spalten auswählen" (*Abbildung unten*), in dem drei grundlegende Datenauswahlparameter für das Diagramm definiert werden. Anhand dieser Parameter wird die Diagrammdatentabelle erstellt:

- *Quell-XPath*: Ein XPath-Ausdruck wird automatisch bei Öffnen des Dialogfelds ausgefüllt. Er wählt den Node im XML-Dokument aus, der beim Aufrufen dieses Dialogfelds ausgewählt war. Sie können den XPath-Ausdruck im Dialogfeld durch Eingabe über die Tastatur bearbeiten. Das Kontrollkästchen *Indizes in auto-generiertes XPath integrieren* bestimmt, ob im XPath Prädikatfilter verwendet werden oder nicht (Details dazu siehe Quell-XPath 357). Der/Die untergeordneten Node(s) des durch den Quell-XPath ausgewählten Node stehen für die Auswahl der Datenspalten für die X- und Y-Achse zur Verfügung. Über die Auswahlliste "Spaltensuchtiefe" können Sie festlegen, wie vielen Ebenen unterhalb des Node nach Nodes durchsucht werden sollen, die für die Auswahl der Daten für die X- und Y-Achse verwendet werden können. Nachdem Sie den Quell-XPath bearbeitet haben, müssen Sie auf **Spalten aktualisieren** klicken, damit die Änderungen wirksam werden und damit die X- und Y-Achsenliste im Dialogfeld aktualisiert wird.
- *X-Achse*: In dieser Auswahlliste legen Sie fest, welcher Node auf der X-Achse abgebildet werden soll. Die Sequenz, die durch diese Auswahl zurückgegeben wird, liefert die Beschriftungen auf der X-Achse.

Mit der Auswahllistenoption *Automatisch enumeriert* erhalten Sie nummerierte Beschriftungen für die X-Achse. Beachten Sie, dass auch XPath-Ausdrücke, die für die Y-Achse erstellt wurden, in der Auswahlliste für die X-Achse zur Auswahl zur Verfügung stehen.

- *Y-Achse*: Die in diesem Fenster aktivierten Einträge werden als die Nodes verwendet, deren numerische Werte auf der numerischen Y-Achse abgebildet werden. Mit den Schaltflächen **Auswahl löschen** und **Alle markieren** wird die Auswahl für alle Einträge aufgehoben bzw. werden alle Einträge im Bereich "Y-Achse" ausgewählt. Über die Schaltfläche **XPath einfügen** kann eine Datenreihe generiert werden, die sonst nicht zur Verfügung steht, weil es sich dabei nicht um einen untergeordneten Node des Node handelt, der vom Quell-XPath zurückgegeben wird. Der Node bzw. der XPath-Ausdruck, der für die X-Achse ausgewählt ist, steht für die Y-Achse nicht zur Verfügung und ist ausgegraut.



So wird die Diagrammdatentabelle erstellt

Die für das Diagramm verwendeten Daten werden von der Auswahl, die im Dialogfeld "Spalten auswählen" getroffen wurde, bestimmt. Wir werden anhand eines Beispiels erklären, wie die Diagrammdaten ausgewählt werden. Da das XML-Dokument (*siehe Abbildung weiter unten*) drei `Region`-Elemente enthält, wählt der Quell-XPath `/Data/Region` diese der Reihe nach aus. Anschließend werden die Datenspalten generiert, wobei die einzelnen `Region`-Elemente den Kontext-Node bilden. Für jedes durch den Quell-XPath ausgewählte `Region` Element werden die folgenden Schritte durchgeführt:

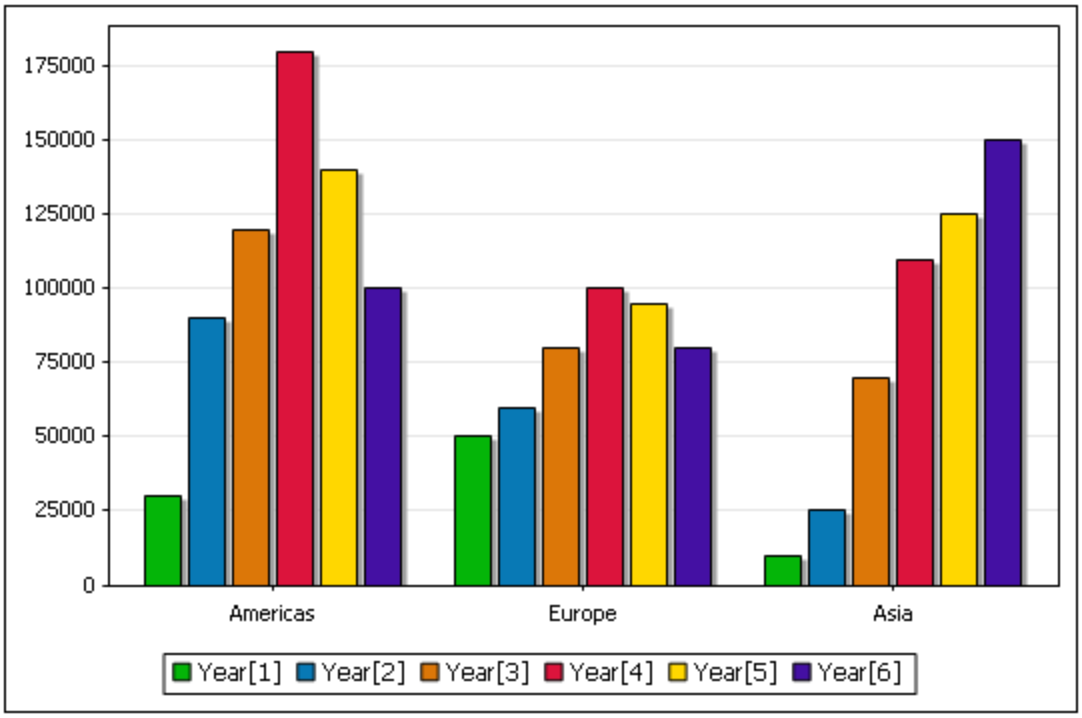
© 2016-2022 Altova GmbH

1. Der Ausdruck für die X-Achse generiert die erste Spalte (standardmäßig handelt es sich bei dieser Spalte um die Spalte, die für die Beschriftungen der X-Achse verwendet wird).
2. Für jede Datenreihe (Y-Achsenauswahl) wird eine Spalte generiert.

Die für das Dialogfeld "Spalten auswählen" generierten Diagrammdaten lassen sich, wie in der folgenden Tabelle dargestellt, visualisieren.

Source XPath	X-Axis	Y-Axis (Series columns)					
Region[1]	@id	Year[1]	Year[2]	Year[3]	Year[4]	Year[5]	Year[6]
Region[2]	@id	Year[1]	Year[2]	Year[3]	Year[4]	Year[5]	Year[6]
Region[3]	@id	Year[1]	Year[2]	Year[3]	Year[4]	Year[5]	Year[6]

Das anhand dieser Daten generierte Balkendiagramm würde in etwa folgendermaßen aussehen:



Beachten Sie bitte die folgenden wichtigen Punkte:

- Die Anzahl der Skalenstriche auf der X-Achse hängt von der Größe der Sequenz ab, die vom Quell-XPath zurückgegeben werden (in diesem Fall drei).
- Die vom Quell-XPath zurückgegebenen Nodes dienen als Kontext-Nodes für die Generierung der Datenreihen für die einzelnen Skalenstriche auf der X-Achse: (i) die Beschriftung der Skalenstriche auf der X-Achse (die Anhand der Auswahl für die X-Achse ermittelt wird) und (ii) alle Datenreihen, die für diesen Skalenstrich dargestellt werden (Diese Datenreihen werden über die Auswahl für die Y-Achse ausgewählt). Die XPath-Ausdrücke, die für die X- und die Y-Achse eingegeben wurden, werden im Kontext dieser (Quell-XPath)-Nodes ausgewertet.
- Die Sequenz, die von der Auswahl für die X-Achse zurückgegeben wird, bildet die Beschriftung für die einzelnen Skalenstriche. Gibt es weniger Beschriftungen als Skalenstriche, bleiben einige Skalenstriche unbeschriftet.

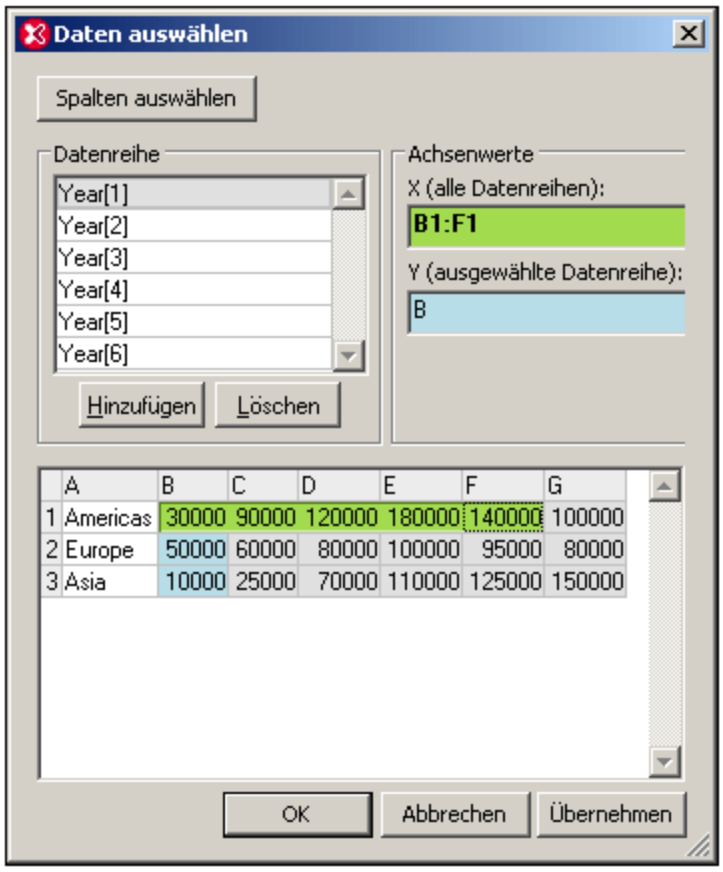
- Jede Datenreihe (für das Beispiel `Year[1]`) wird für jeden Kontext-Node einmal ausgewertet. Für einige Diagramme, wie z.B. Kreisdiagramme oder Einzelbalkendiagramme kann nur eine Datenreihe verwendet werden.
- Die Legenden werden anhand der Namen der Einträge der Datenreihe erstellt.

Hier sehen Sie zu Referenzzwecken das für das Beispiel verwendete XML-Dokument. Das Dokument ist unter dem Namen `YearlySales.xml` im Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\Tutorial` gespeichert.

```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
       <Region id="Americas">
              <Year id="2005">30000</Year>
              <Year id="2006">90000</Year>
              <Year id="2007">120000</Year>
              <Year id="2008">180000</Year>
              <Year id="2009">140000</Year>
              <Year id="2010">100000</Year>
       </Region>
       <Region id="Europe">
              <Year id="2005">50000</Year>
              <Year id="2006">60000</Year>
              <Year id="2007">80000</Year>
              <Year id="2008">100000</Year>
              <Year id="2009">95000</Year>
              <Year id="2010">80000</Year>
       </Region>
       <Region id="Asia">
              <Year id="2005">10000</Year>
              <Year id="2006">25000</Year>
              <Year id="2007">70000</Year>
              <Year id="2008">110000</Year>
              <Year id="2009">125000</Year>
              <Year id="2010">150000</Year>
       </Region>
</Data>
```

In der Tabelle des Dialogfelds "Daten auswählen" (*Abbildung unten*) sehen Sie die im Dialogfeld "Spalten auswählen" ausgewählten Daten. Das Dialogfeld "Daten auswählen" wird durch Klicken auf die Schaltfläche **Daten auswählen** im Diagrammausgabefenster aufgerufen.

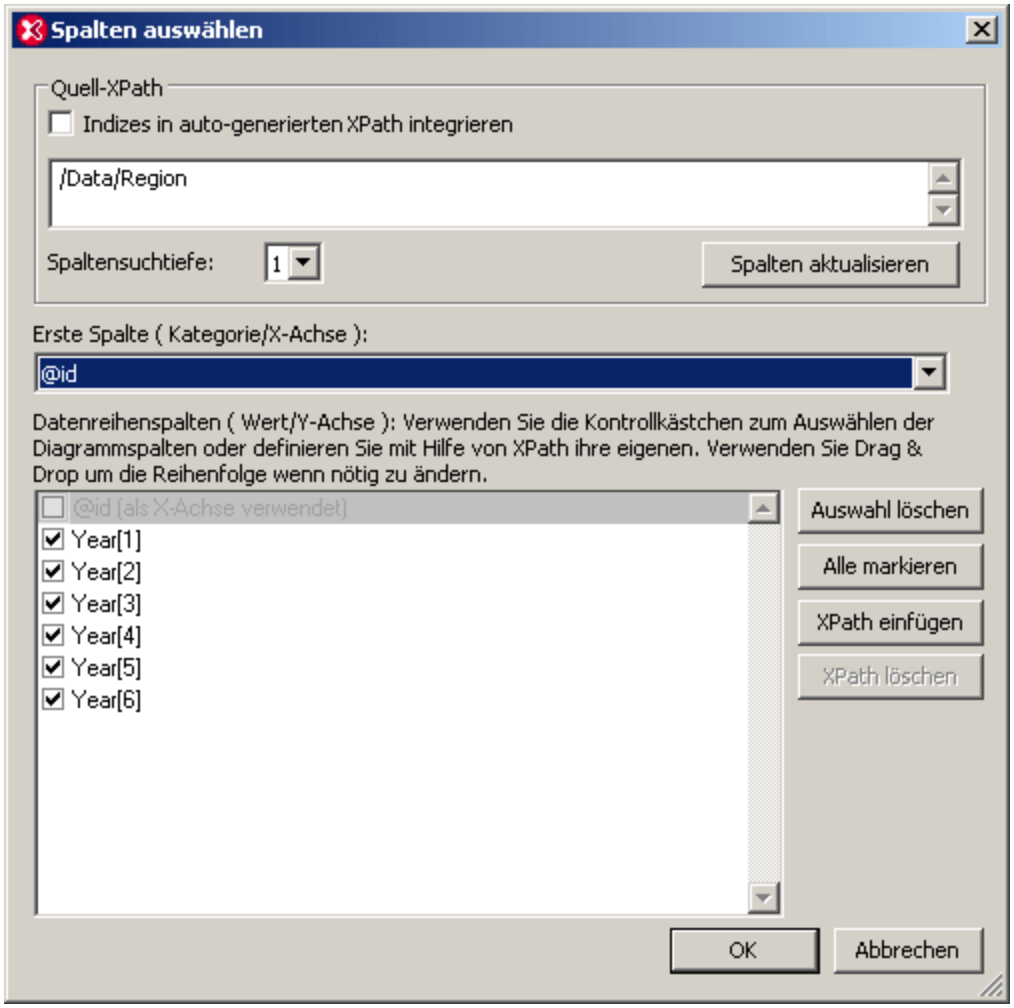
© 2016-2022 Altova GmbH



Nähere Informationen zu den einzelnen Parametern des Dialogfelds "Spalten auswählen" finden Sie in den einzelnen Abschnitten: Quell-XPath[357], Auswahl für die X-Achse[361], Auswahl für die Y-Achse[366] und Diagrammdaten[370].

5.12.2 Quell-XPath

Der Quell-XPath wird im Dialogfeld "Spalten auswählen" definiert. Damit wird festgelegt, welche Nodes im Dokument für die Auswahl der Daten für die X- und die Y-Achse zur Verfügung stehen. Über die Auswahlliste "Spaltensuchtiefe" wird definiert, wie viele untergeordnete Ebenen nach Nodes durchsucht werden, die für die Daten der X- und Y-Achse verwendet werden können.



Nachdem Sie den Quell-XPath ausgewählt haben, müssen Sie auf **Spalten aktualisieren** klicken, damit die Änderungen wirksam werden und damit die Auswahl der Daten für die X- und Y-Achse in Dialogfeld aktualisiert wird. Wenn der XPath-Ausdruck auch Prädikate enthält (so wird z.B. in `/Data/Region[1]` mit Hilfe des Prädikats `[1]` das erste `Region` Element ausgewählt), so muss das Kontrollkästchen "Indizes...inkludieren" aktiviert werden. Wenn Sie den Quell-XPath-Ausdruck ändern, denken Sie immer daran auf **Spalten aktualisieren** zu klicken.

Quell-XPath von der Cursorposition aus

Anhand des unten gezeigten XML-Dokuments wird erläutert, welcher Quell-XPath beim Öffnen des Dialogfelds "Spalten auswählen" ausgewählt wird. Hier sehen Sie zu Referenzzwecken das für das Beispiel verwendete XML-Dokument. Das Dokument ist unter dem Namen `YearlySales.xml` im Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\Tutorial` gespeichert.

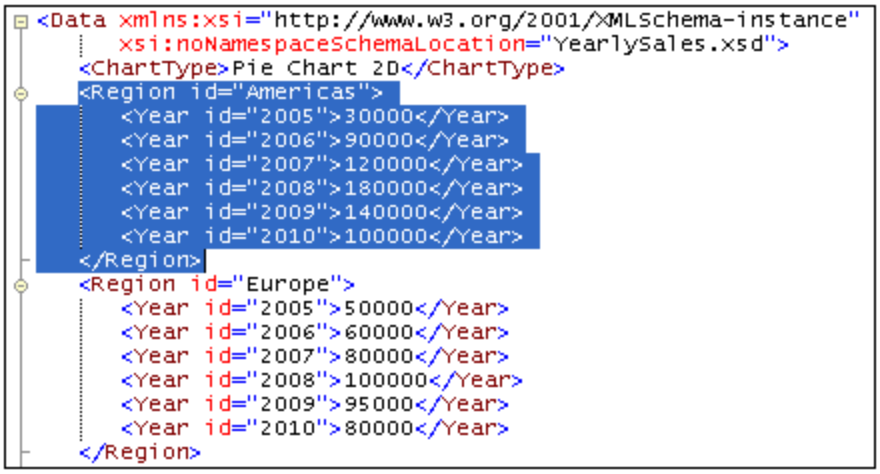
```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
       <Region id="Americas">
              <Year id="2005">30000</Year>
              <Year id="2006">90000</Year>
```


© 2016-2022 Altova GmbH

```
            <Year id="2007">120000</Year>
            <Year id="2008">180000</Year>
            <Year id="2009">140000</Year>
            <Year id="2010">100000</Year>
      </Region>
      <Region id="Europe">
            <Year id="2005">50000</Year>
            <Year id="2006">60000</Year>
            <Year id="2007">80000</Year>
            <Year id="2008">100000</Year>
            <Year id="2009">95000</Year>
            <Year id="2010">80000</Year>
      </Region>
      <Region id="Asia">
            <Year id="2005">10000</Year>
            <Year id="2006">25000</Year>
            <Year id="2007">70000</Year>
            <Year id="2008">110000</Year>
            <Year id="2009">125000</Year>
            <Year id="2010">150000</Year>
      </Region>
</Data>
```

Die folgende Fälle sind möglich:

- Wenn der Cursor an eine beliebige Stelle zwischen den Start-Tag (auch in einen Attributwert) und den End-Tag des Elements `Data` oder an eine beliebige Stelle innerhalb des Elements `Data` aber nicht in einen untergeordneten Node platziert wird, so lautet der Quell-XPath: `/Data`
- Wenn der Cursor an eine beliebige Stelle zwischen den Start-Tag (auch in einen Attributwert) und den End-Tag des Elements `Region` oder an eine beliebige Stelle innerhalb des Elements `Region` aber nicht in einen untergeordneten Node davon platziert wird, so lautet der Quell-XPath: `/Data/Region`
- Wenn der Cursor an eine beliebige Stelle zwischen den Start-Tag (auch in einen Attributwert) und den End-Tag des Elements `Year` oder an eine beliebige Stelle innerhalb des Elements `Year` platziert wird, so lautet der Quell-XPath: `/Data/Region[N]/Year`. Der Prädikatfilter `[N]` wählt das entsprechende `Region` Element aus, das das ausgewählte `Year` Element enthält. Daher werden die für die X- und die Y-Achse ausgewählten Daten auf `Year` Elemente dieses bestimmten `Region` Elements beschränkt.
- Wenn Sie nur ein `Region` Element auswählen möchten, beispielsweise `/Data/Region[1]`, dann markieren Sie dieses Element. In diesem Fall ist dies, wie in der Abbildung unten gezeigt, das erste `Region` Element.



- Wenn Sie zwei `Region` Elemente markieren, wird ein XPath-Ausdruck generiert, der nur diese beiden `Region` Elemente auswählt.

Der Quell-XPath-Ausdruck kann anschließend im Dialogfeld "Spalten auswählen" bearbeitet werden.

In der Grid-Ansicht erfolgt die Auswahl durch Klicken auf einen Node oder Markieren eines Bereichs. Der Quell-XPath, der anhand dieser Auswahl generiert wird, entspricht dem oben für die Textansicht beschriebenen.

Indizes in XPath integrieren

Das Kontrollkästchen *Indizes in auto-generiertes XPath integrieren* bestimmt, ob in den XPath-Ausdrücken Prädikatfilter verwendet werden, ob diese Prädikatfilter automatisch während des Aufrufs des Dialogfelds "Spalten auswählen" eingegeben werden oder ob sie manuell eingegeben werden. wenn, zum Beispiel, der Cursor in ein untergeordnetes Element des ersten Elements `Region` des Dokuments gesetzt wird und das Kontrollkästchen *Indizes in auto-generiertes XPath integrieren* aktiviert ist, lautet der automatisch eingegebene XPath-Ausdruck, zum Beispiel: `/Data/Region[1]/Year`. Ist das Kontrollkästchen *Indizes in auto-generiertes XPath integrieren* nicht aktiviert, so lautet der Ausdruck: `/Data/Region/Year`.

Das Kontrollkästchen *Indizes in auto-generiertes XPath integrieren* bestimmt ferner ob manuell eingegebene Prädikate beibehalten werden. Wenn Sie Prädikate im Quell-XPath Ausdruck verwenden möchten, müssen Sie das Kontrollkästchen *Indizes in auto-generiertes XPath integrieren* daher aktivieren.

Auswirkungen der Quell-XPath-Auswahl

Beachten Sie die folgenden Auswirkungen der Quell-XPath-Auswahl.

- Die Anzahl der Elemente in der vom Quell-XPath zurückgegebenen Sequenz bestimmt die Anzahl der Skalenstriche auf der X-Achse. Die Anzahl der Skalenstriche auf der X-Achse kann außer durch Ändern des Quell-XPath-Ausdrucks nur auf eine Art geändert werden, nämlich durch Auswahl einer Anzahl für Beschriftungen für eine Datenreihe, wobei deren Anzahl die Anzahl der Skalenstriche übersteigt. Nähere Informationen zu diesem Szenario finden Sie im Abschnitt Auswahl für die X-Achse 361.
- Der Quell-XPath Node ist der übergeordnete Node für alle Nodes, die für die Auswahl der Daten für die X- und Y-Achse zur Verfügung stehen und für alle XPath-Ausdrücke, die Sie eingeben.

 © 2016-2022 Altova GmbH

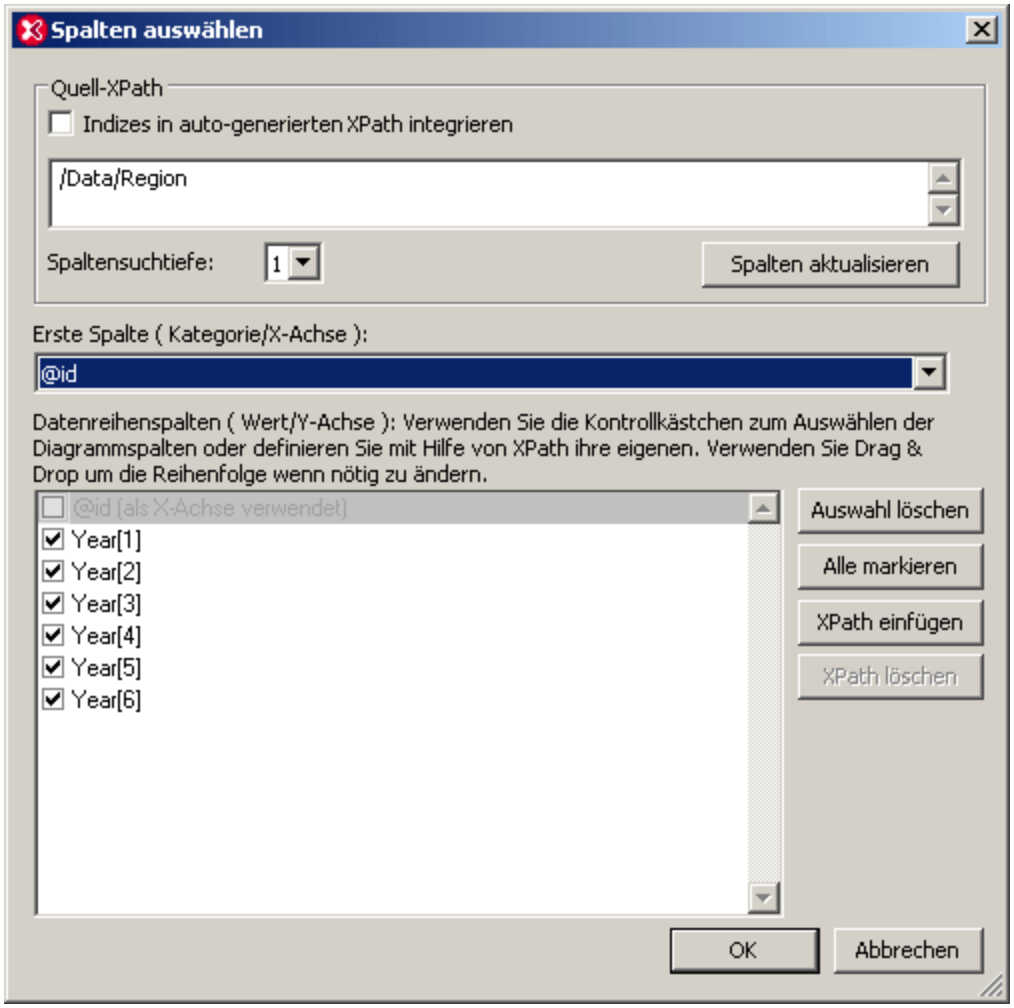
- Aufgrund der oben genannten Tatsache, wirkt sich jede Änderung am Quell-XPath-Ausdruck nicht nur auf die Anzahl der Skalenstriche auf der X-Achse sondern auch auf den Kontext für alle XPath-Ausdrücke, die mit dem Diagramm im Zusammenhang stehen, aus.

Hier sehen Sie z.B., wie sich einige XPath-Ausdrücke in Bezug auf das oben gezeigte XML-Dokument auswirken.

- /Data/Region: Gibt die drei `Region` Elemente zurück, d.h. drei Skalenstriche auf der X-Achse. Jedes `Region` Element bildet wiederum den Kontext-Node für XPath-Ausdrücke.
- /Data/Region/Year: Gibt 18 `Year` Elemente zurück, d.h. 18 Skalenstriche auf der X-Achse. Jedes `Year` Element bildet wiederum den Kontext-Node für XPath-Ausdrücke.
- /Data/Region[1]/Year: Gibt die sechs `Year` Element Children des ersten `Region` Elements zurück, d.h. sechs Skalenstriche auf der X-Achse. Jedes `Year` Element des ersten `Region` Elements liefert wiederum den Kontext-Node für XPath-Ausdrücke.
- distinct-values(//Year/@id): Gibt sechs Elemente zurück (die einzelnen Werte des `Year/@id` Attributs: `2005`, `2006`, `2007`, `2008`, `2009`, `2010`). Da dieser XPath-Ausdruck aber kein Node-Element zurückgibt, kann er nicht als Kontext-Node für einen XPath-Ausdruck verwendet werden. Wenn die Elemente in dieser Sequenz für Nodes im XML-Dokument verwendet werden (z.B. mit der Funktion `current()` oder (abgekürzt: .)), dann muss der XPath-Ausdruck, der das aktuelle Element verwendet an der Dokument-Root beginnen, damit der Kontext verwendet werden kann. Beispiel: `/Data/Region[1]/Year[@id eq .]`.

5.12.3 Auswahl für die X-Achse

Die Datenauswahl für die X-Achse wird im Dialogfeld "Spalten auswählen" (*Abbildung unten*) definiert. Damit werden die Beschriftungen festgelegt, die auf der X-Achse angezeigt werden. Die Beschriftungen können später im Dialogfeld "Daten auswählen" (*siehe unten*) bearbeitet werden.



Nehmen wir als Beispiel das folgende XML-Dokument. (Es hat den Namen `YearlySales.xml` und befindet sich im Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\Tutorial`). Der Cursor wird in den Start-Tag des ersten `Region` Elements platziert und man klickt auf die Schaltfläche "Neues Diagramm" des Diagrammausgabefensters. Daraufhin wird das Dialogfeld "Spalten auswählen" angezeigt, in dem der folgende Quell-XPath zu sehen ist: `/Data/Region` (*siehe Abbildung oben*).

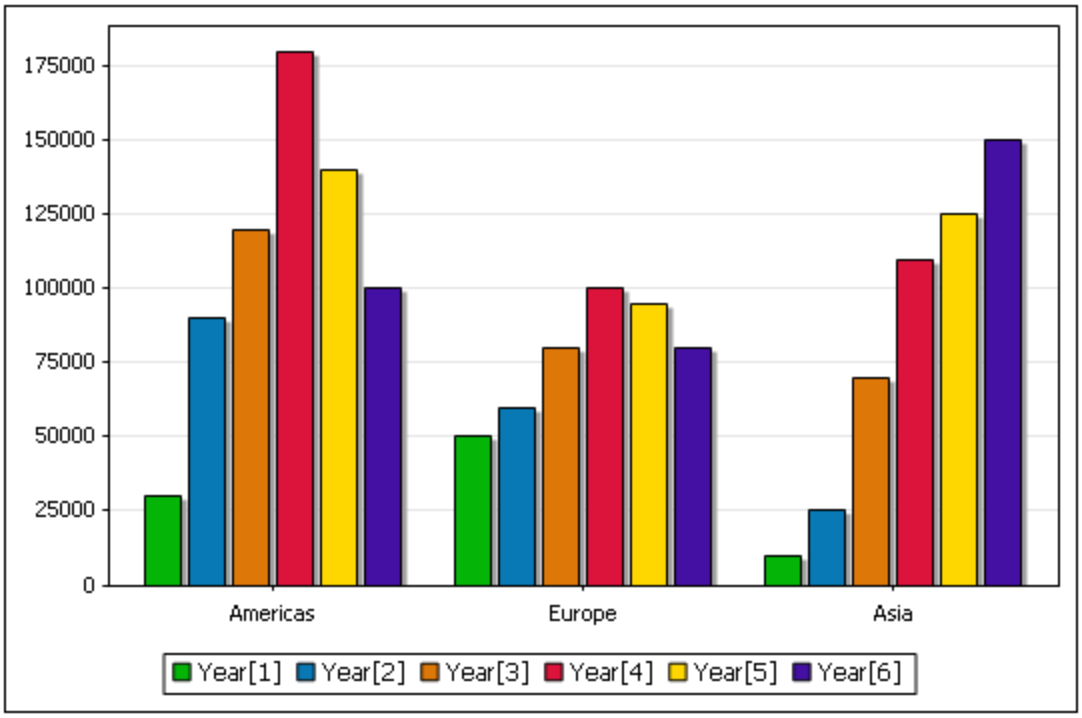
```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
       <Region id="Americas">
              <Year id="2005">30000</Year>
              <Year id="2006">90000</Year>
              <Year id="2007">120000</Year>
              <Year id="2008">180000</Year>
              <Year id="2009">140000</Year>
              <Year id="2010">100000</Year>
       </Region>
       <Region id="Europe">
              <Year id="2005">50000</Year>
```


© 2016-2022 Altova GmbH

```
            <Year id="2006">60000</Year>
            <Year id="2007">80000</Year>
            <Year id="2008">100000</Year>
            <Year id="2009">95000</Year>
            <Year id="2010">80000</Year>
        </Region>
        <Region id="Asia">
            <Year id="2005">10000</Year>
            <Year id="2006">25000</Year>
            <Year id="2007">70000</Year>
            <Year id="2008">110000</Year>
            <Year id="2009">125000</Year>
            <Year id="2010">150000</Year>
        </Region>
</Data>
```

Wie im Abschnitt Quell-XPath[357] erläutert, wird mit diesem Quell-XPath-Ausdruck ein Diagramm eingerichtet, das drei Skalenstriche auf der X-Achse hat (da der Quell-XPath drei Elemente zurückgibt, nämlich die drei `Region` Elemente). Da als Beschriftungen für die drei Skalenstriche im Diagramm die Namen der drei Regionen zu sehen sein sollen, wählen wir in der Auswahlliste der Datenauswahl für die X-Achse das Attribut `@id` aus (*siehe Abbildung Dialogfeld "Spalten auswählen" weiter oben*).

Um die Diagrammdaten für die einzelnen Skalenstriche zu erstellen, werden die einzelnen `Region` Elemente der Reihe nach ausgewertet. Für jedes `Region` Element wird anhand des Attributs `id` die richtige Beschriftung für den Skalenstrich auf der X-Achse generiert. Die X-Achse würde in etwa so wie in der Abbildung unten aussehen.

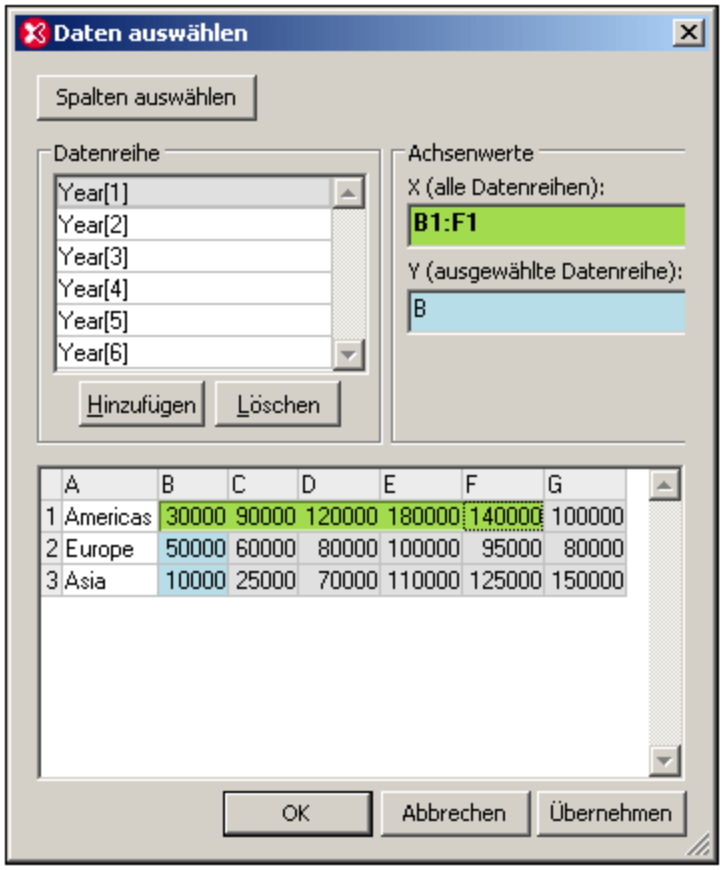


Wenn in der Auswahlliste für die X-Achse ein anderer XPath-Ausdruck ausgewählt wird, so wird dieser Ausdruck innerhalb des entsprechenden `Region` Elementkontexts ausgewertet und das Ergebnis wird als Beschriftung für den entsprechenden Skalenstrich angezeigt. Mit der Option *Automatisch enumeriert* wird eine Zahlensequenz generiert, die der Anzahl der Skalenstriche entspricht: Der erste Skalenstrich erhält die Nummer 1, der zweite 2, usw.

Ändern der Beschriftungen und der Anzahl der Skalenstriche für die X-Achse

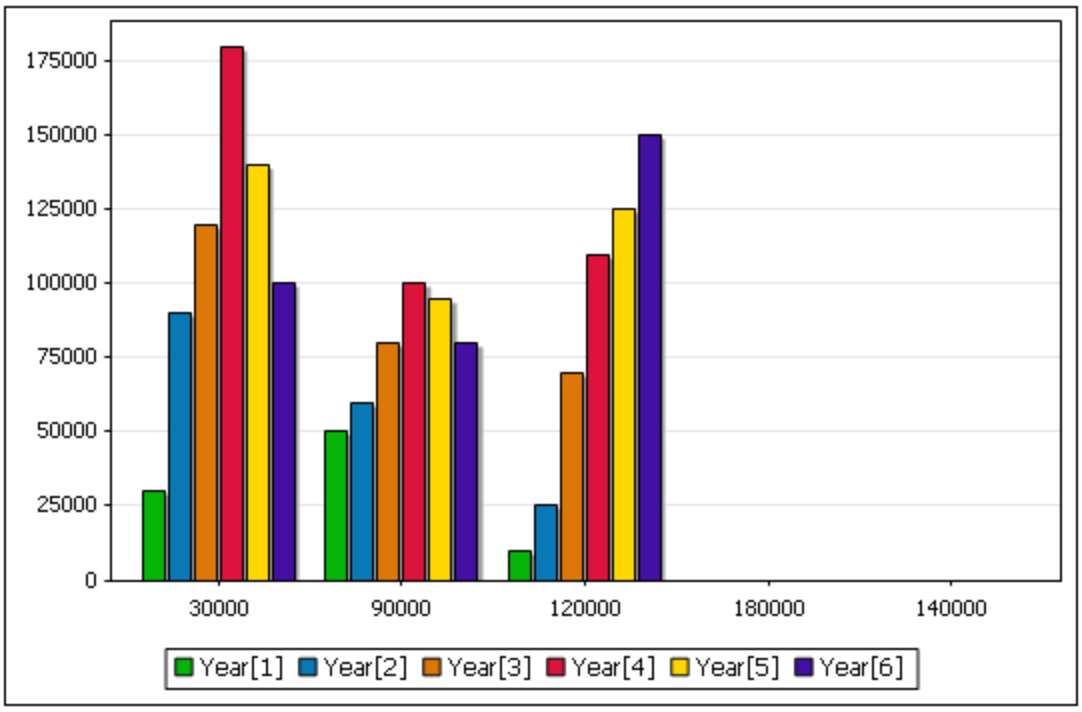
Die Auswahl der Beschriftungen für die X-Achse kann im Dialogfeld "Daten auswählen", das durch Klicken auf die Schaltfläche **Daten auswählen** des Diagrammausgabefensters, aufgerufen wird, geändert werden.

Klicken Sie z.B. im Dialogfeld "Daten auswählen", das Sie in der Abbildung unten sehen, im Bereich "Achsenwerte" auf das Textfeld für die X-Achse, damit Sie die Auswahl für die X-Achse ändern können. Klicken Sie nun auf das Feld B1 und ziehen Sie die Maus bis zu F1, um die Auswahl zu definieren. Klicken Sie auf **OK**, um das neue Diagramm anzuzeigen.



Mit dieser Auswahl werden die Beschriftungen für die Skalenstriche definiert (*siehe Abbildung unten*). Beachten Sie außerdem: Da die neue Auswahl fünf Elemente umfasst, wurden fünf Skalenstriche generiert. Allerdings sind nur die ersten drei mit Daten befüllt, da der Quell-XPath drei Nodes zurückgibt und dies die Nodes sind, die für die Diagramme verarbeitet werden. Diese drei Nodes entsprechen den Zeilen der Tabelle[354], die im Dialogfeld "Daten auswählen" angezeigt werden. Beachten Sie, dass die Anzahl der Zeilen in der Tabelle nur durch Änderung des Quell-XPath-Ausdrucks geändert werden kann.

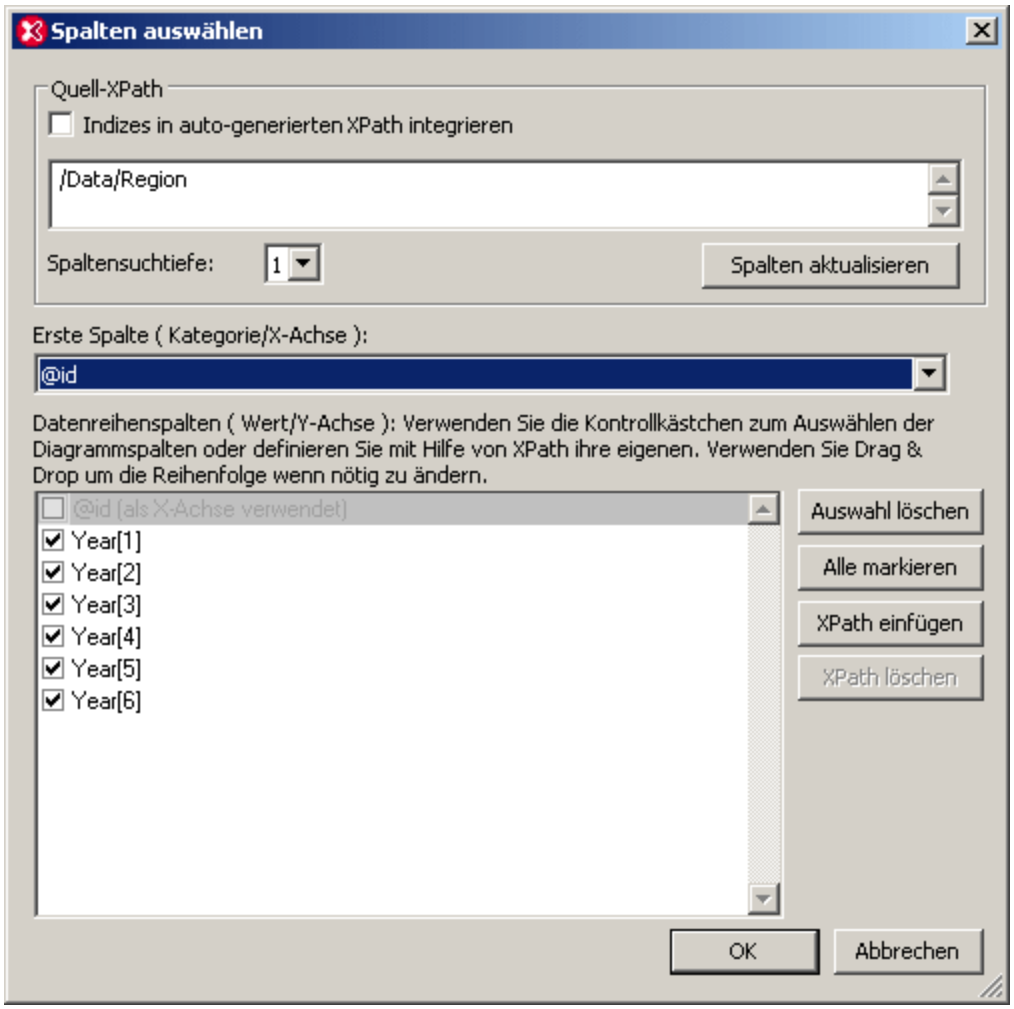
© 2016-2022 Altova GmbH



Nähere Informationen dazu, wie sich der Quell-XPath auf die Auswahl für die X-Achse auswirkt, finden Sie unter Vorgangsweise bei der Erstellung von Diagrammdaten[354].

5.12.4 Auswahl für die Y-Achse

Die Y-Achse (*siehe Abbildung unten*) wird auch als Datenreihenachse bezeichnet.

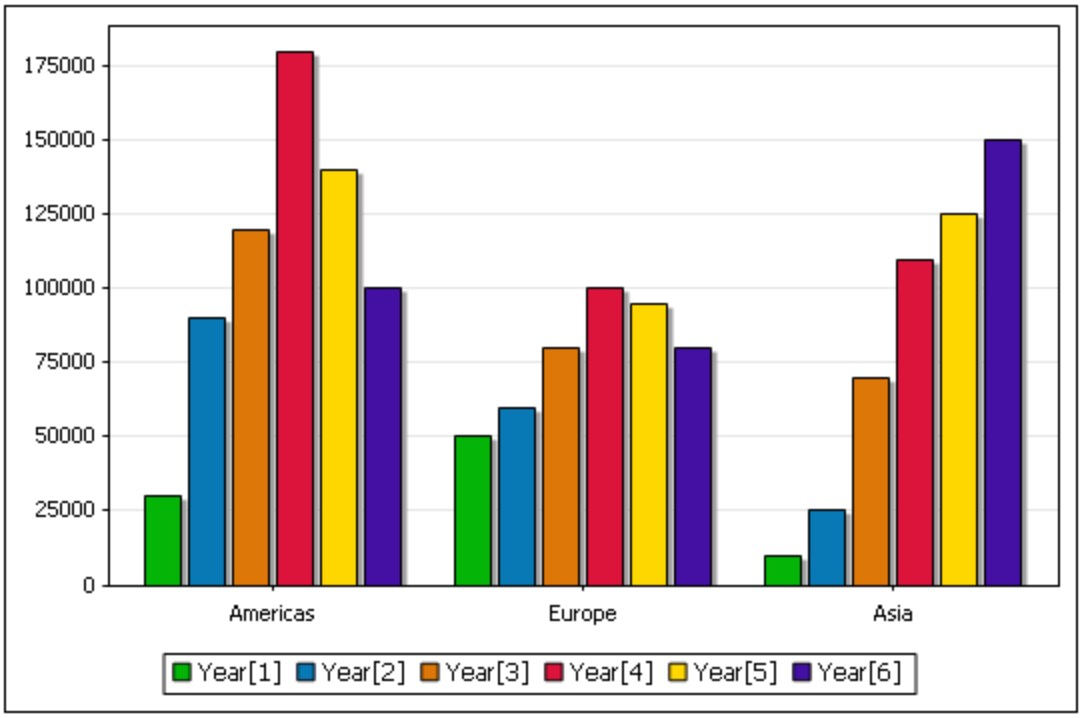


Die Datenauswahl für diese Achse legt fest, wie viele Datenreihen für jeden Skalenstrich auf der X-Achse dargestellt werden. Wenn nur eine Datenreihe ausgewählt ist, wird für jeden Skalenstrich der X-Achse der Wert, der vom XPath-Ausdruck für diese Datenreihe zurückgegeben wird, dargestellt. Wurden, wie in der Abbildung unten, in der sechs Datenreihen ausgewählt wurden, mehrere Datenreihen ausgewählt, so sieht die Diagrammdatenauswahl aus, wie in der unten gezeigten Tabelle. (Der Kontext-Node[354] für die Datenauswahl für die Y-Achse ist das entsprechende `Region` Element.)

Source XPath	X-Axis	Y-Axis (Series columns)					
`Region[1]`	`@id`	`Year[1]`	`Year[2]`	`Year[3]`	`Year[4]`	`Year[5]`	`Year[6]`
`Region[2]`	`@id`	`Year[1]`	`Year[2]`	`Year[3]`	`Year[4]`	`Year[5]`	`Year[6]`
`Region[3]`	`@id`	`Year[1]`	`Year[2]`	`Year[3]`	`Year[4]`	`Year[5]`	`Year[6]`

Das erzeugte Diagramm sieht in etwa folgendermaßen aus:

© 2016-2022 Altova GmbH



Die X-Achse hat drei Skalenstriche, die mit den Werten der entsprechenden `Region/@id` Attribute beschriftet sind. An jedem Skalenstrich der X-Achse wird der XPath-Ausdruck für die einzelnen Datenreihen ausgewertet. In unserem Beispiel wird für jeden Skalenstrich der X-Achse jede der sechs Datenreihen ausgewertet und dargestellt. So wird z.B. die erste Datenreihe (`Year[1]`) für alle drei Regionen dargestellt, dasselbe geschieht für `Year[2]` bis `Year[6]`.

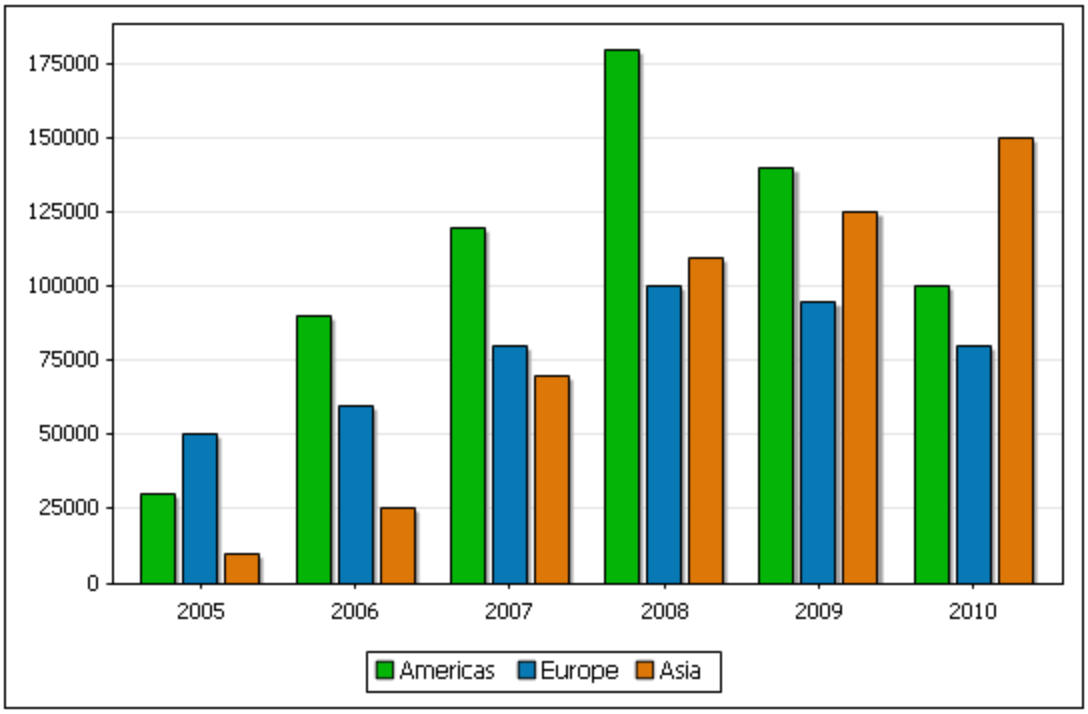
Anmerkung: Einige Diagramme, wie z.B. Kreisdiagramme und Einzelbalkendiagramme haben nur eine Achse. So hat z.B. in einem Einzelbalkendiagramm jeder Skalenstrich der X-Achse nur einen einzigen Balken, der für eine einzige Datenreihe steht. In einem **Kreisdiagramm** machen die Werte der einzigen Datenreihe 100 % des Kreises aus, wobei jeder Wert einem Skalenstrich der X-Achse zugewiesen ist.

Legenden zur Y-Achse

Die Legenden, die unterhalb vom Diagramm angezeigt werden, sind die Namen der Datenreihen. Diese Namen können im Dialogfeld "Daten auswählen" 370 geändert werden.

Tauschen der Auswahl für die X-Achse mit der für die Y-Achse

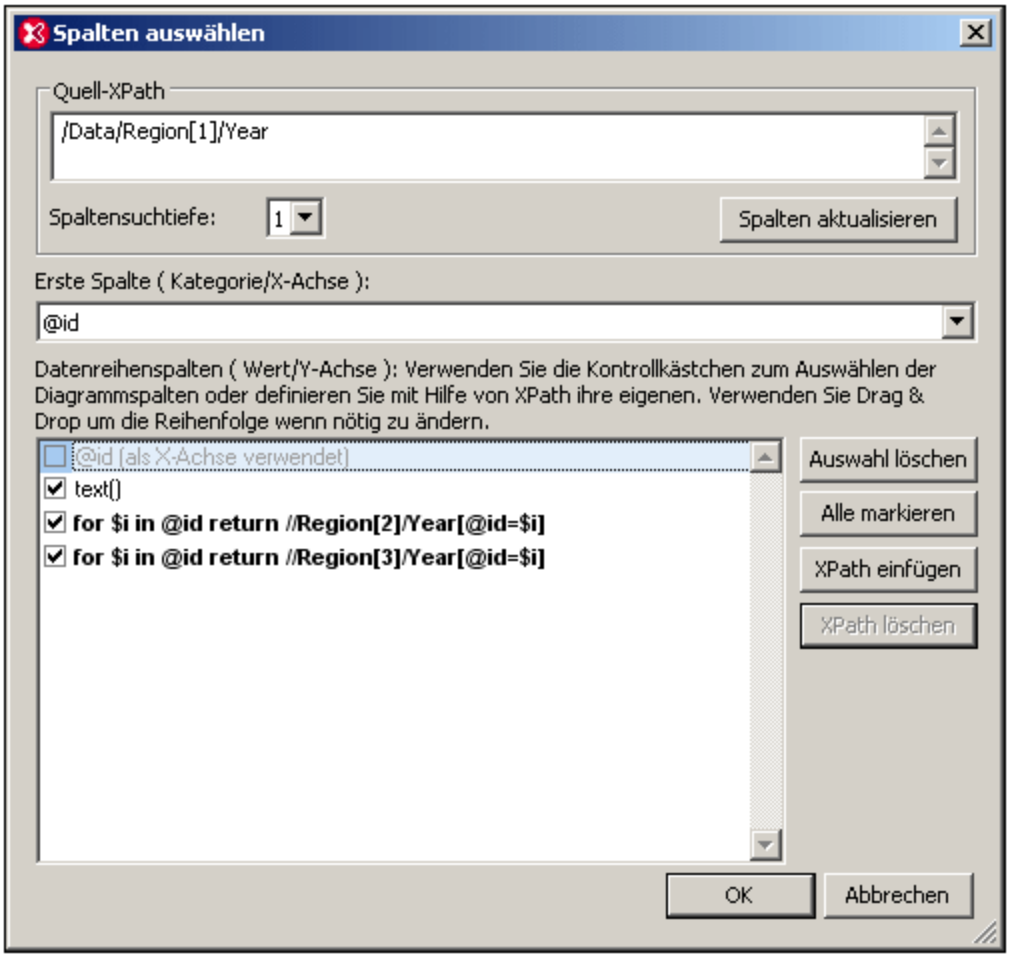
Im Beispiel oben werden die Regionen auf der X-Achse und die Jahresumsätze für die einzelnen Regionen auf der Y-Achse dargestellt; Die `Year` Elemente bilden die Datenreihe. Was wäre aber nun, wenn wir die Jahre auf der X-Achse darstellen möchten und die regionalen Jahresumsätze wie im Diagramm unten miteinander vergleichen möchten? Wir würden sechs Skalenstriche auf der X-Achse benötigen (die über die Quell-XPath-Auswahl ermittelt werden), um diese Skalenstriche anschließend mit den entsprechenden Jahren zu beschriften und schließlich drei Datenreihen (für die Regionen) auszuwählen, von denen jeweils alle für die einzelnen Skalenstriche der X-Achse dargestellt werden. In der Abbildung unten aus dem Dialogfeld "Spalten auswählen" sehen Sie wie diese Datenauswahl vorgenommen werden könnte.



Auswahl der Datenreihe

Um einen Node als Datenreihe für ein Diagramm zu definieren, aktivieren Sie das Kästchen neben diesem Node. Sie können den Quell-XPath-Ausdruck und die Spaltensuchtiefe im Bereich "Datenreihen" ändern. Alternativ dazu können Sie, wie in der Abbildung unten gezeigt, auch einen XPath-Ausdruck hinzufügen, um einen Node auszuwählen. Eine Beschreibung zu diesem Szenario finden Sie unter Diagrammbeispiel: Komplex 406.

© 2016-2022 Altova GmbH



Referenzdokument

Zu Refenzzwecken sehen Sie hier das XML-Dokument, das für das Beispiel in diesem Abschnitt verwendet wurde. Es hat den Namen `YearlySales.xml` und befindet sich im Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\Tutorial`.

```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
       <Region id="Americas">
              <Year id="2005">30000</Year>
              <Year id="2006">90000</Year>
              <Year id="2007">120000</Year>
              <Year id="2008">180000</Year>
              <Year id="2009">140000</Year>
              <Year id="2010">100000</Year>
       </Region>
       <Region id="Europe">
              <Year id="2005">50000</Year>
              <Year id="2006">60000</Year>
```

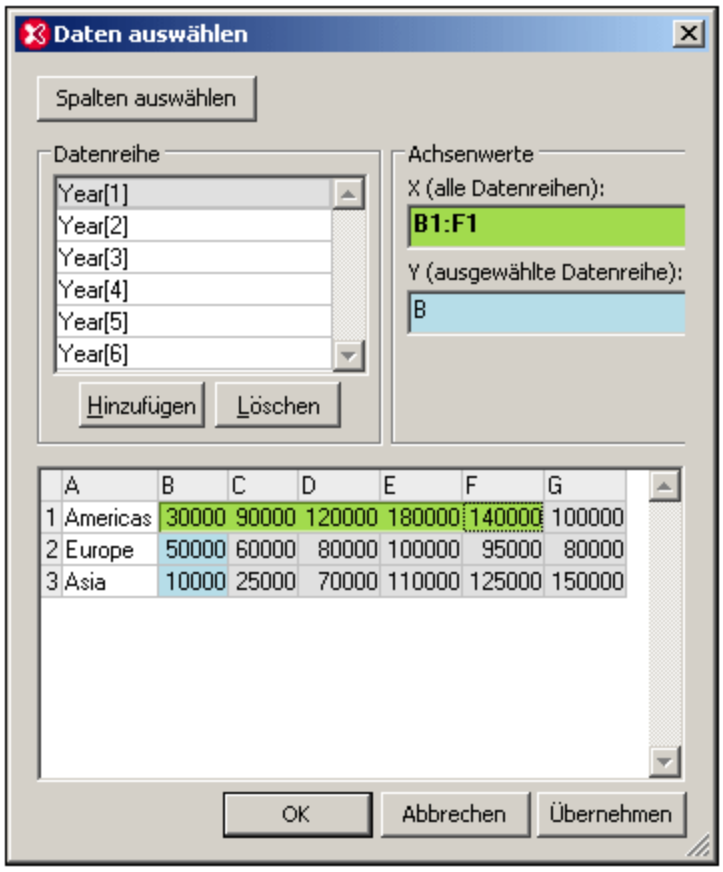
```
            <Year id="2007">80000</Year>
            <Year id="2008">100000</Year>
            <Year id="2009">95000</Year>
            <Year id="2010">80000</Year>
      </Region>
      <Region id="Asia">
            <Year id="2005">10000</Year>
            <Year id="2006">25000</Year>
            <Year id="2007">70000</Year>
            <Year id="2008">110000</Year>
            <Year id="2009">125000</Year>
            <Year id="2010">150000</Year>
      </Region>
</Data>
```

5.12.5 Diagrammdaten

Wenn Sie auf die Schaltfläche **Daten auswählen** klicken, wird das Dialogfeld "Daten auswählen" (*Abbildung unten*) angezeigt, das aus drei Bereichen besteht: (i) dem Bereich "Datenreihen"; (ii) dem Bereich "Achsenwerte" und (iii) der Diagrammdatentabelle. Die einzelnen Bereiche werden im Folgenden näher beschrieben.

Wenn Sie auf die Schaltfläche **Spalten auswählen** klicken, wird das Dialogfeld "Spalten auswählen" 353 aufgerufen, in dem Sie den Quell-XPath und die Datenauswahl für die X- und die Y-Achse ändern können.

© 2016-2022 Altova GmbH



Der Bereich "Datenreihen"

Die Datenreihen, die anfangs im Bereich "Datenreihen" angezeigt werden, sind diejenigen, die im Dialogfeld "Spalten auswählen 353" ausgewählt wurden. Die Datenreihe, die in diesem Bereich zu sehen ist, wenn Sie auf **OK** klicken, ist die Datenreihe, die im Diagramm dargestellt wird. Sie können im Bereich "Datenreihen" drei Operationen durchführen:

- *Hinzufügen und Löschen von Datenreihen:* Auf diese Art können Sie festlegen, welche und wie viele Datenreihen im Diagramm angezeigt werden sollen.
- *Bearbeiten der Namen von Datenreihen:* Die Namen der Datenreihen bilden die Legenden, die im Diagramm angezeigt werden.
- *Auswählen von Daten für einzelne Datenreihen:* Wenn im Bereich "Datenreihen" eine Datenreihe ausgewählt ist, können die Daten für die X- und die Y-Achse im Bereich "Achsenwerte" definiert werden. Eine Anleitung dazu finden Sie weiter unten.

Der Bereich "Achsenwerte"

Die Daten für die X- und Y-Achse können im Bereich "Achsenwerte" in den entsprechenden Textfeldern definiert werden. Wenn Sie in eines dieser Textfelder klicken, kann der Wert darin bearbeitet werden; angezeigt wird dies durch ein Sternchen rechts vom Textfeld. Die Daten können als Bereich aus der Diagrammdatentabelle ausgewählt werden. Ein Bereich kann entweder (i) eine ganze Spalte oder Reihe oder oder (ii) ein Teil einer Spalte oder Reihe sein.Alternativ dazu können die Daten über die Tastatur eingegeben werden (z.B. `A` oder `3` or

`B1:F1`). Um einen Bereich zu markieren, wählen Sie die erste Zelle im Bereich und ziehen Sie den Cursor zur letzten Zelle im Bereich. Um eine ganze Spalte oder Reihe zu markieren, wählen Sie die Spalten- bzw.Zeilenkopfzeile.

Die Auswahl der X-Achse bestimmt die Beschriftung der Nodes auf der X-Achse und gilt für alle Datenreihen. Die Anzahl der Skalenstriche auf der X-Achse wird nicht geändert.

Die Auswahl der Y-Achse bestimmt, welcher Bereich von Zellen für die ausgewählten Datenreihen verwendet wird. Wenn die Anzahl der ausgewählten Zellen kleiner als die Anzahl der Skalenstriche auf der X-Achse ist, wird diese Datenreihe für die letzten Skalenstriche auf der X-Achse nicht berücksichtigt. Sollte die Anzahl der Zellen die Anzahl der Skalenstriche auf der X-Achse übersteigen, werden zusätzliche Skalenstriche generiert. Diese zusätzlichen Skalenstriche entsprechen der Anzahl der zusätzlich ausgewählten Zellen. Die zusätzlichen Werte werden für diese Datenreihe auf der zusätzlichen Anzahl von Skalenstrichen dargestellt.

Diagrammdatentabelle

Die Struktur der Diagrammdatentabelle (im unteren Bereich des Dialogfelds "Daten auswählen") wird anhand der Auswahl im Dialogfeld "Spalten auswählen" 353 ermittelt.

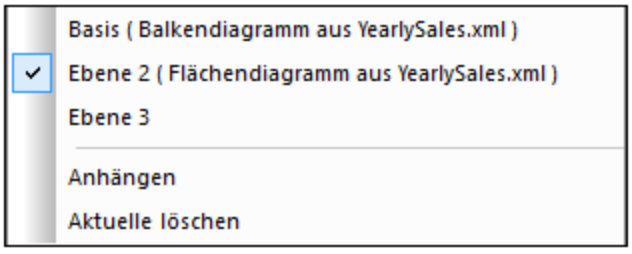
- Die Anzahl der Zeilen in der Tabelle entspricht der Anzahl der Elemente in der Sequenz, die vom Quell-XPath-Ausdruck zurückgegeben wird.
- Die Spalten erhalten einen Buchstaben beginnend mit A, damit diese im Bereich "Achsenwerte" ausgewählt werden können (z.B. `B1:F1`).
- Die erste Spalte wird durch Auswertung der Auswahl für die X-Achse im Dialogfeld "Spalten auswählen" 353 im Kontext der vom Quell-XPath-Ausdruck zurückgegebenen Nodes ermittelt.
- Alle anderen Spalten mit Ausnahme der ersten werden durch Auswertung der Auswahl für die Y-Achse im Dialogfeld "Spalten auswählen" 353 ermittelt. Jede Datenreihe in der Auswahl für die Y-Achse im Dialogfeld "Spalten auswählen" 353 entspricht einer Spalte in der Diagrammdatentabelle.

Die Diagrammdatentabelle kann als Übermenge der Daten, die mit Hilfe der Parameter im Dialogfeld "Spalten auswählen" 353 ausgewählt wurden, betrachtet werden. Von dieser Übermenge aus können Sie anschließend (im Bereich "Achsenwerte") den gewünschten Datenbereich für die einzelnen Datenreihen auswählen.

5.12.6 Overlays

Ein Overlay ist ein Diagramm, das über das Basisdiagramm gelegt wird. Um ein Overlay hinzuzufügen, gehen Sie folgendermaßen vor:

1. Klicken Sie auf die Schaltfläche **Overlays,** um das Overlays-Menü anzuzeigen (*Abbildung unten*).



2. Klicken Sie auf **Anhängen**. Daraufhin wird eine neue Ebene hinzugefügt.

 © 2016-2022 Altova GmbH

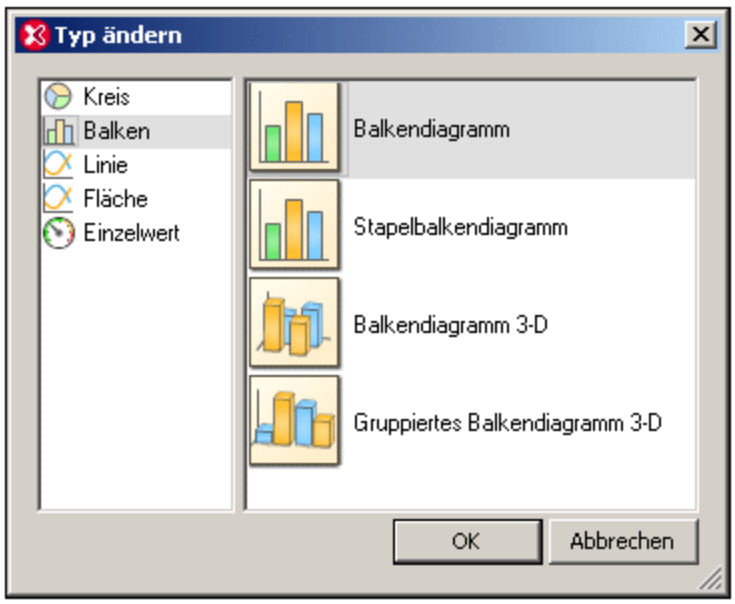
3. Klicken Sie, während die neue Ebene ausgewählt ist, auf die Schaltfläche **Neues Diagramm** und wählen Sie die Daten, wie im Abschnitt Erstellen eines Diagramms[353] beschrieben, aus.
4. Um den Diagrammtyp und die Darstellung zu ändern, klicken Sie auf die Schaltfläche **Typ ändern** bzw. **Darstellung ändern**.

Sie können beliebig viele Overlays hinzufügen. Jede neue Ebene wird an die bestehenden Ebenen im Diagramm angehängt und über diese gelegt. Wenn Sie die Reihenfolge der Ebenen ändern möchten, müssen Sie diese in der richtigen Reihenfolge neu erstellen. Sie können die aktuell ausgewählte Ebene durch Klicken auf **Aktuelle löschen** löschen.

Anmerkung: Jedes Overlay verdeckt die darunterliegenden Ebenen. Da nur Flächendiagramme transparent gemacht werden können, sind einige Ebenenanordnungen eventuell nicht optimal. So würde z.B. eine Balkendiagrammebene über einer Liniendiagrammebene Teile der Linie verdecken. Dies sollten Sie bei der Planung der Ebenenreihenfolge bedenken.

5.12.7 Diagrammeinstellungen: Kurzanleitung

Um den Diagrammtyp im Dialogfeld "Typ ändern" (*Abbildung unten*) auszuwählen, klicken Sie auf den gewünschten Diagrammtyp und dann auf **OK**. Das Dialogfeld "Diagrammtyp" wird durch Klicken auf die Schaltfläche **Typ ändern** aufgerufen.

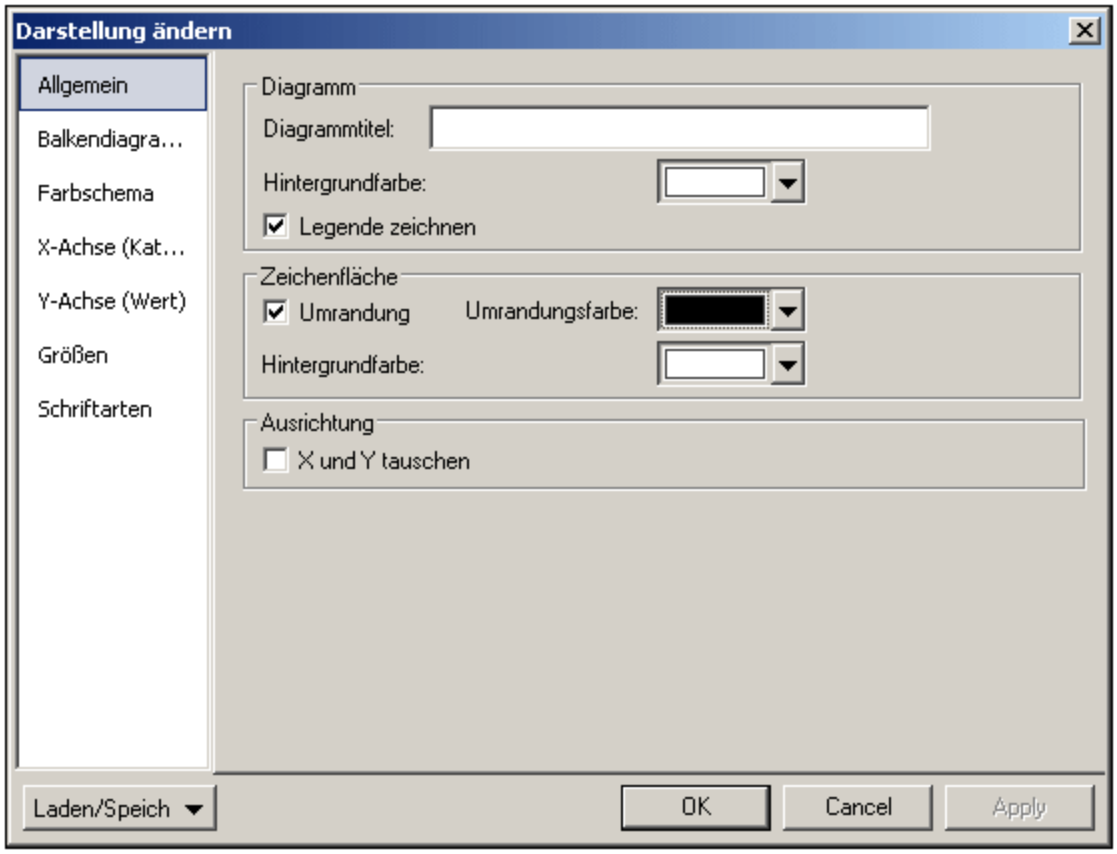


Nachdem der Diagrammtyp ausgewählt wurde, müssen die Diagrammeinstellungen (wie z.B. Titel, Höhe und Breite) im Dialogfeld "Darstellung ändern" vorgenommen und die Diagrammdaten müssen definiert werden. Eine Anleitung zur Auswahl der einzelnen Diagrammtypen finden Sie im Abschnitt Erstellen eines Diagramms[353] und Diagrammdaten[370].

Diagrammdarstellung

Die Diagrammeinstellungen (wie z.B. der Diagrammtitel, das Farbschema und die Schriftgrößen) werden im Dialogfeld "Darstellung ändern" (*Abbildung unten, in der Sie die Einstellungen für ein Balkendiagramm sehen*)

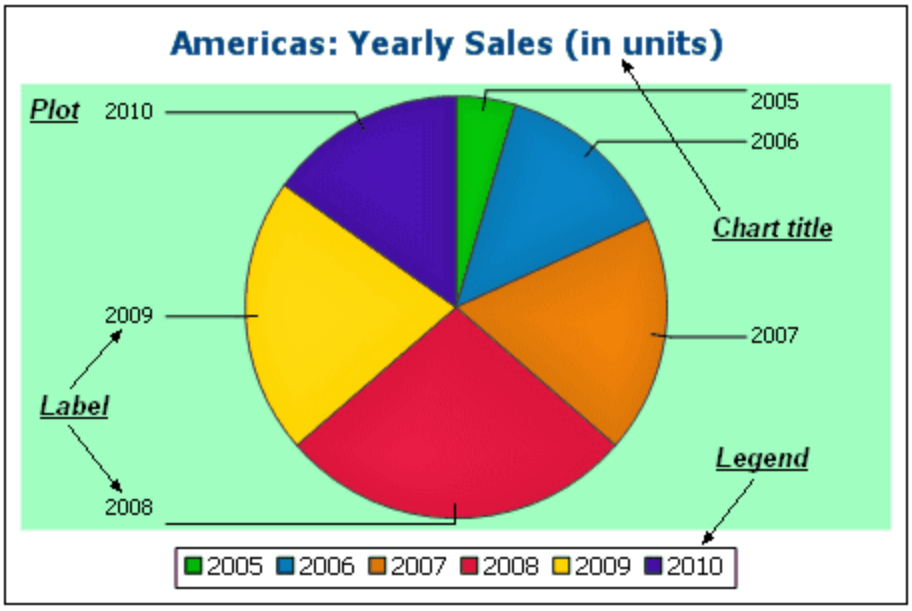
vorgenommen. Dieses Dialogfeld wird über die Schaltfläche "Darstellung ändern" aufgerufen und die Einstellungen darin sind je nach Diagrammtyp unterschiedlich.



Die verschiedenen Einstellungen sind auf die folgenden Register aufgeteilt:

- ***Allgemein:*** Auf diesem Register können der Diagrammtitel (*siehe Abbildung unten*) sowie die Diagrammhintergrundfarbe und die Farbe und Hintergrundfarbe der Diagrammumrandung und der Hintergrundfläche bearbeitet werden. Die Legenden befinden sich am unteren Rand des Diagramms. Sie enthalten die Farbcodes im Diagramm und können im Dialogfeld ein- oder ausgeblendet werden.

© 2016-2022 Altova GmbH



- ***Farbschema:*** Es stehen vier vordefinierte Farbschemen sowie ein benutzerdefiniertes Farbschema zur Verfügung. Sie können jedes dieser Farbschemen durch Hinzufügen und/oder Löschen von Farben in einem Schema anpassen. Das auf diesem Register ausgewählte Farbschema wird im Diagramm verwendet.
- ***Größen:*** Die Größe verschiedener Diagrammaspekte kann entweder in Form von Pixel oder in Form eines Prozentsatzes definiert werden.
- ***Schriftart:*** Auf diesem Register können die Schriftarteigenschaften des Diagrammtitels und der Legende definiert werden. Die Größe kann als Prozentsatz der Diagrammgröße oder in Form von Pixel angegeben werden.

Zusätzlich dazu gibt es eigene Einstellungen für die einzelnen Diagrammtypen. Diese Einstellungen sind:

- ***Kreisdiagramme:*** Einstellungen für: (i) den Winkel, von dem aus das erste Segment gezeichnet werden soll; (ii) die Richtung, in der die Segmente gezeichnet werden sollen; (iii) die Konturfarbe; (iv) ob die Farben hervorgehoben werden sollen (in 3-D-Kreisdiagrammen: ob Schlagschatten und transparente Flächen verwendet werden); (v) ob Beschriftungen angezeigt werden sollen; und (vi) ob zu den Beschriftungen Werte und Prozentwerte hinzugefügt werden sollen und wie viele Dezimalstellen zum Prozentwert hinzugefügt werden sollen.
- ***Balkendiagramme:*** Einstellungen für: *(Allgemein)* Wenn die X- und Y-Achse vertauscht werden, wird ein horizontales Balkendiagramm gezeichnet; *(Balkendiagramm)* Balkenkonturen und Schlagschatten (Schlagschatten nur bei 2-D-Balkendiagrammen); *(X-Achse)* Beschriftung und Farbe der X-Achse und vertikale Rasterlinien; *(Y-Achse)* Beschriftung und Farbe der Y-Achse, horizontale Rasterlinien, der anzuzeigende Wertebereich und die Skalenstriche auf der Y-Achse; *(Z-Achse, nur 3-D)* Beschriftung und Farbe der Z-Achse; *(3D)* die vertikale Neigung, die horizontale Drehung und die Breite der Ansicht.
- ***Liniendiagramme:*** Einstellungen für: *(Allgemein)* Vertauscht gezeichnete X- und Y-Achse; *(Liniendiagramm)* Inkludieren der Diagrammpunkte oder Ausblenden; *(X-Achse)* Beschriftung und Farbe der X-Achse und vertikale Rasterlinien; *(Y-Achse)* Beschriftung und Farbe der Y-Achse, horizontale Rasterlinien, den Bereich der anzuzeigenden Werte und die Skalenstriche auf der Y-Achse.
- ***Einzelwertdiagramm:*** Einstellungen für: (i) den Winkel, bei dem das Tachometerdiagramm beginnt und den Pfeilwinkel der Skala (*nur Tachometerdiagramm*); (ii) den Bereich der anzuzeigenden Werte (*Tachometerdiagramm und Einzelwertbalken)*; (iii) das Intervall und die Farbe der Skalenhaupt- und

Zwischenwerte (*Tachometerdiagramm und Einzelwertbalken*); (iv) die Farben der Scheibe, der Nadel und der Umrandung.

5.12.8 Diagrammeinstellungen und Diagrammdarstellung

Die Diagrammeinstellungen sind folgendermaßen gegliedert:

- Diagrammgrundeinstellungen 376: Die wichtigste Grundeinstellung ist der Diagrammtyp. Um den Diagrammtyp auszuwählen, klicken Sie in der Symbolleiste des Diagrammfensters auf "**Typ ändern**". Daraufhin wird das Dialogfeld "Typ ändern" 376 angezeigt.
- Erweiterte Diagrammeinstellungen 383,Hier können Sie das Aussehen eines Diagramms (Titel, Legende, Farben, Schriftarten, usw.) ändern. Die erweiterten Einstellungen werden im Dialogfeld "Darstellung ändern" 383 vorgenommen. Um dieses Dialogfeld aufzurufen, klicken Sie in der Symbolleiste des Diagrammfensters auf die Schaltfläche **Alle Einstellungen**.

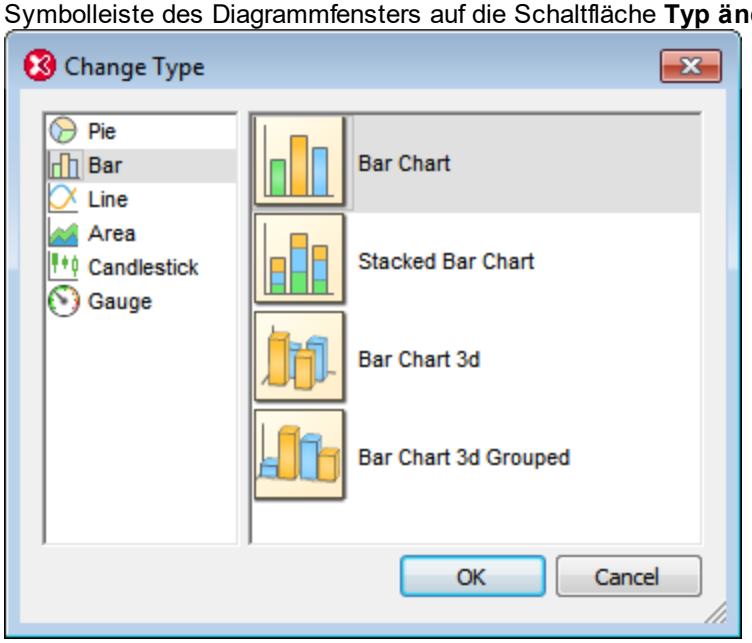
5.12.8.1 Grundlegende Diagrammeinstellungen

In diesem Abschnitt

- Einstellen des Diagrammtyps 376
- Liste der Diagrammtypen 377
- Andere Grundeinstellungen 382

Einstellen des Diagrammtyps

Die wichtigste Grundeinstellung ist der Diagrammtyp. Klicken Sie zur Auswahl des Diagrammtyps in der Symbolleiste des Diagrammfensters auf die Schaltfläche **Typ ändern.**



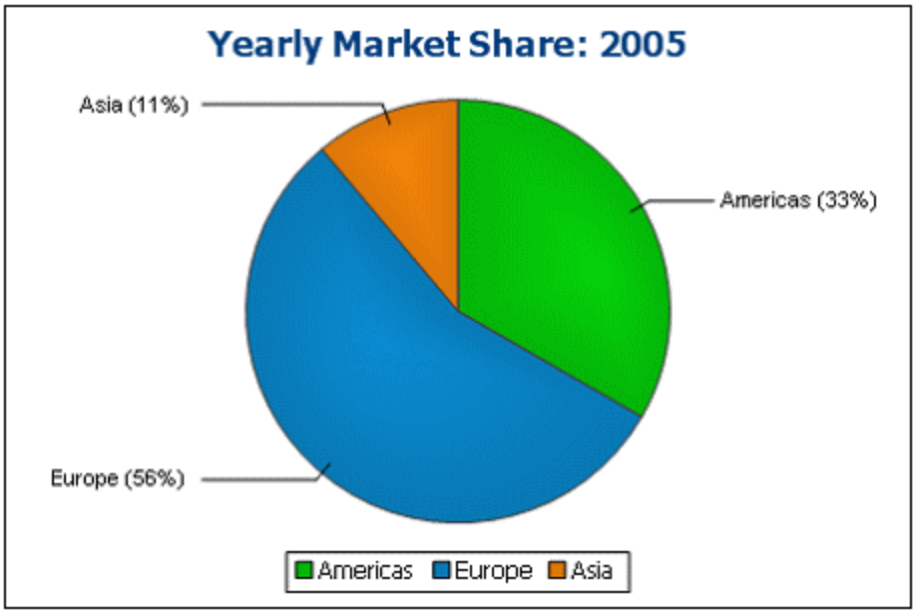
© 2016-2022 Altova GmbH

Diagrammtypen

Es stehen die folgenden unten aufgelisteten Diagrammtypen zur Verfügung. Wählen Sie den Diagrammtyp im Dialogfeld "Typ ändern" 376 (*Abbildung oben*) aus und klicken Sie auf **OK**.

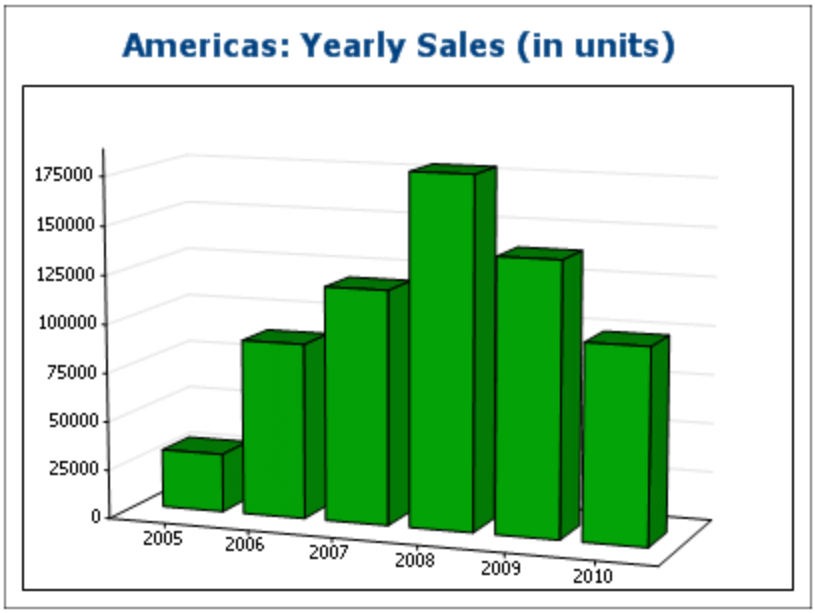
- *Kreisdiagramme*

  In Kreisdiagrammen liefert eine Spalte/Achse die Werte, eine weitere die Beschriftungen für diese Werte. Die Beschriftungsspalte/-achse kann auch nicht-numerische Werte enthalten.

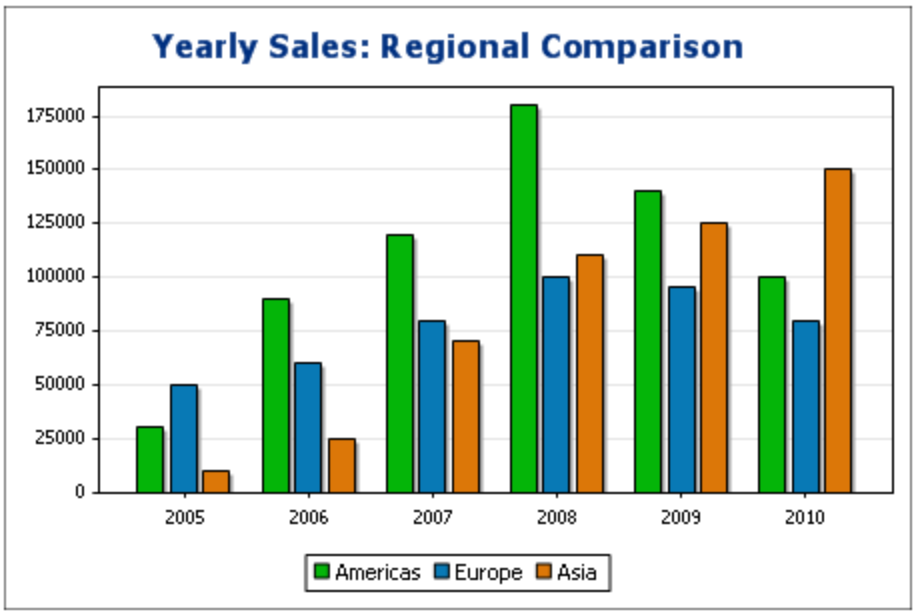


- *Balkendiagramme*

  In Balkendiagramme können zwei Wertegruppen auf zwei Achsen dargestellt werden (*siehe unten*).

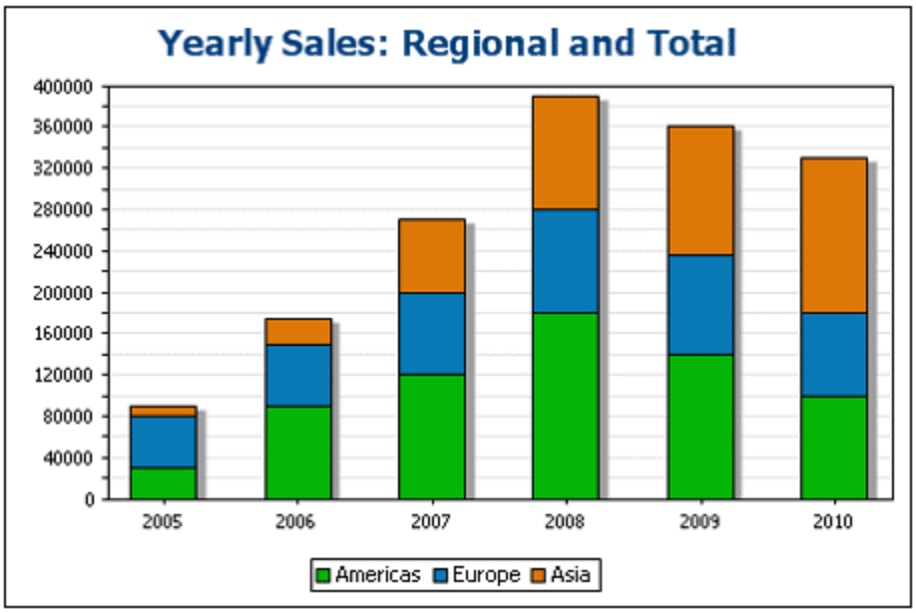


Es können auch, wie im Beispiel unten, drei Wertegruppen verwendet werden: (i) Kontinent, (ii) Jahr, (iii) Umsatzvolumen. Balkendiagramme können in 2-D (*siehe unten*) oder 3-D (*oben*) dargestellt werden.



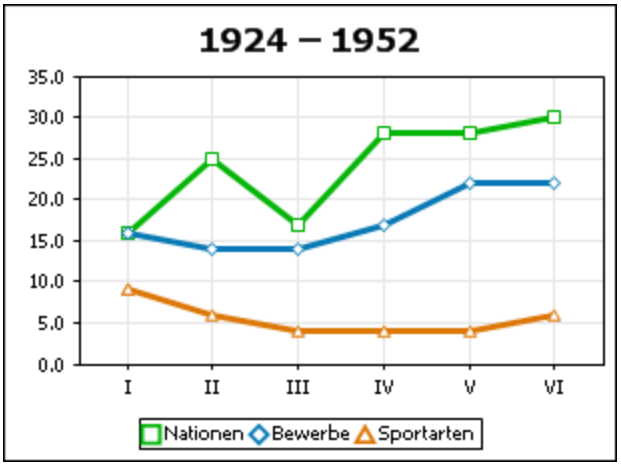
Ein Balkendiagramm mit drei Achsen kann auch gestapelt dargestellt werden, wenn Sie die Gesamtsumme darstellen möchten. Vergleichen Sie das unten gezeigte Stapelbalkendiagramm mit dem Diagramm oben. Im Stapelbalkendiagramm wird der Gesamtumsatz auf allen Kontinenten angezeigt.

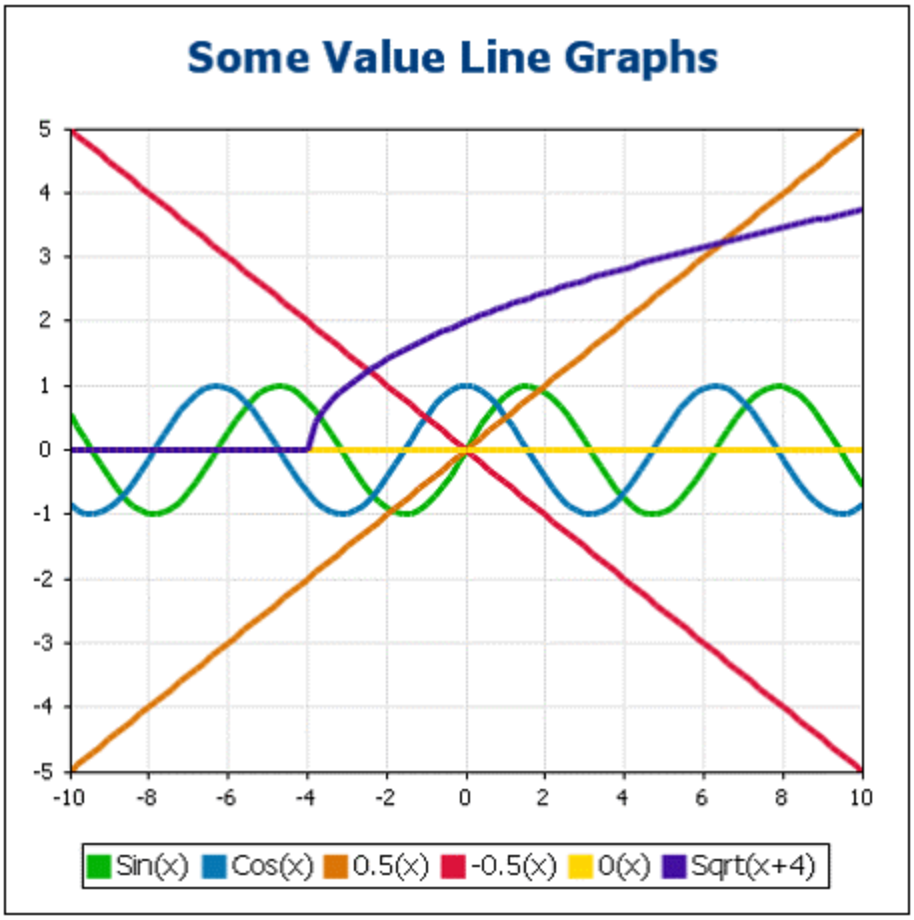
© 2016-2022 Altova GmbH



- *Liniendiagramme*

  Der Unterschied zwischen normalen Liniendiagrammen (*Abbildung unten links*) und Wert-Liniendiagrammen (*Abbildung unten rechts*) besteht darin, dass in Wert-Liniendiagrammen nur numerische Werte für die x-Achse verwendet werden können. Wenn Sie Liniendiagramme mit Textwerten auf der X-Achse anzeigen müssen, verwenden Sie Liniendiagramme.

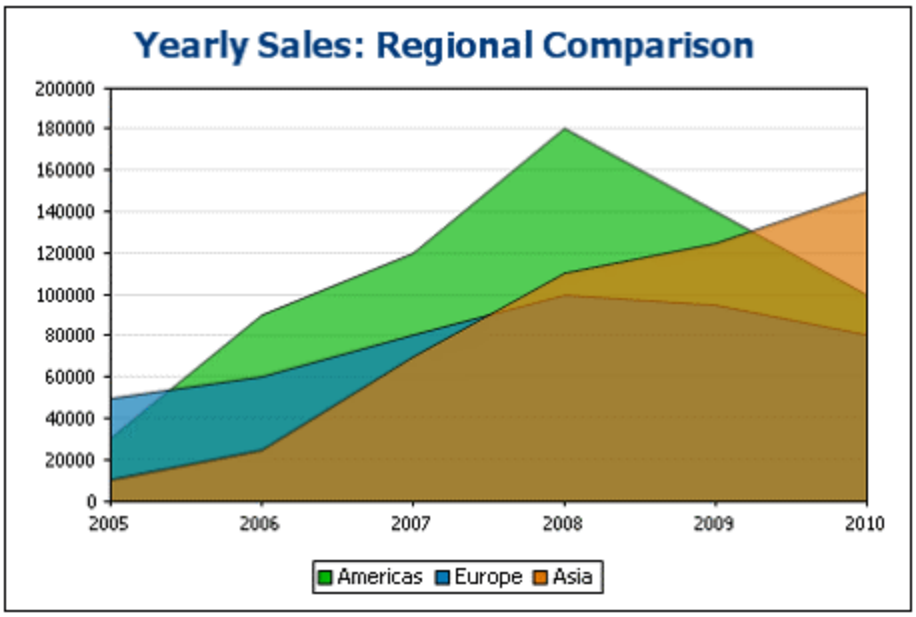




- *Flächendiagramme*

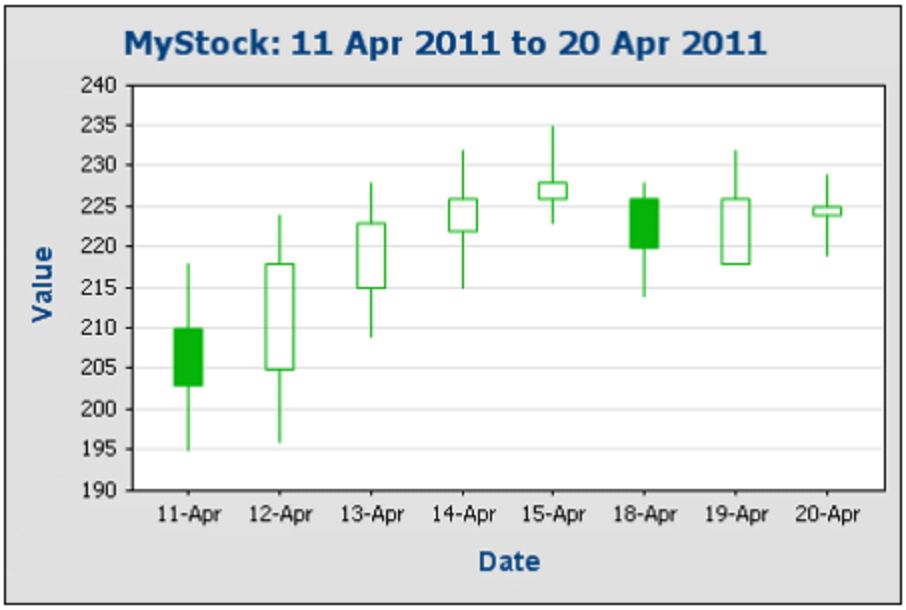
  Flächendiagramme sind eine Variante von Liniendiagrammen, bei der die Fläche unterhalb der Linien ebenfalls eingefärbt ist. Beachten Sie, dass Flächendiagramme ebenfalls gestapelt werden können (*siehe Balkendiagramme oben).*

© 2016-2022 Altova GmbH



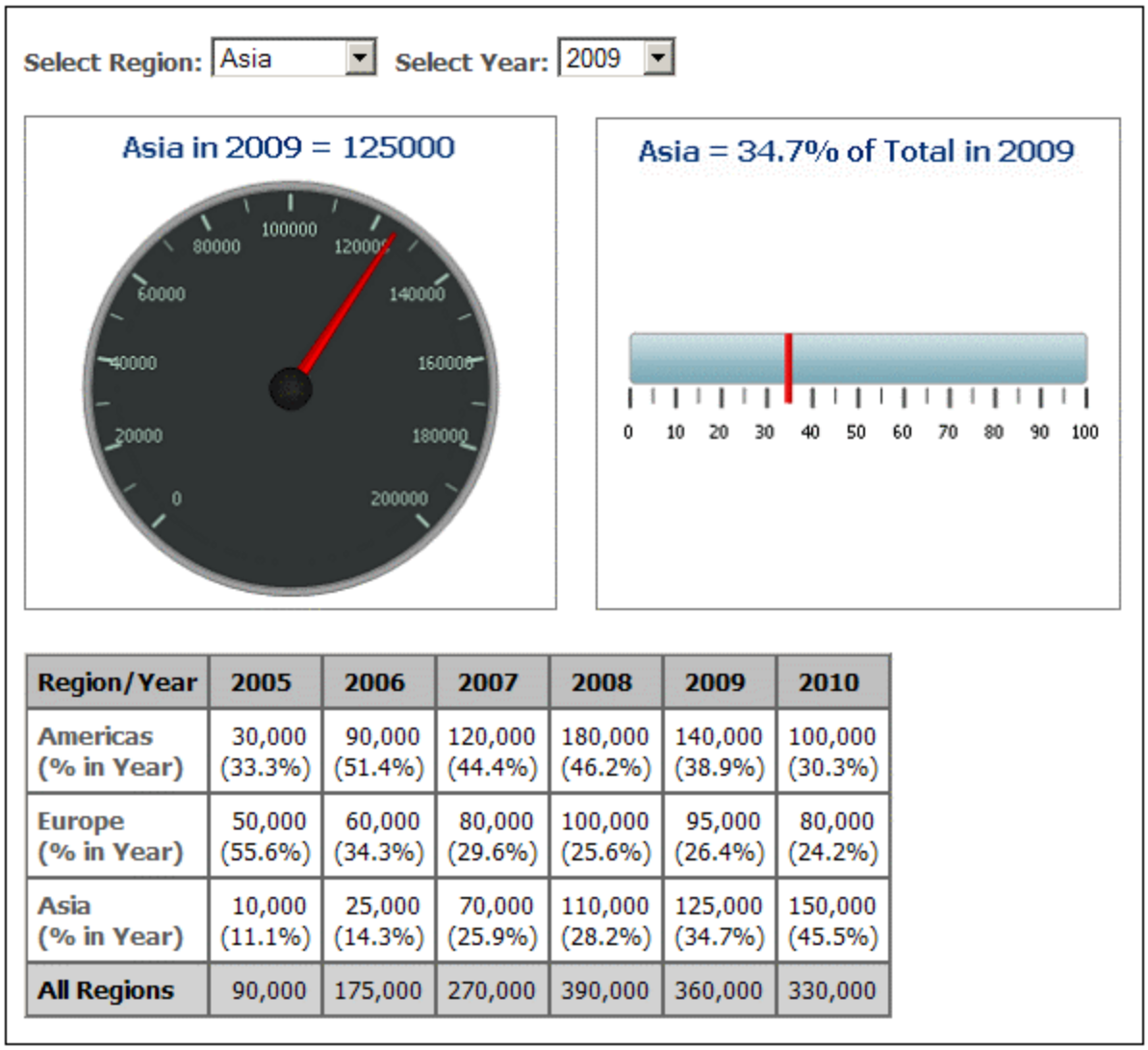
- *Kerzendiagramme*

  Ein Kerzendiagramm kann verwendet werden, um Preisbewegungen von Wertpapieren, Waren oder Währungen usw. über eine bestimmte Zeitspanne abzubilden. Das Diagramm vermittelt einen Eindruck davon, wie sich der Vermögenswert im Lauf der Zeit entwickelt hat und gibt auch Informationen über Schlusskurs, Hoch- und Tiefststand sowie optional auch den Anfangskurs des Tages. Auf der Y-Achse werden drei oder vier Datenreihen (Schlusskurs, Hoch-, Tiefstand und (optional) Anfangskurs) angezeigt. In der Abbildung unten sehen Sie ein Kerzendiagramm mit vier Datenreihen.



- *Einzelwertdiagramme*

  Einzelwertdiagramme werden verwendet, um einen einzelnen Wert abzubilden und ihn im Verhältnis zu einem Minimal- und Maximalwert darzustellen.



Region/Year	2005	2006	2007	2008	2009	2010
Americas (% in Year)	30,000 (33.3%)	90,000 (51.4%)	120,000 (44.4%)	180,000 (46.2%)	140,000 (38.9%)	100,000 (30.3%)
Europe (% in Year)	50,000 (55.6%)	60,000 (34.3%)	80,000 (29.6%)	100,000 (25.6%)	95,000 (26.4%)	80,000 (24.2%)
Asia (% in Year)	10,000 (11.1%)	25,000 (14.3%)	70,000 (25.9%)	110,000 (28.2%)	125,000 (34.7%)	150,000 (45.5%)
All Regions	90,000	175,000	270,000	390,000	360,000	330,000

Andere grundlegende Einstellungen

Im Bereich "Diagrammeinstellungen" können Sie auch den Titel des Diagramms einstellen (*siehe Abbildung unten*).

© 2016-2022 Altova GmbH

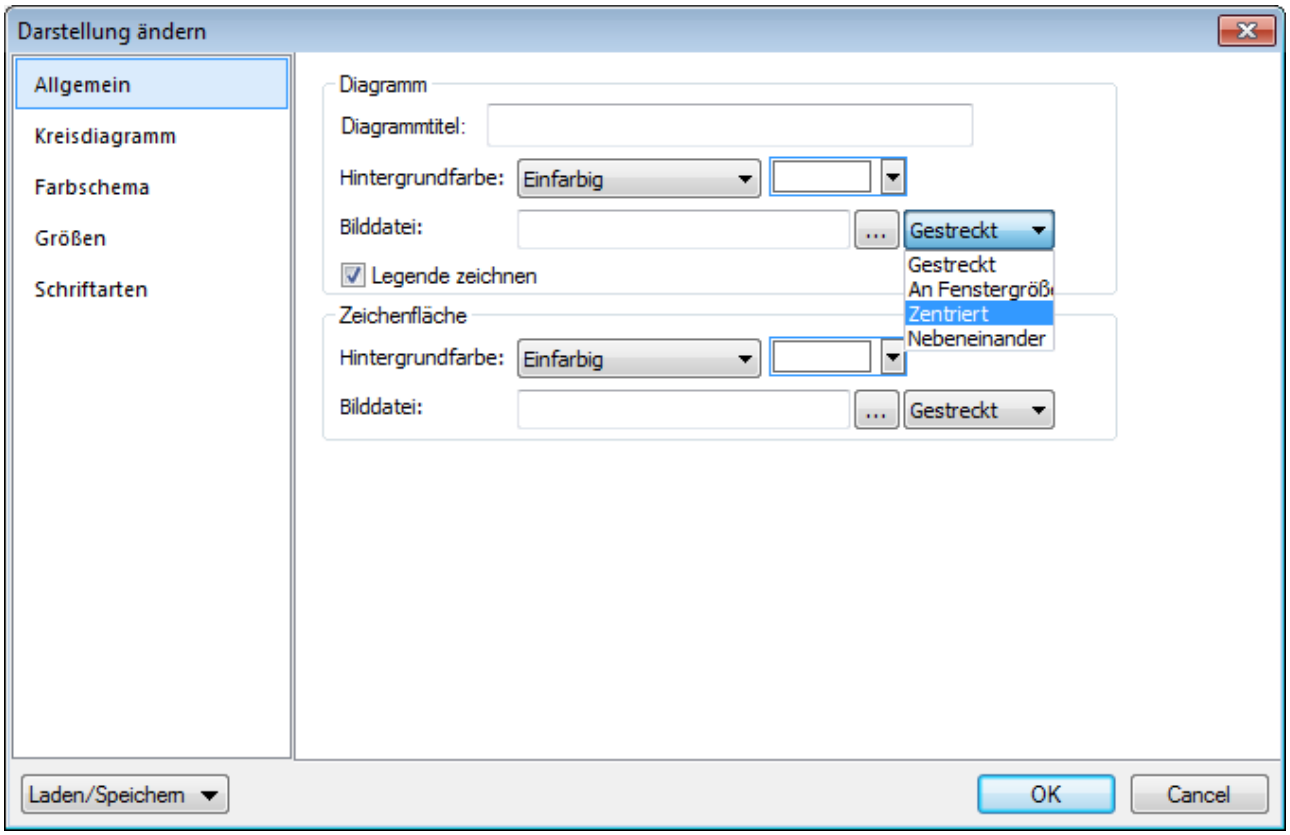
5.12.8.2 Erweiterte Diagrammeinstellungen

Dieser Abschnitt:

- Aufrufen der erweiterten Einstellungen 383
- Übersicht über die erweiterten Einstellungen 383
- Laden, Speichern, Zurücksetzen von Diagrammeinstellungen 386

Aufrufen der erweiterten Einstellungen

Die erweiterten Einstellungen eines Diagramms werden durch Klicken auf die Schaltfläche **Alle Einstellungen** in der Symbolleiste des Diagrammfensters aufgerufen. Daraufhin erscheint das Dialogfeld "Darstellung ändern" für den jeweiligen Diagrammtyp (*In der Abbildung unten sehen Sie das Dialogfeld "Darstellung ändern" für ein Kreisdiagramm*).



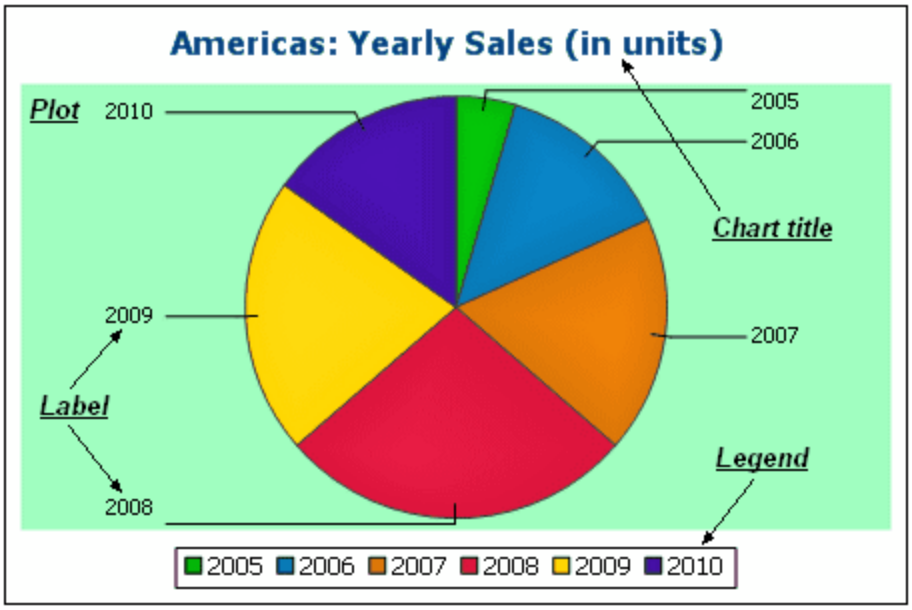
Übersicht über die erweiterten Einstellungen

Die erweiterten Einstellungen sind auf Registern gegliedert. Einige davon gelten für alle Diagrammtypen und einige für einen bestimmten Diagrammtyp.

Allgemeine Diagrammeinstellungen

- *Allgemein*

Der Diagrammtitel (*siehe Abbildung unten*) ist mit dem Titel in den Grundeinstellungen (*siehe oben*) identisch und kann auch unter den erweiterten Einstellungen bearbeitet werden. Weitere Einstellungen in diesem Dialogfeld sind die Hintergrundfarbe des Diagramms und die Farbe und Hintergrundfarbe der Umrandung. Die Diagrammfläche in der Abbildung unten hat eine hellgrüne Hintergrundfarbe erhalten. Sie können als Hintergrundbild des Diagramms und/oder der Diagrammfläche auch eine Bilddatei verwenden. Dieses Bild kann auf den gesamten Bereich des Diagramms oder die gesamte Diagrammfläche gestreckt werden; es kann in der Größe auch an die Höhe oder Breite des Diagramms/der Diagrammfläche angepasst werden; es kann zentriert werden oder nebeneinander angezeigt werden. Die Legende enthält den Schlüssel zu den Farbcodes im Diagramm und kann ein- oder ausgeblendet werden.



- *Farbschema*

  Es stehen vier vordefinierte Farbschemen sowie ein benutzerdefiniertes Farbschema zur Verfügung. Sie können jedes dieser Farbschemen durch Hinzufügen und/oder Löschen von Farben in einem Schema anpassen. Das auf dem Register ausgewählte Farbschema wird im Diagramm verwendet.

- *Größen*

  Die Größe verschiedener Diagrammaspekte kann entweder in Form von Pixel oder in Form eines Prozentsatzes definiert werden.

- *Schriftart*

  Auf diesem Register können die Schriftarteigenschaften des Diagrammtitels und der Legende definiert werden. Die Größe kann als Prozentsatz der Diagrammgröße oder absolut in Form von Punkten angegeben werden.

- *Schaltfläche "Laden/Speichern"*

© 2016-2022 Altova GmbH

Die Einstellungen können in einer XML-Datei gespeichert und aus einer XML-Datei mit der korrekten Struktur geladen werden. Um die Struktur zu sehen, speichern Sie die Einstellungen eines Diagramms und öffnen Sie anschließend die XML-Datei. Wenn Sie auf diese Schaltfläche klicken, haben Sie auch die Option, die Diagrammeinstellungen auf die Standardeinstellungen zurückzusetzen.

Typspezifische Diagrammeinstellungen

- *Kreisdiagramme*

  Einstellungen für: (i) den Winkel, von dem aus das erste Segment gezeichnet werden soll; (ii) die Richtung, in der die Segmente gezeichnet werden sollen; (iii) die Konturfarbe; (iv) ob die Farben hervorgehoben werden sollen (in 3-D-Kreisdiagrammen: ob Schlagschatten und transparente Flächen verwendet werden); (v) ob Beschriftungen angezeigt werden sollen und (vi) ob zu den Beschriftungen Werte und Prozentwerte hinzugefügt werden sollen und wie viele Dezimalstellen für die Prozentwerte angezeigt werden sollen.

- *Balkendiagramme*

  Einstellungen für: *(Allgemein)* Wenn die X- und Y-Achse vertauscht werden, wird ein horizontales Balkendiagramm gezeichnet; *(Balkendiagramm)* Balkenkonturen und Schlagschatten (Schlagschatten nur bei 2-D-Balkendiagrammen); *(X-Achse)* Beschriftung und Farbe der X-Achse und vertikale Rasterlinien; *(Y-Achse)* Beschriftung und Farbe der Y-Achse, horizontale Rasterlinien, der anzuzeigende Wertebereich und die Skalenwerte auf der Y-Achse; *(Z-Achse, nur 3-D)* Beschriftung und Farbe der Z-Achse; *(3D)* die vertikale Neigung, die horizontale Drehung und die Breite der Ansicht.

- *Liniendiagramme*

  Einstellungen für: *(Allgemein)* Vertauscht gezeichnete X- und Y-Achse; *(Liniendiagramm)* Inkludieren der Diagrammpunkte oder Ausblenden; *(X-Achse)* Beschriftung und Farbe der X-Achse und vertikale Rasterlinien; *(Y-Achse)* Beschriftung und Farbe der Y-Achse, horizontale Rasterlinien, den Bereich der anzuzeigenden Werte und die Skalenwerte auf der Y-Achse.

- *Einzelwertdiagramme*

  Einstellungen für: (i) den Winkel, bei dem das Tachometerdiagramm beginnt und den Pfeilwinkel der Skala; (ii) den Bereich der anzuzeigenden Werte; (iii) das Intervall und die Farbe der Skalenhaupt- und Zwischenwerte; (iv) die Farben der Scheibe, der Nadel und der Umrandung.

- *Flächendiagramme*

  Die Transparenz von Flächen kann als ein Wert zwischen 0 (keine Transparenz) und 255 (maximale Transparenz) definiert werden. Bei Flächendiagrammen, bei denen es sich nicht um Stapelflächendiagrammen handelt, werden bei transparenten Flächen andere darunter liegenden Flächenbereiche sichtbar. Es können auch Konturen für die Flächen definiert werden.

- *Kerzendiagramme*

Die Füllfarbe kann für die folgenden beiden Situationen definiert werden: (i) wenn der Wert des Schlusskurses höher ist als der Wert des Anfangskurses und (ii) wenn der Wert des Anfangskurses höher ist als der Wert des Schlusskurses. Im zweiten Fall steht auch die Farbe der Datenreihe als Option zur Verfügung. Die Farbe der Datenreihe wird im Dialogfeld "Darstellung ändern" auf dem Register "Farbschema" definiert.

Laden, Speichern, Zurücksetzen von Diagrammeinstellungen

Diagrammeinstellungen, die nicht den Standardeinstellungen entsprechen, können in einer XML-Datei gespeichert werden. Diese Einstellungen können später als die Einstellungen eines Diagramms geladen werden, wodurch Sie Zeit sparen können. Wenn Sie auf die Schaltfläche **Laden/Speichern** klicken (siehe erste Abbildung in diesem Abschnitt 383), stehen die folgenden Optionen zur Verfügung:

- **Auf Standardeinstellungen setzen**: Verwirft alle Änderungen an den Einstellungen und stellt die Standardeinstellungen wieder her. Bitte beachten Sie, dass dadurch der Standard in allen Abschnitten des Dialogfelds wiederhergestellt wird und nicht nur im gerade angezeigten Abschnitt!

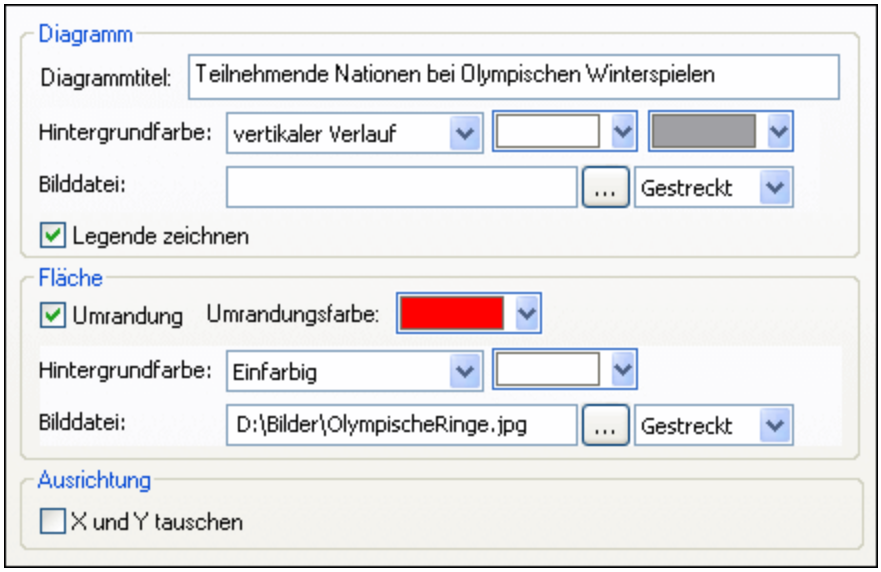
- **Aus Datei laden**: Ermöglicht das Importieren von bereits in einer XML-Datei gespeicherten Einstellungen (*siehe nächster Befehl).* Zeigt das Windows-Dialogfeld Öffnen an, in dem Sie eine Datei auswählen und früher gespeicherte Einstellungen importieren können.

- **In Datei speichern**: Öffnet das Windows Dialogfeld Speichern unter, in dem Sie einen Dateinamen und Pfad angeben können, unter dem die Einstellungen gespeichert werden sollen. Die von den Standardeinstellungen abweichenden Einstellunghen werden in der Datei gespeichert.

5.12.8.2.1 Allgemein

Im Abschnitt Allgemein des Dialogfelds Darstellung ändern können Sie die Überschrift des Diagramms definieren, Legenden hinzufügen oder entfernen sowie Hintergrundbilder und -farben und, für Balken-, Linien-, Flächen- und Kerzendiagramme, die Ausrichtung des Diagramms definieren.

© 2016-2022 Altova GmbH



Diagramm

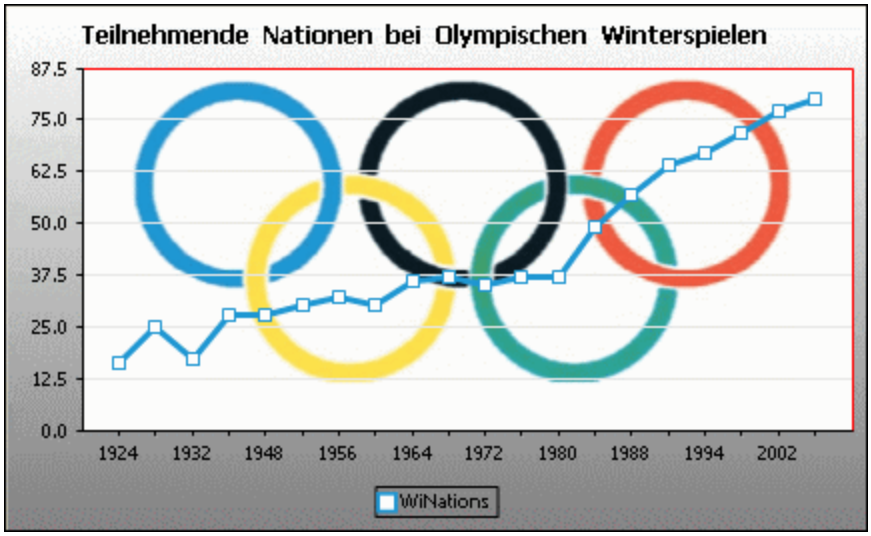
Geben Sie im Feld `Diagrammtitel` einen aussagekräftigen Titel für Ihr Diagramm ein und wählen Sie eine Hintergrundfarbe für das gesamte Diagramm aus der Dropdown-Liste. Sie können einen einfarbigen Hintergrund, einen vertikalen oder einen horizontalen Verlauf sowie Start- und Endfarben für den Verlauf definieren, falls erforderlich. Zusätzlich zum bzw. anstatt eines farbigen Hintergrunds können Sie auch ein Hintergrundbild definieren und eine der verfügbaren Optionen aus der Dropdown-Liste wählen:

- Gestreckt: Das Bild wird auf die Höhe und Breite des Diagramms gestreckt
- An Fenstergröße anpassen: Das Bild wird in den Rahmen des Diagramms eingepasst wobei das Seitenverhältnis des Bilds beibehalten wird
- Zentriert: Das Bild wird in Originalgröße in der Mitte des Diagramms angezeigt
- Nebeneinander: Wenn das Bild kleiner als das Diagramm ist, werden mehrere Ausgaben des Bilds angezeigt um den Hintergrund auszufüllen

Das Kontrollkästchen `Legende zeichnen` ist standardmäßig aktiviert, sie müssen es deaktivieren, wenn Sie keine Legende in Ihrem Diagramm anzeigen möchten.

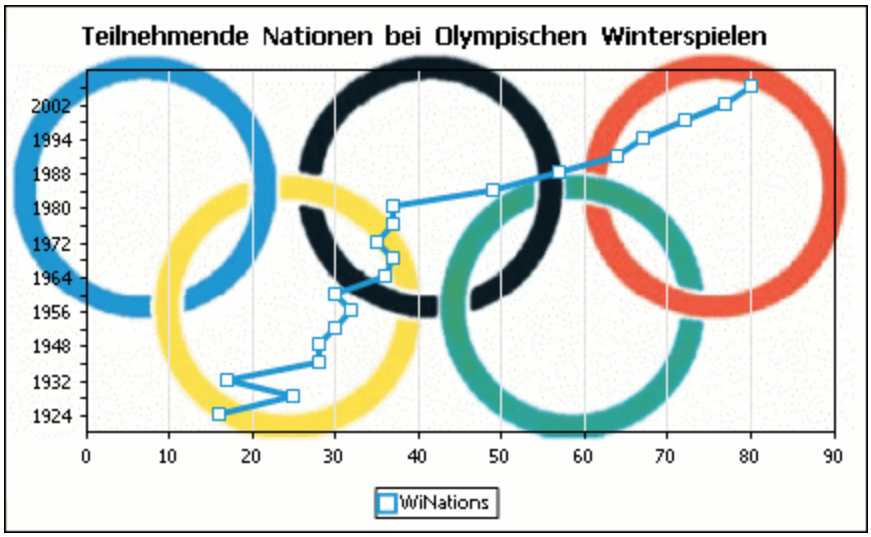
Fläche

Die Fläche ist jener Bereich, in dem die Daten des Diagramms angezeigt werden. Sie können rund um die Fläche eine Umrandung ziehen und für den Bereich der Fläche eine andere Hintergrundfarbe oder ein Hintergrundbild festlegen. In der unten stehenden Abbildung wurde die Hintergrundfarbe des Diagramms auf Grau (vertikaler Verlauf) geändert, während die Fläche weiterhin weiß ist, eine rote Umrandung rundherum gezogen und ein Hintergrundbild hinzugefügt wurde.



Ausrichtung

Wenn Sie eine kleine Datenreihe mit großen Werten haben, kann es für eine bessere Darstellung von Vorteil sein, die x- und die y-Achsen zu tauschen. Beachten Sie, dass in der unten stehenden Abbildung auch die Hintergrundfarbe der Fläche auf "Transparent" gesetzt und das Hintergrundbild dem gesamten Diagramm zugewiesen wurde.



Beachten Sie, dass diese Option für Kreis- und Einzelwertdiagramme nicht verfügbar ist.

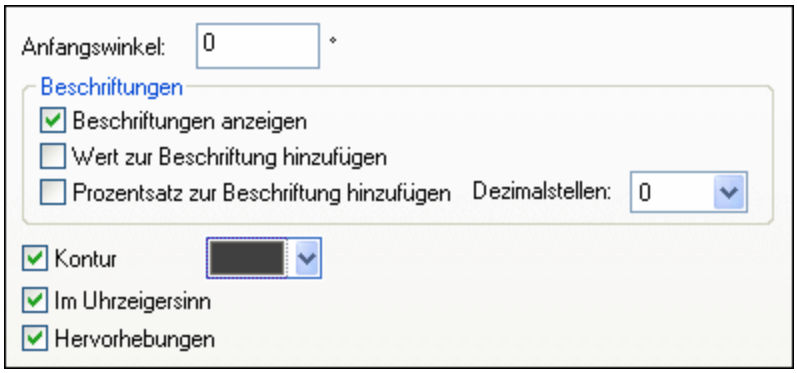
© 2016-2022 Altova GmbH

5.12.8.2.2 Typbezogene Funktionen

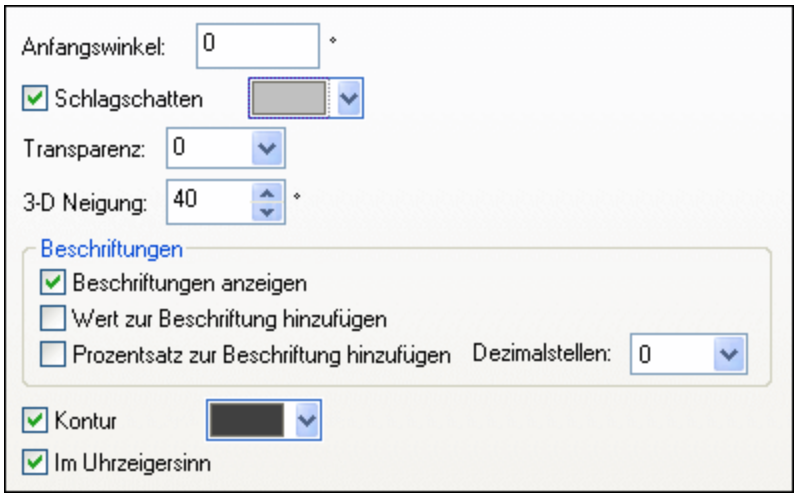
Für jeden Diagrammtyp und sogar für die verschiedenen Untertypen gibt es im Dialogfeld "Darstellung ändern" einen Abschnitt, in dem Sie die typbezogenen Funktionen des Diagramms definieren können.

Kreisdiagramm

Die meisten Einstellungen sind für die 2-D und 3-D Version gleich. In 2-D Kreisdiagrammen können Sie zusätzlich Hervorhebungen einzeichnen.



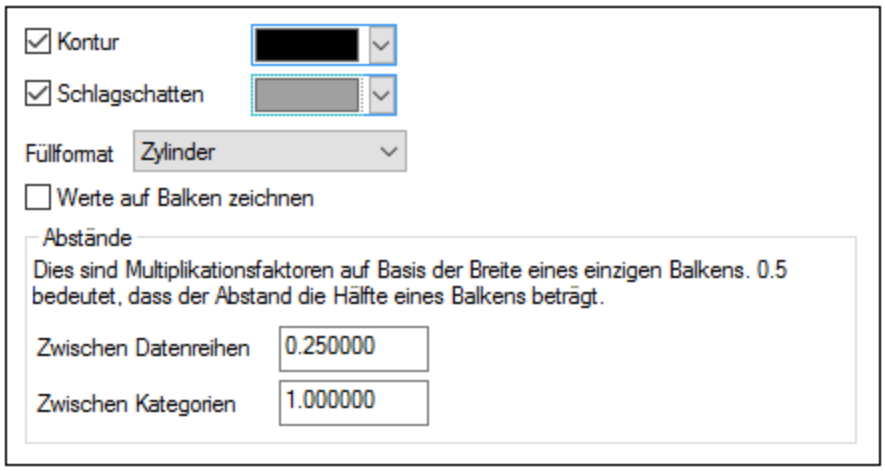
In 3D Kreisdiagrammen können Sie Schlagschatten anzeigen, Transparenz hinzufügen und die 3-D Neigung definieren.



Der Wert `Anfangswinkel` definiert, wo die erste Zeile der ausgewählten Spalte im Diagramm angezeigt wird. Ein Winkel von 0 Grad entspricht der 12 Uhr Position auf einer Uhr.
Sie können zusätzlich zur bzw. anstatt der Legende Beschriftungen anzeigen, Werte und/oder Prozentangaben zu den Beschriftungen hinzufügen und definieren, mit wie vielen Dezimalstellen die Prozentwerte angezeigt werden sollen.
Die Farbe, die Sie neben dem Kontrollkästchen `Kontur` auswählen können, wird für die optionale Umrandung des Diagramms und der einzelnen Segmente verwendet. Über das Kontrollkästchen `Im Uhrzeigersinn` wird definiert, ob die Zeilen im Uhrzeigersinn oder Gegenuhrzeigersinn aufgelistet werden sollen.

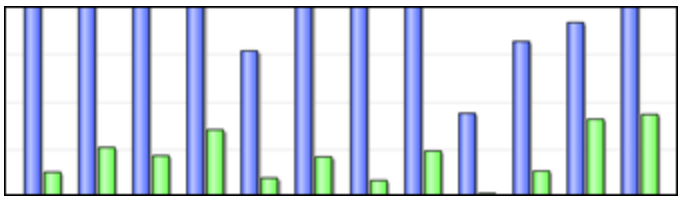
In 3-D Kreisdiagrammen können Sie einen Schlagschatten zeichnen und seine Farbe definieren sowie Transparenz zum Diagramm hinzufügen und die 3-D Neigung definieren. In 2-D Diagrammen fügt die Option `Hervorhebungen` eine zusätzliche Struktur zum Diagramm hinzu.

Balkendiagramm



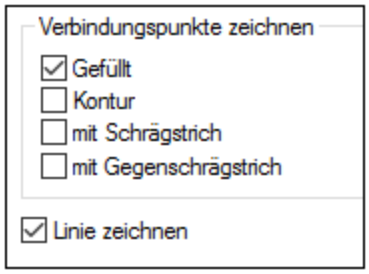
Für Balkendiagramme können Sie die folgenden Einstellungen vornehmen:

- Sie können eine Umrandung zu den Balken hinzufügen und ihre Farbe definieren.
- In 2-D Balkendiagrammen können Sie auch einen Schlagschatten zeichnen und seine Farbe definieren (diese Option steht für 3-D Balkendiagramme nicht zur Verfügung).
- In der Standardeinstellung erinnert die Form der Balken an einen Zylinder, Sie können aber auch "vertikaler Verlauf" oder "Einfarbig" aus der Dropdown-Liste `Füllformat` wählen (Diese Option steht nur für 2D-Diagramme zur Verfügung).
- Die Werte eines Balkens (sie entsprechen der Höhe des Balkens auf der y-Achse) können auf dem Balken gezeichnet werden. Die Schriftart der Werte kann in der Einstellung *Schriftarten* definiert werden (Diese Option steht nur für 2D-Diagrammem, nicht aber für Stapelbalkendiagramme zur Verfügung).
- Der Abstand zwischen den Datenreihen einer Balkengruppe und zwischen Balkengruppen kann als Dezimalbruch der Breite eines einzigen Balkens definiert werden. In der nachstehenden Abbildung, in der Sie aus je einer blauen und einer grünen Datenreihe bestehende Balkengrupen sehen, wurde z.B. der Abstand zwischen den Datenreihen mit 25 % (`=0,25`) der Breite eines Balkens definiert; der Abstand zwischen Balkengruppen wurde auf 100% (`=1,0`) der Breite eines Balkens gesetzt. Diese Option steht nur für 2D-Balkendiagramme zur Verfügung.



© 2016-2022 Altova GmbH

Liniendiagramm



Um Verbindungspunkte zum Markieren der Werte in Liniendiagrammen anzuzeigen, muss im Gruppenfeld "Verbindungspunkte zeichnen" zumindest ein Kontrollkästchen aktiviert sein. In DatabaseSpy werden fünf verschiedene Formen zur Kennzeichnung der Datenreihen verwendet: Quadrat, Raute, Dreieck, umgekehrtes Dreieck und Kreis. Wenn in Ihrem Diagramm mehr als fünf Datenreihen vorkommen, können Sie die Formen kombinieren, indem Sie im Gruppenfeld "Verbindungspunkte zeichnen" mehr als eine Option auswählen. In der unten stehenden Abbildung wurden sowohl `Gefüllt` als auch `mit Schrägstrich` gewählt; der Typ `mit Schrägstrich` wird ab der sechsten Datenreihe verwendet.

Durch die Option *Linie zeichnen* kann das Diagramm (i) nur mit Verbindungspunkten oder (ii) mit durch eine Linie verbundenen Verbindungspunkten gezeichnet werden.

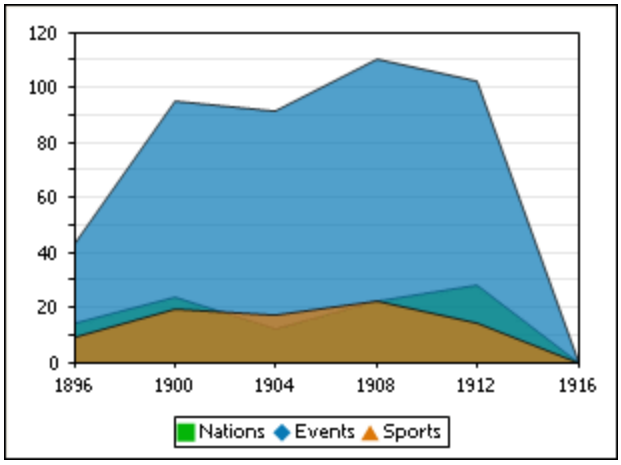


Verbindungspunkte stehen sowohl für Liniendiagramme als auch für Werte-Liniendiagramme zur Verfügung.

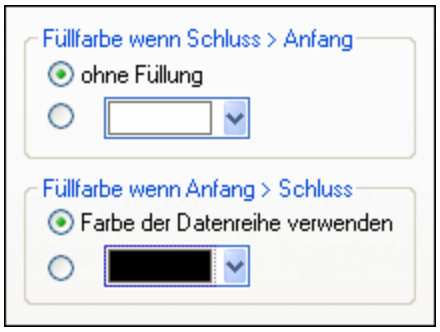
Flächendiagramm



Zu den Eigenschaften, die Sie für Flächendiagramme ändern können, zählt Transparenz: Auf diese Weise können Sie verhindern, dass eine Datenreihe im Diagramm von einer anderen verdeckt wird. Darüber hinaus können Sie eine Kontur zu den einzelnen Datenflächen hinzufügen und ihre Farbe definieren (*siehe unten stehende Abbildung*).



Kerzendiagramm



Wenn sowohl Anfangs- als Schlusswert als Datenreihen definiert sind, können Sie die Farben definieren und festlegen, ob die Kerze gefüllt sein soll, wenn der Schlusswert größer ist als der Anfangswert.

© 2016-2022 Altova GmbH

Einzelwertdiagramm



Der Wert `Start` im Gruppenfeld "Winkel" definiert die Position der Nullmarke und der Wert `Skalenwinkel` ist der Winkel, der für die Darstellung verwendet wird. Im Gruppenfeld "Wertebereich" können Sie die Minimal- und Maximalwerte, die angezeigt werden sollen, definieren. Skalenwerte werden mit (Skalenhauptwerte) oder ohne (Skalenzwischenwerte) den zugehörigen Wert angezeigt; Sie können dafür auch unterschiedliche Farben definieren. Im Gruppenfeld "Farben" können Sie Farben für die Scheibenfüllung, Nadel, Nadelbasis (verbirgt den ersten Teil der Nadel im Zentrum des Diagramms) und die Umrandung des Diagramms definieren. Der aktuelle Wert und die zusätzliche Beschriftung können in jedem beliebigen Winkel angezeigt werden.

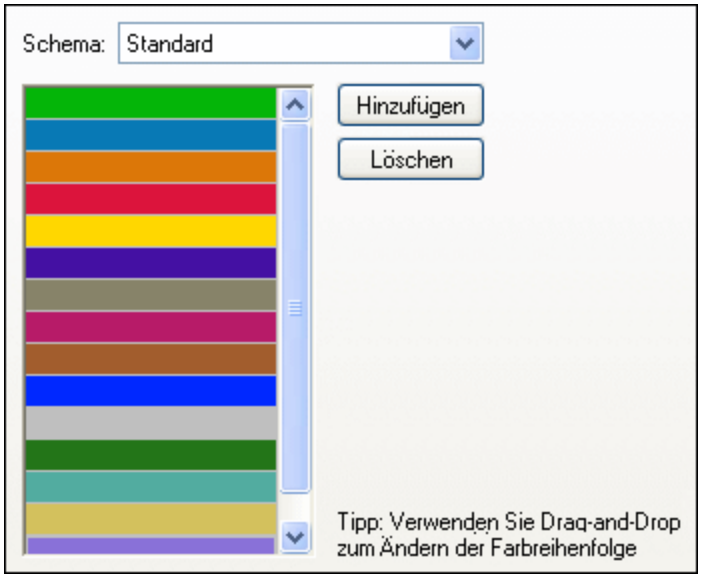
5.12.8.2.3 Farben

Abhängig vom gewählten Diagrammtyp gibt es in XMLSpy zwei verschiedene Abschnitte für die Definition der in Diagrammen verwendeten Farben:

- Farbschema für Kreis-, Balken-, Linien-, Flächen- und Kerzendiagramme
- Farbbereich für Einzelwertdiagramme

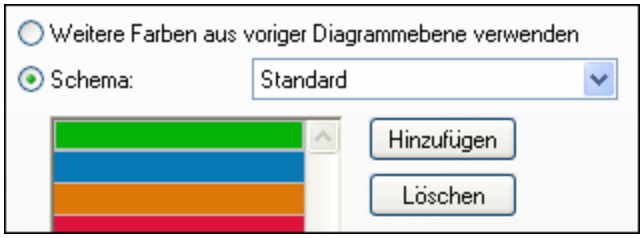
Farbschema

Im Abschnitt "Farbschema" des Dialogfelds Darstellung ändern gibt es vier vordefinierte Farbschemas (Standard, Grauskala, Bunt und Pastell), die Sie anpassen können; darüber hinaus können Sie Ihr eigenes Farbschema auch komplett neu definieren.



Die oberste Farbe wird für die erste Datenreihe verwendet, dann die zweite Farbe und so weiter. Sie können die Reihenfolge der Farben verändern, indem Sie eine Farbe markieren und sie mit der Maus an die neue Position ziehen. Um eine neue Farbe hinzuzufügen oder eine unerwünschte Farbe zu löschen, klicken Sie auf die entsprechende Schaltfläche. In Kerzendiagrammen wird nur die erste Farbe verwendet.

Wenn Sie eine oder mehrere Ebenen von Overlay-Diagrammen zu einem Diagrammfenster hinzugefügt haben, enthält der Abschnitt "Farbschema" des Dialogfelds Darstellung ändern die zusätzliche Optionsschaltfläche `Weitere Farben aus voriger Diagrammebene verwenden`, die standardmäßig aktiviert ist.



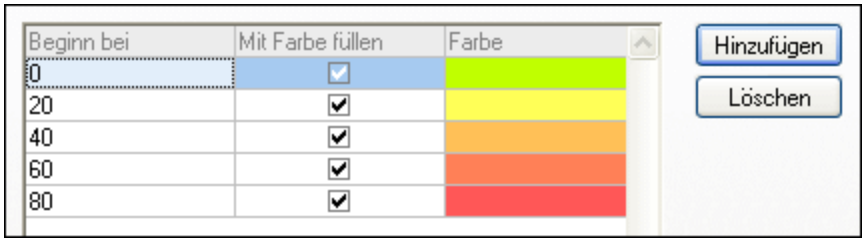
Wenn die Optionsschaltfläche aktiviert ist, wird das Farbschema der vorherigen Ebene benutzt und Sie können für das Overlay kein eigenes Farbschema auswählen. Die Datenreihen der aktiven Ebene werden mit Hilfe der nachfolgenden Farben des Farbschemas aus der vorigen Ebene dargestellt. Auf diese Weise haben alle Datenreihen im Diagrammfenster unterschiedliche Farben und können daher leichter unterschieden werden.

Sie können diese Verbindung in jeder beliebigen Ebene aufheben und ein anderes Farbschema wählen, das dann in den nachfolgenden Ebenen wiederverwendet werden kann.

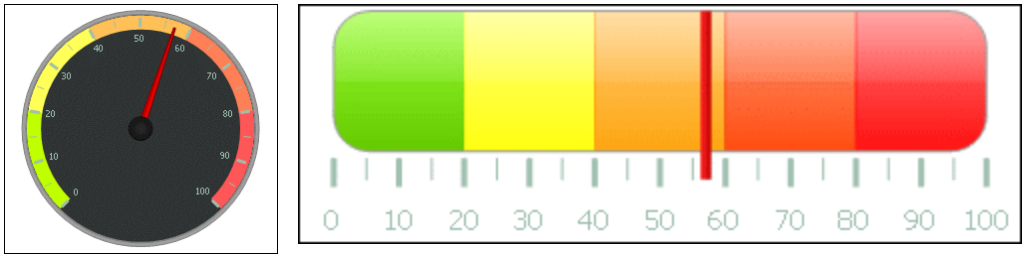
© 2016-2022 Altova GmbH

Farbbereich

In Einzelwertdiagrammen können Sie die Darstellung des Diagramms anpassen, indem Sie bestimmten Wertebereichen Farben zuweisen.

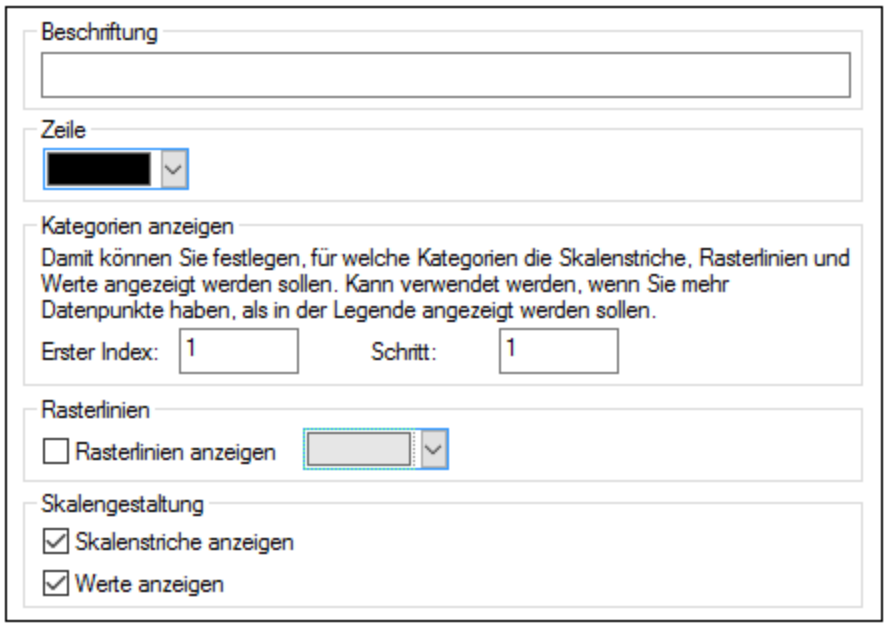


Die Definition, die in der oben stehenden Abbildung gezeigt wird, sieht in den Einzelwertdiagrammen folgendermaßen aus:

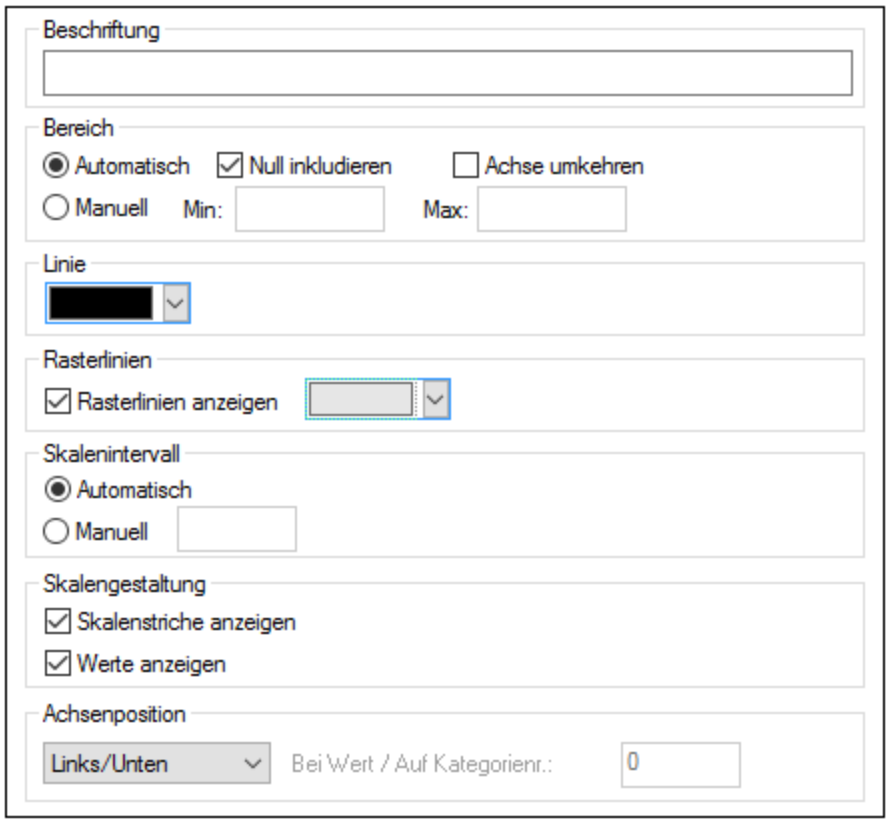


5.12.8.2.4 X-Achse

Im Abschnitt "X-Achse" des Dialogfelds Darstellung ändern können Sie eine Beschriftung für die Achse eingeben und Farben für die Achse und die Rasterlinien (falls angezeigt) definieren. Sie können auch festlegen ob Sie Skalenstriche und Skalenwerte anzeigen möchten oder nicht. Dieser Abschnitt ist für alle Balken-, Linien-, Flächen- und Kerzendiagramme gleich. Mit Hilfe der Option *Kategorien anzeigen* können Sie festlegen, dass nur eine Untermenge aller Kategorien (Werte der X-Achse) angezeigt wird, d.h. es werden nur die Skalenstriche, Rasterlinien und Werte der ausgewählten Kategorien angezeigt. Erstellen Sie die Untermenge der anzuzeigenden Kategorien durch Eingabe (i) des Index des ersten anzuzeigenden Werts und (ii) der Anzahl der Indizes, die übersprungen werden sollen. Wenn es z.B. 101 Kategorien gibt: `1900, 1901, 1902 ... 1999, 2000`, so können Sie festlegen, dass jedes zehnte Jahr von `1900` bis `2000` angezeigt wird, indem Sie den Wert *Erster Index* auf `1` und *Schritt* auf `10` setzen.



In Werte-Liniendiagrammen können Sie auch den Wertebereich definieren und entscheiden, in welchem Intervall Skalenstriche angezeigt werden sollen.



© 2016-2022 Altova GmbH

Beschriftung
Der Text, den Sie in das Feld "Beschriftung" eingeben, wird unterhalb der Achse als Beschreibung der x-Achse angezeigt.

Bereich
Standardmäßig ist im Gruppenfeld "Bereich" die Optionsschaltfläche `Auto` ausgewählt. Wenn Sie einen Ausschnitt des Diagramms mehr im Detail zeigen möchten, aktivieren Sie die Optionsschaltfläche `Manuell` und geben Sie in den entsprechenden Feldern einen Minimal- und einen Maximalwert ein. Wenn die Spalte, die für die x-Achse verwendet wird, nicht den Wert Null enthält, können Sie das Kontrollkästchen `Null inkludieren` deaktivieren; die x-Achse beginnt dann mit dem kleinsten Wert, der in der Datenreihe vorhanden ist. Mit der Option `Achse umkehren` können Sie die Werte der x-Achse umkehren. Wenn die Werte z.B. von `0` bis `360` laufen, wird die x-Achse bei Auswahl dieser Option so generiert, dass sich `360` am Ausgangspunkt befindet und sich die Werte die x-Achse entlang fortlaufend bis `0` verringern.

Linie
Die Achse wird in der Farbe angezeigt, die Sie aus der Dropdown-Liste `Linie` auswählen. Sie können eine der vordefinierten Farben verwenden oder auf die Schaltfläche **Andere Farbe...** klicken um eine Standardfarbe zu wählen oder eine eigene Farbe zu definieren. Klicken Sie im Register "Benutzerdef." auf die Schaltfläche **Auswählen** und verwenden Sie die Pipette um eine Farbe auszuwählen, die irgendwo auf Ihrem Schirm angezeigt wird.

Rasterlinien
Wenn das Kontrollkästchen `Rasterlinien anzeigen` aktiviert ist, können Sie aus der zugehörigen Dropdown-Liste eine Farbe auswählen.

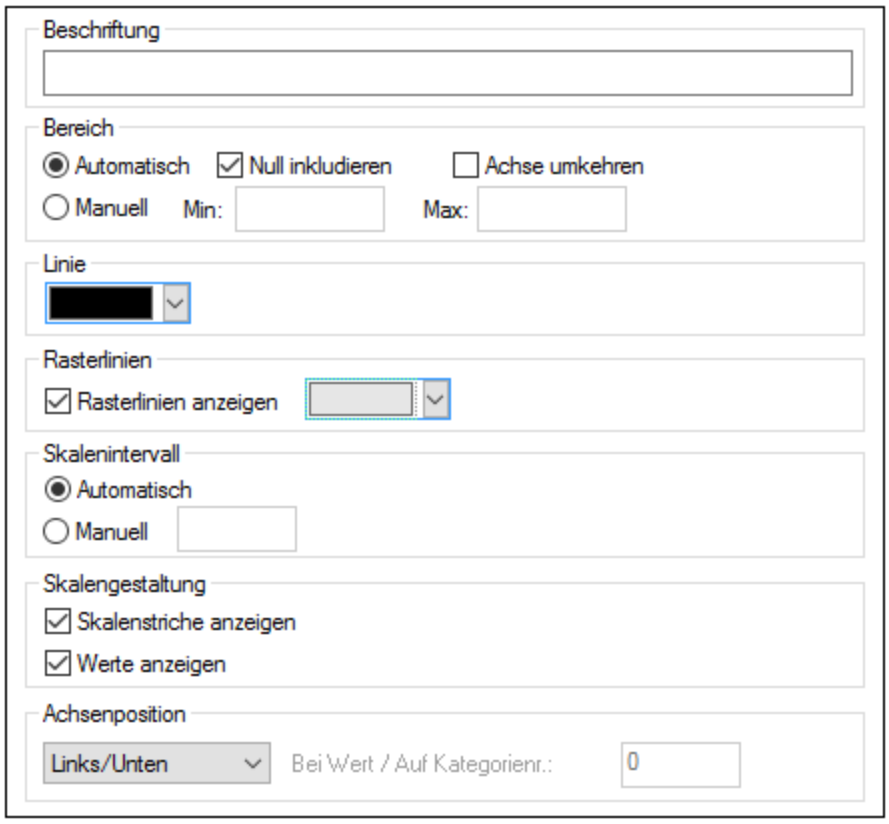
Skalenintervall
Wenn Sie mit den Standardskalenwerten nicht zufrieden sind, können Sie im Gruppenfeld "Skalenintervall" die Optionsschaltfläche `Manuell` aktivieren und den Abstand zwischen den einzelnen Einheiten in das entsprechende Feld eingeben.

Skalengestaltung
Sie können die Anzeige von Skalenstrichen auf der Achse und/oder Achsenwerten ein- und ausschalten.

Achsenposition
Im Dropdown-Listenfeld können Sie die Position wählen, an der die Achse angezeigt werden soll. Wenn Sie "Bei Wert / Auf Kategorienummer" wählen, können Sie die Achse auch an einer beliebigen Stelle der Fläche positionieren.

5.12.8.2.5 Y-Achse

Im Abschnitt "Y-Achse" des Dialogfelds Darstellung ändern können Sie eine Beschriftung für die Achse eingeben, Farben für die Achse und die Rasterlinien (falls angezeigt) definieren, den Wertebereich definieren und entscheiden, ob und wo Skalenstriche angezeigt werden sollen und ob Sie die Achsenwerte anzeigen möchten oder nicht. Dieser Abschnitt ist für alle Balkendiagramme und Liniendiagramme gleich.



Beschriftung
Der Text, den Sie in das Feld "Beschriftung" eingeben, wird links neben der Achse als Beschreibung der y-Achse angezeigt.

Bereich
Standardmäßig ist im Gruppenfeld "Bereich" die Optionsschaltfläche `Auto` ausgewählt. Wenn Sie einen Ausschnitt des Diagramms mehr im Detail zeigen möchten, aktivieren Sie die Optionsschaltfläche `Manuell` und geben Sie in den entsprechenden Feldern einen Minimal- und einen Maximalwert ein. Wenn die Spalte, die für die y-Achse verwendet wird, nicht den Wert Null enthält, können Sie das Kontrollkästchen `Null inkludieren` deaktivieren; die y-Achse beginnt dann mit dem kleinsten Wert, der in der Datenreihe vorhanden ist. Mit der Option `Achse umkehren` können Sie die Werte der y-Achse umkehren. Wenn die Werte z.B. von `0` bis `360` laufen, wird die y-Achse bei Auswahl dieser Option so generiert, dass sich `360` am Ausgangspunkt befindet und sich die Werte die y-Achse hoch fortlaufend bis `0` verringern.

Linie
Die Achse wird in der Farbe angezeigt, die Sie aus der Dropdown-Liste `Linie` auswählen. Sie können eine der vordefinierten Farben verwenden oder auf die Schaltfläche **Andere Farbe...** klicken um eine Standardfarbe zu wählen oder eine eigene Farbe zu definieren. Klicken Sie im Register "Benutzerdef." auf die Schaltfläche **Auswählen** und verwenden Sie die Pipette um eine Farbe auszuwählen, die irgendwo auf Ihrem Schirm angezeigt wird.

Rasterlinien
Wenn das Kontrollkästchen `Rasterlinien anzeigen` aktiviert ist, können Sie aus der zugehörigen Dropdown-Liste eine Farbe auswählen.

© 2016-2022 Altova GmbH

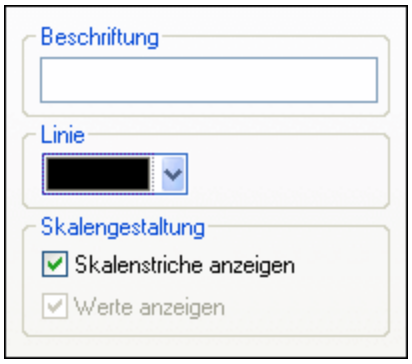
Skalenintervall
Wenn Sie mit den Standardskalenwerten nicht zufrieden sind, können Sie im Gruppenfeld "Skalenintervall" die Optionsschaltfläche `Manuell` aktivieren und den Abstand zwischen den einzelnen Einheiten in das entsprechende Feld eingeben.

Skalengestaltung
Sie können die Anzeige von Skalenstrichen auf der Achse und/oder Achsenwerten ein- und ausschalten.

Achsenposition
Im Dropdown-Listenfeld können Sie die Position wählen, an der die Achse angezeigt werden soll. Wenn Sie "Bei Wert / Auf Kategorienummer" wählen, können Sie die Achse auch an einer beliebigen Stelle der Fläche positionieren.

5.12.8.2.6 Z-Achse

Im Abschnitt "Z-Achse" des Dialogfelds Darstellung ändern können Sie eine Beschriftung für die Achse eingeben, Farben für die Achse definieren und festlegen, ob Sie auf der Achse Skalenstriche anzeigen möchten oder nicht. Dieser Abschnitt ist für alle 3-D Balkendiagramme (Balkendiagramm 3-D und Gruppiertes Balkendiagramm 3-D) gleich.

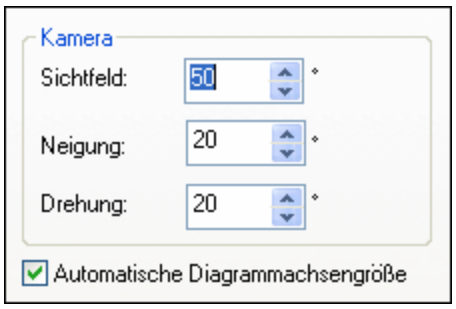


Beschriftung
Der Text, den Sie in das Feld "Beschriftung" eingeben, wird rechts neben der Achse als Beschreibung der z-Achse angezeigt.

Linie
Die Achse wird in der Farbe angezeigt, die Sie aus der Dropdown-Liste `Linie` auswählen. Sie können eine der vordefinierten Farben verwenden oder auf die Schaltfläche **Andere Farbe...** klicken um eine Standardfarbe zu wählen oder eine eigene Farbe zu definieren. Klicken Sie im Register "Benutzerdef." auf die Schaltfläche **Auswählen** und verwenden Sie die Pipette um eine Farbe auszuwählen, die irgendwo auf Ihrem Schirm angezeigt wird.
Skalengestaltung
Sie können die Anzeige von Skalenstrichen auf der Achse ein- und ausschalten.

5.12.8.2.7 3-D Winkel

In 3-D Balkendiagrammen können Sie die im Abschnitt "3-D Winkel" des Dialogfelds "Darstellung ändern" die 3-D Darstellung des Diagramms anpassen.



Durch die Option **Sichtfeld** wird das Diagramm so angezeigt, als würde es aus geringer oder großer Entfernung betrachtet. Gültige Werte sind Werte von 1 bis 120. Je höher der Wert, aus desto größerer Entfernung wird das Diagramm angezeigt.

Der Wert **Neigung** bestimmt die Rotation um die x-Achse, während der Wert **Drehung** die Rotation um die y-Achse definiert. Sie können die Größe der Diagrammachsen automatisch an die Breite des Fensters "Diagramme" anpassen, indem Sie das entsprechende Kontrollkästchen aktivieren.

Wenn das Kontrollkästchen **Automatische Diagrammachsengröße** aktiviert ist, berechnet XMLSpy für die Größe des aktuellen Diagrammfensters automatisch die optimal Größe sowohl der x-Achse als auch der y-Achse. Die Breite und Höhe des Diagramms ändern sich dynamisch wenn Sie die Größe des Fensters "Diagramme" ändern.

5.12.8.2.8 Größen

Im Abschnitt "Größen" des Dialogfelds "Darstellung ändern" können Sie verschiedene Ränder sowie die Größe von Achsen- und Skalenwerten in Einzelwertdiagrammen definieren. Beachten Sie, dass nicht alle Eigenschaften für alle Diagrammtypen zur Verfügung stehen.

Allgemein

Äußerer Rand	Abstand zwischen Fläche und Rand des Diagrammfensters.
Überschrift bis Fläche	Abstand zwischen der Diagrammüberschrift und dem oberen Rand der Fläche.
Legende bis Fläche	Abstand zwischen dem unteren Rand der Fläche und der Legende.

Kreis

Fläche bis Beschriftung	In Kreisdiagrammen, der Abstand zwischen dem äußerst linken bzw. rechten Rand des Kreisdiagramms und seinen Beschriftungen
Höhe Kreisdiagramm	In 3-D Kreisdiagrammen die Höhe der Scheibe.
Schlagschatten Kreisdiagramm	In 3-D Kreisdiagrammen die Länge des Schatten (wenn im Abschnitt "Kreis" aktiviert).

© 2016-2022 Altova GmbH

x-Achse

x-Achse bis Achsenbeschriftung	In Balken- und Liniendiagrammen der Abstand zwischen der x-Achse und ihrer Beschriftung.
x-Achse bis Fläche	In 2-D Balkendiagrammen und Liniendiagrammen der Abstand zwischen der x-Achse und der Fläche.
Skalenstrichgröße x-Achse	In Balken- und Liniendiagrammen die Länge der Skalenstriche auf der x-Achse.

y-Achse

y-Achse bis Achsenbeschriftung	In Balken- und Liniendiagrammen der Abstand zwischen der y-Achse und ihrer Beschriftung
y-Achse bis Fläche	In 2-D Balkendiagrammen und Liniendiagrammen der Abstand zwischen der y-Achse und der Fläche.
Skalenstrichgröße y-Achse	In Balken- und Liniendiagrammen die Länge der Skalenstriche auf der y-Achse.

z-Achse

z-Achse bis Achsenbeschriftung	In 3-D Balkendiagrammen der Abstand zwischen der z-Achse und ihrer Beschriftung.
Skalenstrichgröße z-Achse	In 3-D Balkendiagrammen die Länge der Skalenstriche auf der z-Achse.

Linienführung

Größe der Verbindungspunkte	In Liniendiagrammen die Größe der Quadrate, die die Werte im Diagramm markieren.
Linienbreite	Die Linienbreite in Liniendiagrammen

3-D Achsengrößen

Verhältnis der manuellen x-Achse zur Basis	Definiert in 3-D Balkendiagrammen das Verhältnis zwischen der Länge der x-Achse und der Größe des Diagrammfensters. Bitte beachten Sie, dass das Kontrollkästchen `Automatische Diagrammachsengröße` im Abschnitt "3-D-Winkel" deaktiviert sein muss, da sonst die Größe dennoch automatisch berechnet wird.
Verhältnis der manuellen y-Achse zur Basis	Definiert in 3-D Balkendiagrammen das Verhältnis zwischen der Länge der y-Achse und der Größe des Diagrammfensters. Bitte beachten Sie, dass das Kontrollkästchen `Automatische Diagrammachsengröße` im Abschnitt "3-D-Winkel" deaktiviert sein muss, da sonst die Größe dennoch automatisch berechnet wird.
z-Achse Datenreihenabstand	In 3-D Balkendiagrammen der Abstand zwischen den einzelnen Datenreihen auf der z-Achse.

Einzelwert

Breite Umrandung	In Tachometerdiagrammen die Breite der Umrandung rund um das Diagramm.

Skalenstriche

Abstand zwischen Umrandung und Skalenstrich	In Tachometerdiagrammen der Abstand zischen dem inneren Rand der Umrandung und den Skalenstrichen, die die Werte markieren.

Länge Skalenhauptwerte	In Tachometerdiagrammen die Länge der Skalenhauptwerte (d.h. der Skalenstriche mit Beschriftung).
Breite Skalenhauptwerte	In Tachometerdiagrammen die Breite der Skalenhauptwerte (d.h. der Skalenstriche mit Beschriftung).
Länge Skalenzwischenwerte	In Tachometerdiagrammen die Länge von Skalenstrichen für die kein Wert angezeigt wird.
Breite Skalenzwischenwerte	In Tachometerdiagrammen die Breite von Skalenstrichen für die kein Wert angezeigt wird.

Anzeigenadel

Nadellänge	In Tachometerdiagrammen die Länge der Nadel. (Beachten Sie, dass der Prozentsatz auf dem Durchmesser der Scheibe berechnet wird, wenn Sie also einen Wert wählen, der größer als 50% ist, zeigt die Nadel aus der Scheibe heraus!)
Nadelbreite bei Basis	In Tachometerdiagrammen die Breite der Nadel im Zentrum der Scheibe.
Radius Nadelbasis	In Tachometerdiagrammen der Radius der Basis, die das Zentrum der Scheibe abdeckt.

Farbbereich

Anzeige

Abstand zwischen Umrandung und Farbbereich	In Tachometerdiagrammen der Abstand zwischen dem inneren Rand der Umrandung und dem äußeren Rand des Farbbereichs[395].
Breite Farbbereich	In Tachometerdiagrammen die Breite des anpassbaren Farbbereichs. (Beachten Sie, dass der Prozentsatz vom Durchmesser der Scheibe berechnet wird!)

Skalenwert

Abstand zur Mitte	Abstand von der Mitte der Stelle, an der der Einzelwert angezeigt wird.

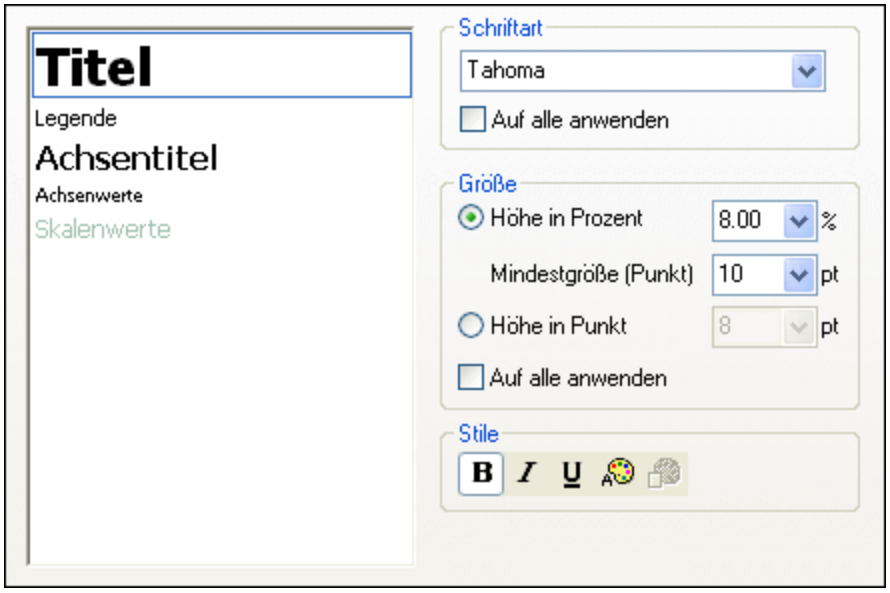
Zusätzlicher Wert

Abstand zur Mitte	Abstand von der Mitte der Stelle, an der die (in den Einzelwertdiagrammeinstellungen[389] definierte) zusätzliche Beschriftung angezeigt wird.

5.12.8.2.9 Schriftarten

Im Abschnitt "Schriftarten" des Dialogfelds Darstellung ändern können Sie die Schriftarten für Objekte im Fenster "Diagramme" konfigurieren.

© 2016-2022 Altova GmbH



Schrifteinstellungen

Sie können für die einzelnen im Fenster "Diagramme" angezeigten Elemente die Schriftart, Größe und den Stil wählen. Die Größe kann als Prozentsatz der Diagrammgröße mit einer Mindestgröße in Punkt definiert, oder als absoluter Wert (in Punkt) angegeben werden. Um die gleiche Schriftart und/oder Größe auf alle Textelemente anzuwenden, aktivieren Sie das entsprechende Kontrollkästchen `Auf alle anwenden`.
Die Elementnamen im Listenfeld sind wie folgt definiert:

- **Titel**: Der Name eines Diagramms
- **Legende**: Der Schlüssel zu den im Diagramm verwendeten Farben
- **Beschriftung**: Die Bezeichnung der Segmente in einem Kreisdiagramm
- **Achsentitel**: Der Name der x-, y- und z-Achse in einem Balken- oder Liniendiagramm
- **Achsenwerte**: Die Einheiten, die in einem Balken- oder Liniendiagramm auf den Achsen angezeigt werden
- **Skalenwerte**: Die Einheiten, die in einem Einzelwertdiagramm angezeigt werden
- **Werte**: Die auf den Balken eines Balkendiagramms dargestellten Werte

5.12.9 Exportieren

Wenn Sie auf die Schaltfläche "Exportieren" klicken, stehen die folgenden Optionen zur Verfügung:

- *Speichern des Diagramms als Bilddatei:* Als Bildformate stehen die Formate PNG, GIF, BMP und JPG zur Verfügung.
- *Kopieren des Bilds in der aktuellen oder einer anderen Größe in die Zwischenablage:* Dadurch kann das Diagramm später aus der Zwischenablage in einen Bericht in einer anderen Applikation kopiert werden.
- *Drucken des Diagramms:* Sendet das Bild an einen Drucker in Ihrem Netzwerk. Die Höhe und Breite des Bilds kann als Prozentwert der Seitengröße definiert werden.

- *Kopieren von XSLT- oder XQuery-Code in die Zwischenablage:* Erstellt ein XSLT- oder XQuery-Fragment. Beide entsprechen im Wesentlichen der Altova Erweiterungsfunktion "`CreateChart`". Diese Funktion kann mit anderen Altova Erweiterungsfunktionen verwendet und mit XMLSpy zur Generierung von Diagrammen verwendet werden. Als Hilfe für die Verwendung der Erweiterungsfunktion `CreateChart` wird mit jedem Fragment auch ein auskommentiertes Verwendungsbeispiel erstellt.

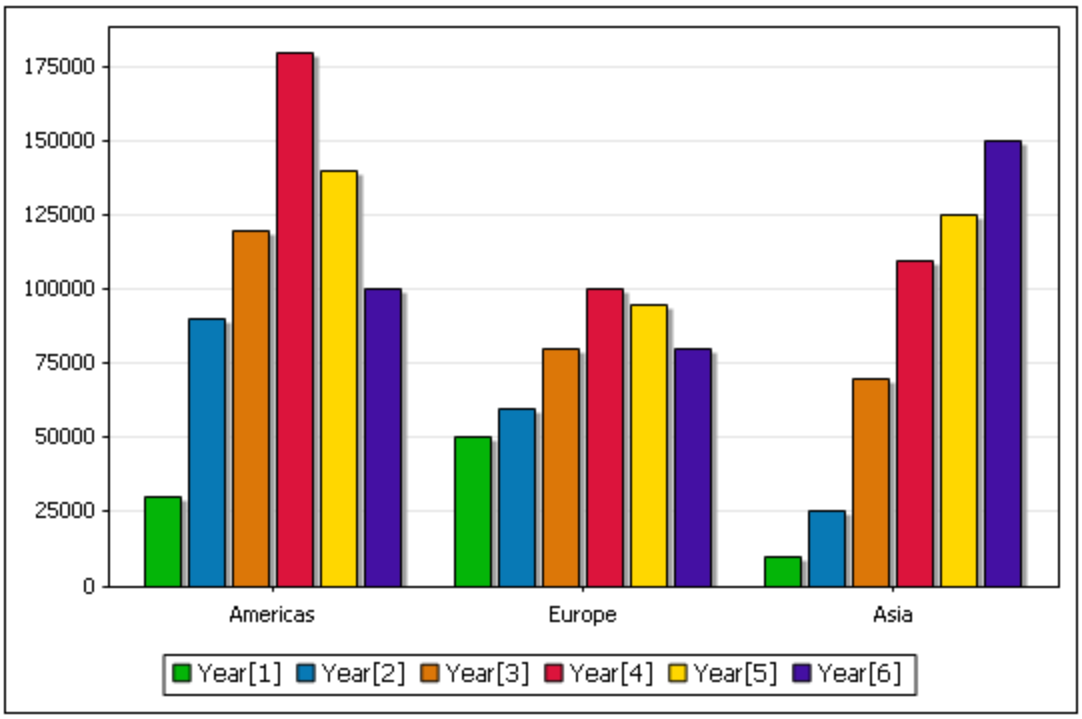
5.12.10 Diagrammbeispiel: Einfach

Angenommen, Sie haben das folgende XML-Dokument. (Es hat den Namen `YearlySales.xml` und befindet sich im Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\Tutorial`.)

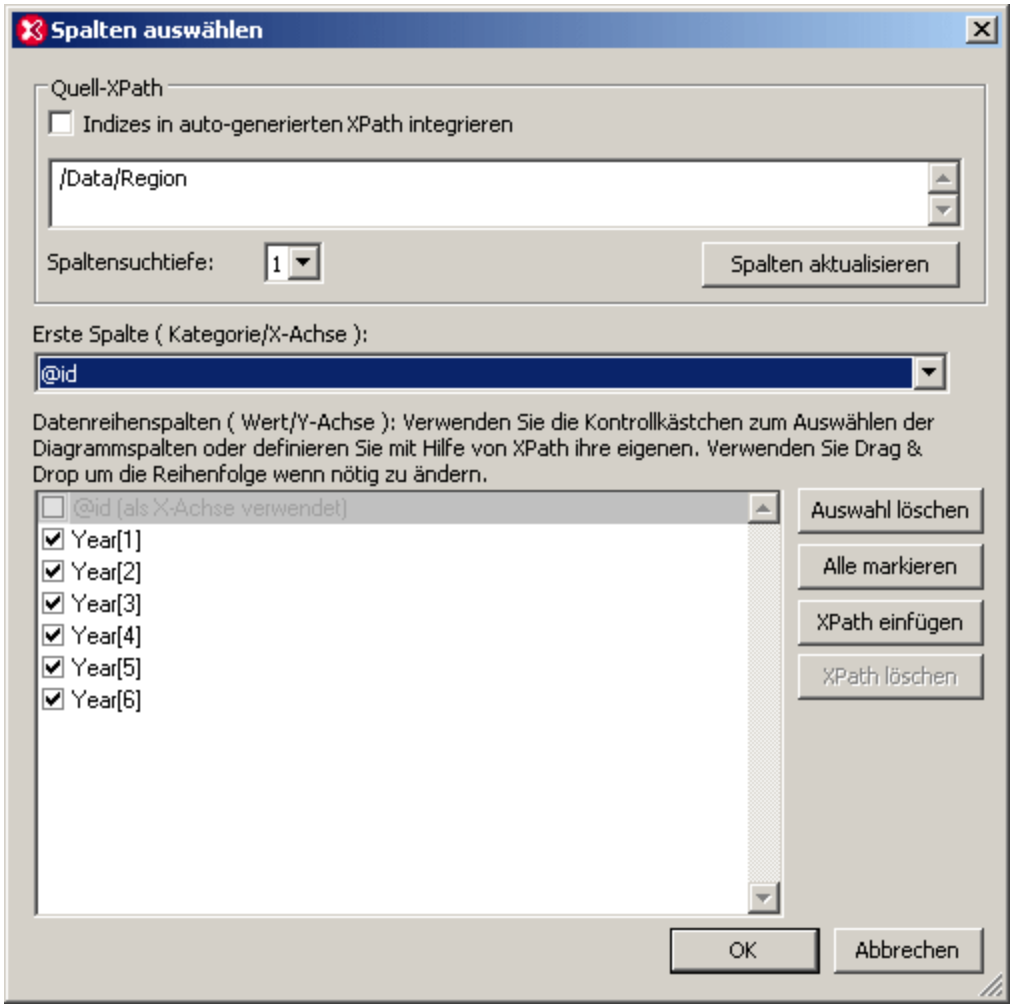
```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
       <Region id="Americas">
              <Year id="2005">30000</Year>
              <Year id="2006">90000</Year>
              <Year id="2007">120000</Year>
              <Year id="2008">180000</Year>
              <Year id="2009">140000</Year>
              <Year id="2010">100000</Year>
       </Region>
       <Region id="Europe">
              <Year id="2005">50000</Year>
              <Year id="2006">60000</Year>
              <Year id="2007">80000</Year>
              <Year id="2008">100000</Year>
              <Year id="2009">95000</Year>
              <Year id="2010">80000</Year>
       </Region>
       <Region id="Asia">
              <Year id="2005">10000</Year>
              <Year id="2006">25000</Year>
              <Year id="2007">70000</Year>
              <Year id="2008">110000</Year>
              <Year id="2009">125000</Year>
              <Year id="2010">150000</Year>
       </Region>
</Data>
```

Ziel ist die Erstellung eines Diagramms, in dem die drei Regionen auf der X-Achse dargestellt werden, wobei für jede Region der Jahresumsatz dargestellt wird. Unser Diagramm sollte in etwa wie das unten gezeigte Balkendiagramm aussehen.

© 2016-2022 Altova GmbH



Dies ist ein einfaches Diagramm, weil wir das Element `Region` als Quell-XPath auswählen können. Der Quell-XPath-Ausdruck gibt eine Sequenz von drei Elementen zurück: die drei `Region` Elemente. Jedes `Region` Element dient wiederum als Kontext-Node für die Auswahl der Daten für die X- und die Y-Achse.



Als Datenreihe möchten wir die `Year` Elemente der einzelnen Regionen verwenden, daher genügt eine Suchtiefe von einer Ebene. Für die X-Achse wählen wir das Attribut `id` des Elements `Region`, folglich werden die Werte des Attributs `id` als Beschriftungen der drei Skalenstriche der X-Achse verwendet. Es werden alle `Year` Datenreihen ausgewählt, da alle `Year` Elemente in der Diagrammdatentabelle enthalten sein sollen.

Wenn Sie auf **OK** klicken, wird das gewünschte Diagramm generiert. Eine Beschreibung komplexerer Diagramme finden Sie im Abschnitt Diagrammbeispiel. Komplex 406.

5.12.11 Diagrammbeispiel: Komplex

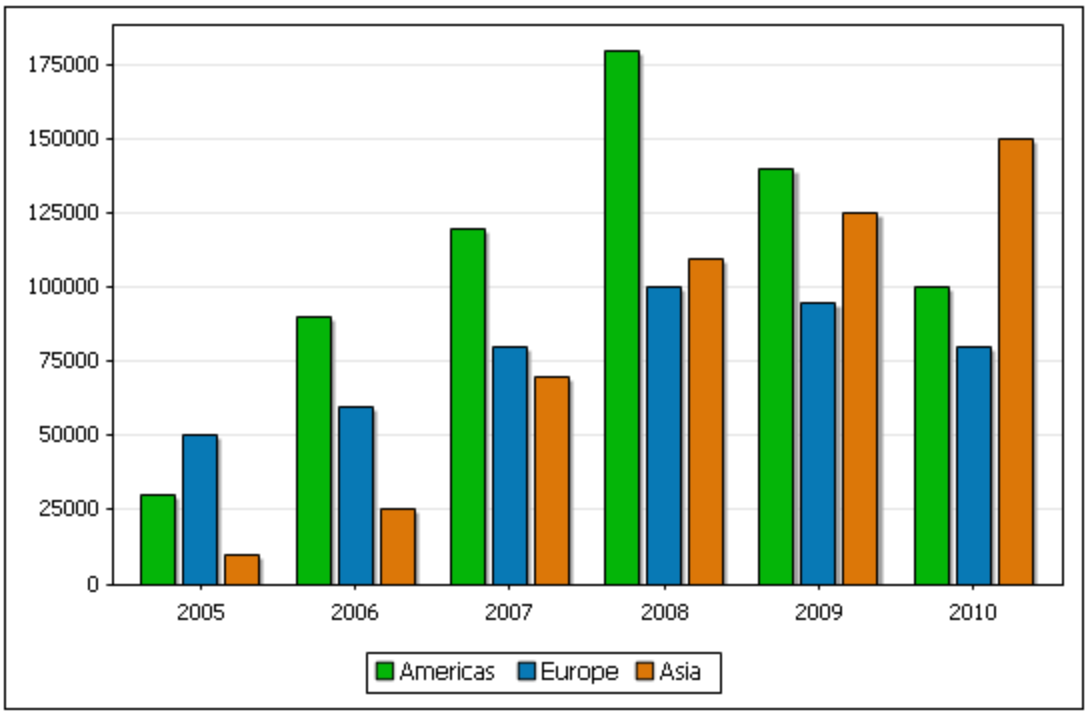
Angenommen, Sie haben das folgende XML-Dokument. (Es hat den Namen `YearlySales.xml` und befindet sich im Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\Tutorial`.)

```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
       <Region id="Americas">
```


© 2016-2022 Altova GmbH

```
            <Year id="2005">30000</Year>
            <Year id="2006">90000</Year>
            <Year id="2007">120000</Year>
            <Year id="2008">180000</Year>
            <Year id="2009">140000</Year>
            <Year id="2010">100000</Year>
      </Region>
      <Region id="Europe">
            <Year id="2005">50000</Year>
            <Year id="2006">60000</Year>
            <Year id="2007">80000</Year>
            <Year id="2008">100000</Year>
            <Year id="2009">95000</Year>
            <Year id="2010">80000</Year>
      </Region>
      <Region id="Asia">
            <Year id="2005">10000</Year>
            <Year id="2006">25000</Year>
            <Year id="2007">70000</Year>
            <Year id="2008">110000</Year>
            <Year id="2009">125000</Year>
            <Year id="2010">150000</Year>
      </Region>
</Data>
```

Ziel ist die Erstellung eines Diagramms, in dem die Jahre auf der X-Achse dargestellt werden, um die regionalen Umsätze für jedes Jahr miteinander vergleichen zu können. Unser Diagramm sollte in etwa wie das unten gezeigte Balkendiagramm aussehen.



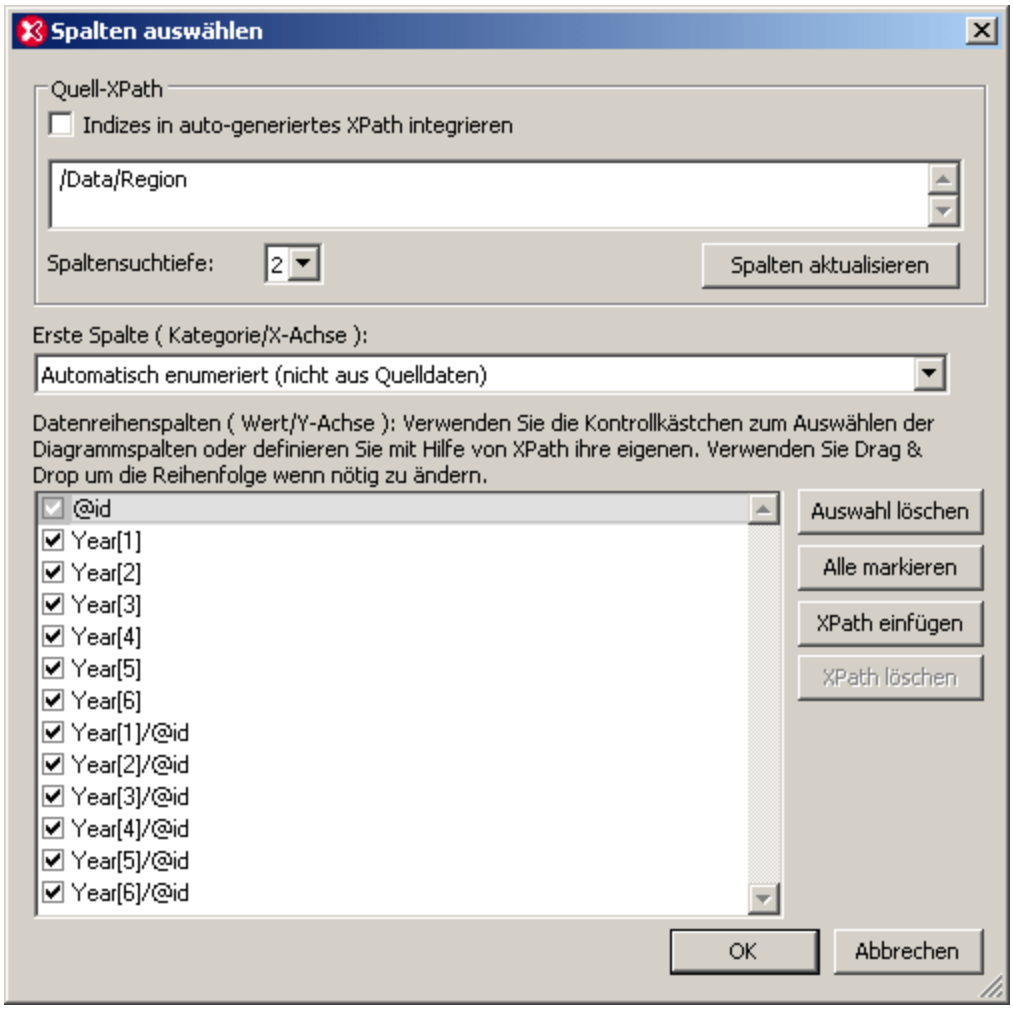
Im Folgenden werden zwei Methoden zur Erstellung dieses Diagramms beschrieben. Anhand dieser beiden Methoden wird gezeigt, wie die verschiedenen Datenauswahlparameter miteinander kombiniert werden können, um das gewünschte Ergebnis zu erhalten.

Methode 1: Ändern der Achsenwerte

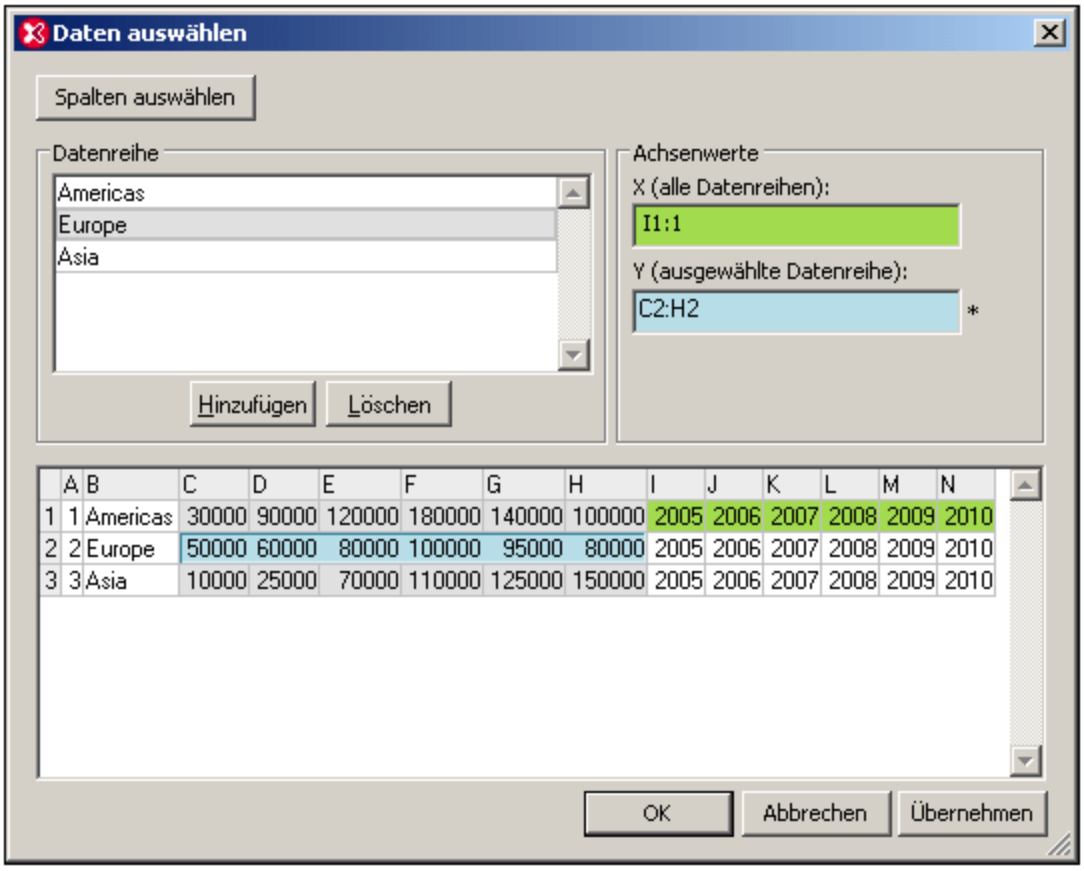
In der ersten Methode werden die Achsen, die im Dialogfeld "Spalten auswählen" ausgewählt wurden, im Dialogfeld "Daten auswählen" geändert. Sie müssen dazu nur sicherstellen, dass alle benötigten Daten im Dialogfeld "Daten auswählen" in der Diagrammdatentabelle zur Verfügung stehen.

1. Stellen Sie im Dialogfeld "Spalten auswählen" sicher, dass alle benötigten Nodes für die Auswahl der Daten für die X- und die Y-Achse zur Verfügung stehen. Beachten Sie, dass die Spaltensuchtiefe in der Abbildung unten auf 2 gesetzt wurde, damit auch die `Year/@id` Attribute ausgewählt werden.

© 2016-2022 Altova GmbH



2. Die Diagrammdatentabelle im Dialogfeld "Daten auswählen" (*Abbildung unten*), enthält die folgenden Spalten: Die erste Spalte ist die Auswahl für die X-Achse (dies ist die Auswahl "Automatisch enumeriert"). Die restlichen Spalten sind die Spalten für die Datenreihen- oder Y-Achse. Dabei handelt es sich um die `Region/@id`-Attribute, den Inhalt des Elements `Year` und die `Year/@id`-Attribute. Beachten Sie außerdem, (i) dass es nur drei Zeilen gibt, daher auch drei Skalenstriche auf der X-Achse; (Wir benötigen aber sechs Skalenstriche auf der X-Achse für die sechs Jahre); und (ii) dass es 13 Datenreihenspalten gbit.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	1	Americas	30000	90000	120000	180000	140000	100000	2005	2006	2007	2008	2009	2010
2	2	Europe	50000	60000	80000	100000	95000	80000	2005	2006	2007	2008	2009	2010
3	3	Asia	10000	25000	70000	110000	125000	150000	2005	2006	2007	2008	2009	2010

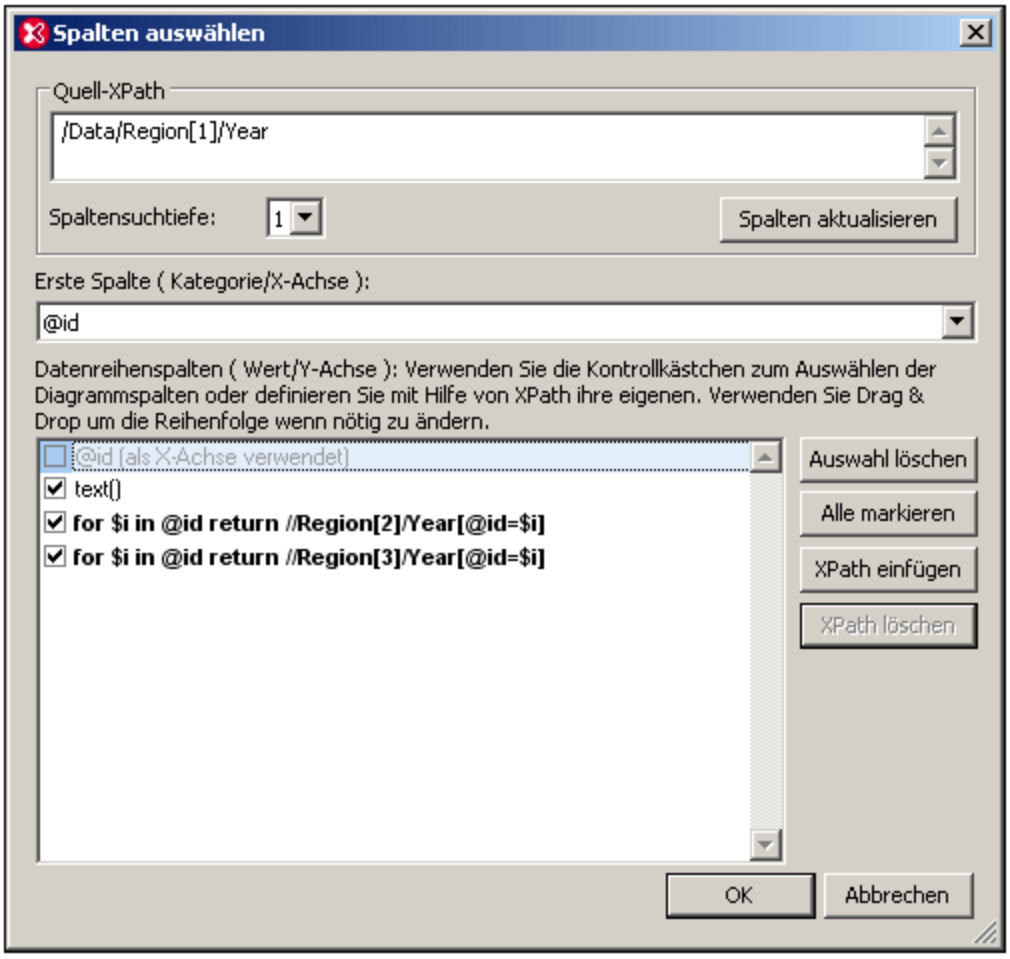
3. Wir löschen im Bereich "Datenreihen" beliebige 10 der 13 Datenreihenzeilen und benennen die verbleibenden drei Datenreihen in `Americas`, `Europe` und `Asia` um, wie in der Abbildung oben gezeigt. In derselben Reihenfolge wie hier werden auch die Skalenstriche auf der X-Achse beschriftet.
4. Wählen Sie im Bereich "Datenreihen" die Datenreihe `Americas` aus. Klicken Sie im Bereich "Achsenwerte" in das Feld für die X-Achse, um den Wert ändern zu können. Klicken Sie anschließend in der Diagrammdatentabelle in die Zelle `I1` und ziehen Sie den Mauszeiger auf die Zelle `N1`. Geben Sie im Textfeld für die Y-Achse entweder `C1:H1` ein oder treffen Sie die Auswahl durch Markieren der Felder von `C1` bis `H1`.
5. Wählen Sie für die Datenreihen "Europe" und "Asia" für die Y-Achse die Felder `C2:H2` bzw. `C3:H3` aus. Die Auswahl für die X-Achse kann dieselbe wie die für die Datenreihe `Americas` sein..
6. Klicken Sie auf **OK**. Das gewünschte Diagramm wird generiert.

Anmerkung: Die Anzahl der Skalenstriche für die X-Achse (die standardmäßig durch die Anzahl der Zeilen in der Diagrammdatentabelle definiert ist) wird von drei auf sechs erhöht, da die Anzahl der X-Achsenbeschriftungen sechs beträgt.

Methode 2: Generieren von Datenreihen mit Hilfe von XPath-Ausdrücken

In der zweiten Methode werden zur Generierung der Datenreihen XPath-Ausdrücke eingefügt. Dies ist notwendig, da der Quell-XPath (*siehe Abbildung unten*) nicht die untergeordneten Nodes hat, die für die Datenreihe benötigt werden. Allerdings erstellt der Quell-XPath-Ausdruck (durch Auswahl der `Year` Elemente des ersten `Region` Elements) sechs Skalenstriche auf der X-Achse. Damit mit Hilfe des Prädikats `[1]` das erste `Region` Element ausgewählt wird, muss das Kontrollkästchen "*Indizes....inkludieren*" aktiviert werden und Sie müssen auf die Schaltfläche **Spalten aktualisieren** klicken.

© 2016-2022 Altova GmbH

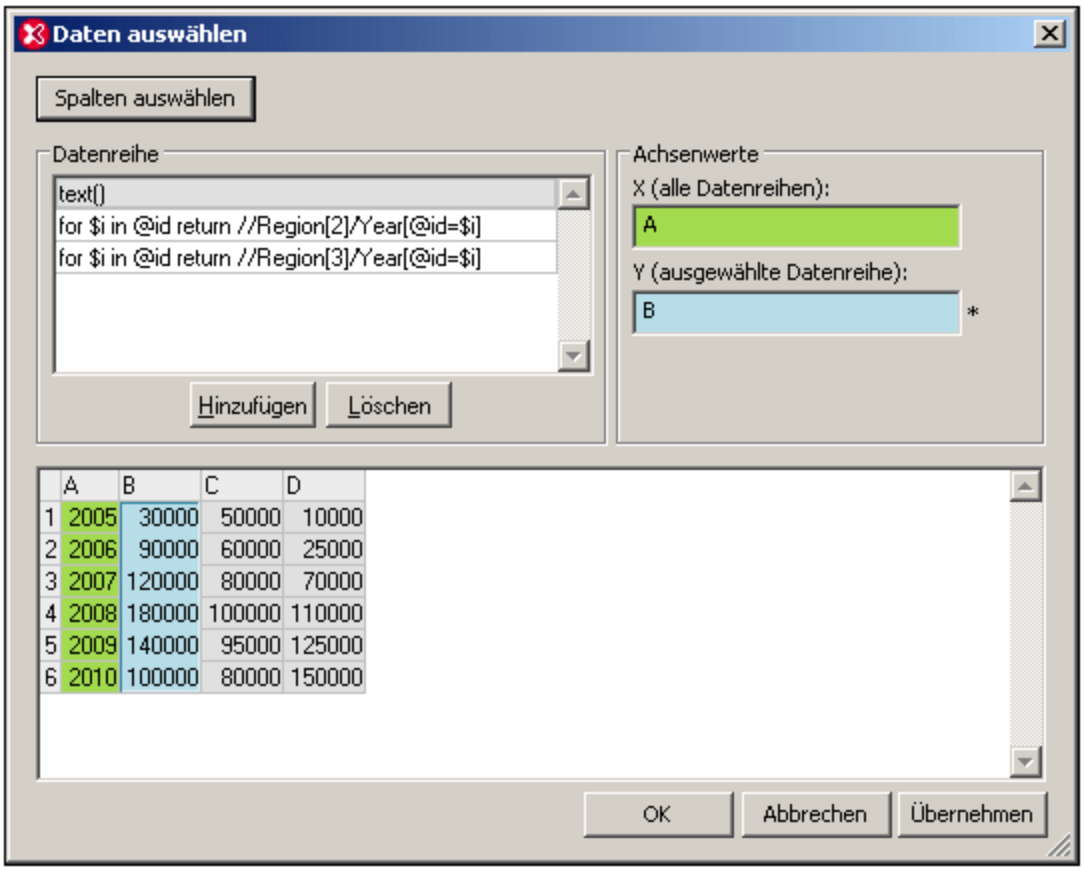


Im Dialogfeld "Spalten auswählen" hat das `Region[1]/Year` Element nur zwei untergeordnete Elemente: `@id` und `text()`. Das `@id` Attribut wird für die X-Achse ausgewählt und dient zur Generierung der richtigen X-Achsenbeschriftung für die sechs Skalenstriche der X-Achse. Die Diagrammdatentabelle würde folgendermaßen ausgewertet werden.

Source XPath	X-Axis	Y-Axis (Series columns)		
`Region[1]/Year[1]`	`@id`	`text()`	`XPath-1`	`XPath-2`
`Region[1]/Year[2]`	`@id`	`text()`	`XPath-1`	`XPath-2`
`Region[1]/Year[3]`	`@id`	`text()`	`XPath-1`	`XPath-2`
`Region[1]/Year[4]`	`@id`	`text()`	`XPath-1`	`XPath-2`
`Region[1]/Year[5]`	`@id`	`text()`	`XPath-1`	`XPath-2`
`Region[1]/Year[6]`	`@id`	`text()`	`XPath-1`	`XPath-2`

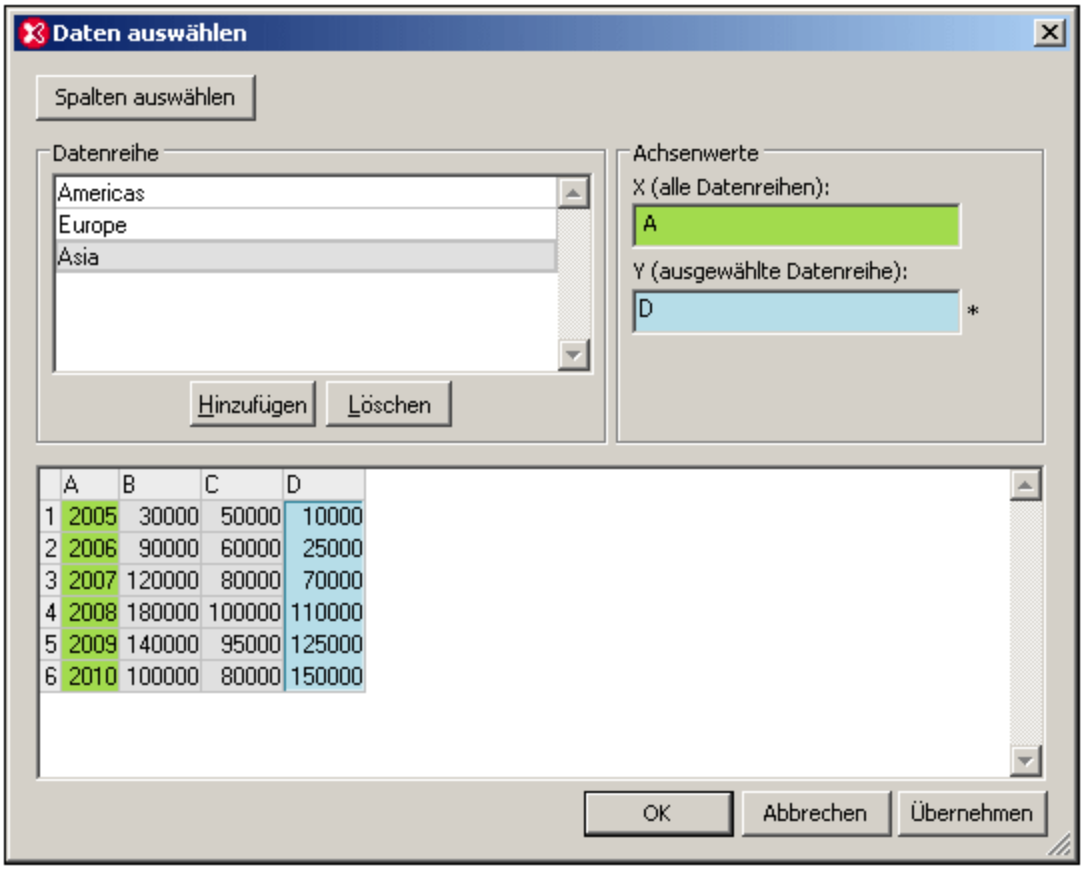
Beachten Sie, dass der Kontext-Node der Reihe nach jedes der sechs `Region[1]/Year` Elemente ist. Der erste XPath-Ausdruck sucht nach dem aktuellen Wert des `Year/@id` Attributs und gibt anschließend das `Region[2]/Year` Element zurück, das denselben `Year/@id` Wert hat, wie der `@id` Wert des aktuellen `Region[1]/Year` Elements. Der zweite XPath-Ausdruck tut dasselbe für die `Region[3]/Year` Elemente. Auf diese Art wird bei jedem der sechs Jahre vorgegangen: Die drei Datenreihen für die Y-Achse sind die Children des jeweiligen Elements `Year` (Der `text()` Node gibt den Inhalt der `Region[1]/Year` Elemente zurück).

Die Diagrammdatentabelle im Dialogfeld "Daten auswählen" würde in etwa folgendermaßen aussehen.



Die Namen der Datenreihen im Dialogfeld "Daten auswählen" können (*wie in der Abbildung unten*) von XPath-Ausdrücken in aussagekräftige Legendentexte geändert werden. Für jede Datenreihe kann im Bereich "Achsenwerte" (durch Klicken auf das Textfeld für die Y-Achse und Auswahl der gewünschten Spalte in der Diagrammdatentabelle) die richtige Datenspalte zugewiesen werden.

© 2016-2022 Altova GmbH



Mit beiden oben gezeigten Methoden werden identische Diagramme erzeugt. Anhand der beiden Beispiele soll gezeigt werden, wie Datenauswahlparameter verwendet werden können.

5.12.12 Diagrammbeispiel: Kerzendiagramm

Kerzendiagramme werden üblicherweise für die Darstellung der Bewegungen von Aktienpreisen auf dem Aktienmarkt verwendet. Es gibt zwei Arten von Kerzendiagrammen:

- Kerzendiagramme mit vier Datenreihen zur Darstellung von Startpreis, Höchststand, Tiefststand und Schlusspreis eines Tages.
- Kerzendiagramme mit drei Datenreihen zur Darstellung von Höchststand, Tiefststand und Schlusspreis eines Tages.

Die Datei `Candlestick.xml`, die im Verzeichnis `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\Tutorial` zur Verfügung steht, ist ein Beispiel einer XML-Dokumentstruktur, die für Kerzendiagramme verwendet werden kann. Die unten stehende Auflistung zeigt die wesentliche Struktur der Datei.

```
<Trades>
  <Stock name="MyStock">
    <Day id="20110103" year="2011" month="Jan" week="01" date="03">
      <Open>90</Open>
      <High>110</High>
      <Low>88</Low>
```

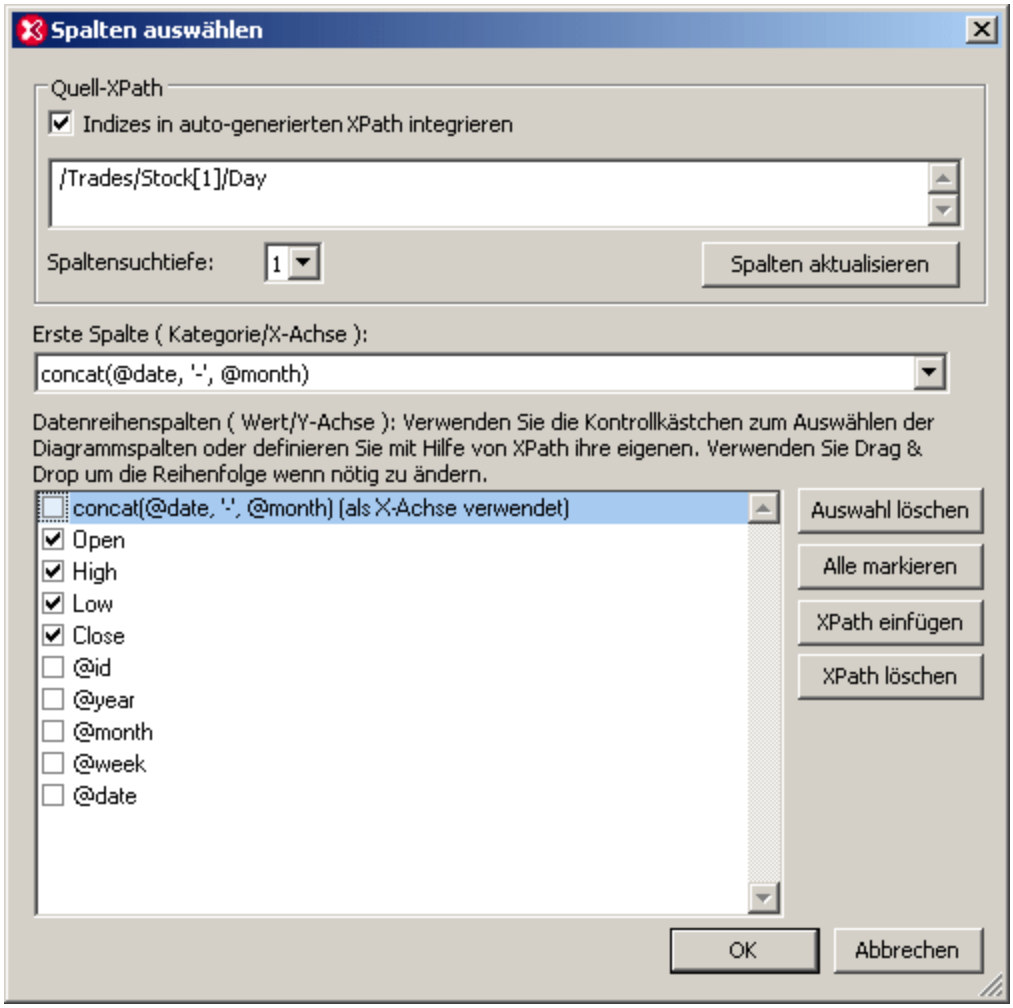
```
      <Close>105</Close>
    </Day>

    .....

  </Stock>
</Trades>
```

Ein Kerzendiagramm für die oben erwähnte Datei kann wie folgt erstellt werden:

1. Platzieren Sie den Mauszeiger in den Tag des Elements "Day" und klicken Sie im Diagrammfenster[350] auf die Schaltfläche **Neues Diagramm**. Dadurch wird das Dialogfeld "Spalten auswählen" (*Abbildung unten*) geöffnet. Falls das Kontrollkästchen *Indizes in auto-generierten XPath integrieren* nicht aktiviert ist, aktivieren Sie es und klicken Sie auf **Spalten aktualisieren**.



2. Klicken Sie auf die Schaltfläche **XPath einfügen** und fügen Sie den XPath-Ausdruck `concat(@date, '-', @month)` ein.
3. Wählen Sie im Listenfeld Erste Spalte (Kategorie/X-Achse) den XPath-Ausdruck, den Sie soeben eingegeben haben. Dadurch wird für jedes Element "Day" als Beschriftung der Skalenstriche auf der X-Achse das Datum und der Monat erstellt.
4. Wählen Sie als Datenreihen der Y-Achse im Bereich "Datenreihenspalten" die Kontrollkästchen Open, High, Low und Close..

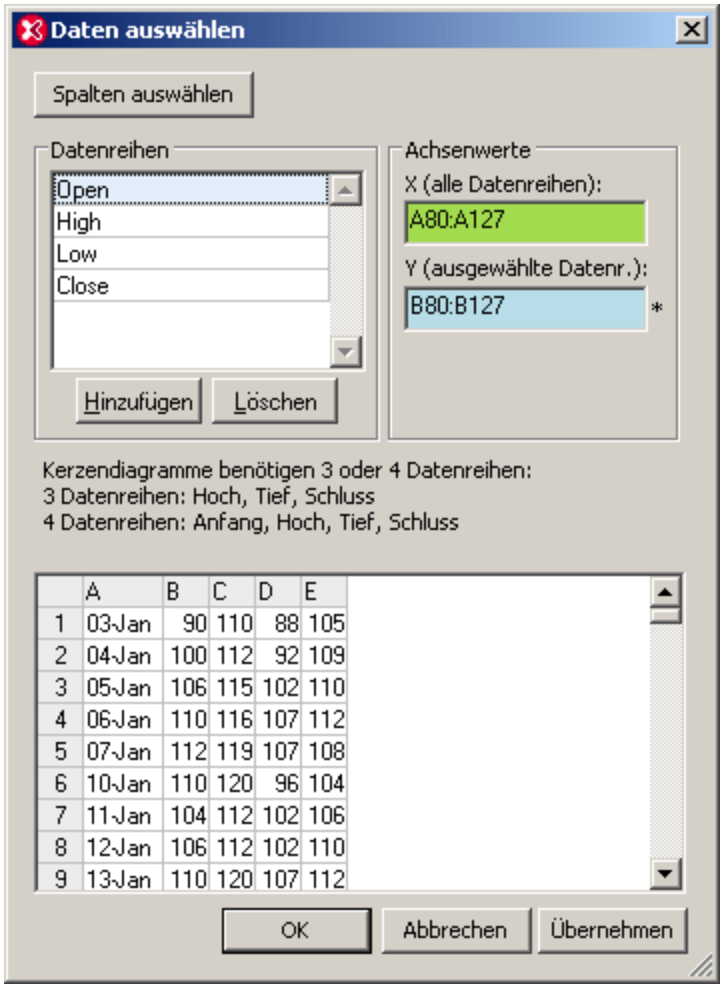
© 2016-2022 Altova GmbH

5. Klicken Sie auf **OK**.
6. Klicken Sie im Diagrammfenster auf die Schaltfläche **Typ ändern** und ändern Sie den Diagrammtyp auf Kerzendiagramm.
7. Klicken Sie auf **OK**. Dadurch wird ein Kerzendiagramm wie in der Abbildung unten erstellt.



Auswählen einer Datenteilmenge

Wenn Sie eine Teilmenge der im Dialogfeld "Daten auswählen" ausgewählten Diagrammdaten anzeigen möchten, zum Beispiel, wenn Sie einen bestimmten Bereich von Daten innerhalb der Diagrammdatenauswahl sehen möchten, klicken Sie im Diagrammfenster auf die Schaltfläche **Daten auswählen**. Dadurch wird das Dialogfeld "Daten auswählen" aufgerufen (*Abbildung unten*).



Geben Sie im Gruppenfeld "Achsenwerte" den Bereich für die X-Achse ein, zum Beispiel, wie in der obigen Abbildung, die Zellen `A80:A127`. Für die einzelnen Datenreihen klicken Sie zuerst in das Textfeld für die Y-Achse, wählen Sie dann die Datenreihe im Gruppenfeld "Datenreihen" und geben Sie den Bereich für die Datenreihe ein. Wiederholen Sie den Vorgang für alle vier Datenreihen. Die obige Abbildung zeigt zum Beispiel den ausgewählten Bereich der Y-Achse für die Datenreihe "Open". Weitere Informationen über das Dialogfeld "Daten auswählen" finden Sie im Abschnitt Diagrammdaten 370.

 © 2016-2022 Altova GmbH

5.13 XML-Signaturen

Eine XML-Datei kann digital signiert werden und die Signatur kann in der Folge überprüft werden. Wenn die Datei geändert wurde, nachdem die Datei signiert wurde, so schlägt die Überprüfung fehl. XMLSpy unterstützt sowohl das Erstellen als auch das Überprüfen von XML-Signaturen.

XML-Signaturen in XMLSpy-Ansichten

XML-Signaturen können für alle Arten von XML-Dateien wie XML-Schemas, WSDL- und XBRL-Dateien erstellt werden. Daher stehen die Befehle **XML | XML-Signatur erstellen** 419 und **XML | XML-Signatur überprüfen** 422 in allen XMLSpy-Ansichten zur Verfügung: in der Textansicht 144, der Grid-Ansicht 169, der Schema-Ansicht 171, der WSDL-Ansicht 253 und der XBRL-Ansicht 266.

Funktionsweise von XML-Signaturen

Der gesamte Vorgang von der Erstellung der Signatur bis zur Überprüfung der Signatur läuft folgendermaßen ab:

1. Die XML-Datei wird entweder mit einem privaten Schlüssel (private key) eines Zertifikats oder mit einem Passwort signiert. Sie können in XMLSpy eine Signatur mit dem Befehl **XML | XML-Signatur erstellen** 419 erstellen. Die Signatur wird durch Verarbeitung des (i) XML-Dokuments und (ii) des privaten Schlüssels eines Zertifikats oder eines Passworts erzeugt.
2. Die Signatur kann entweder in die XML-Datei inkludiert oder in einer separaten Datei gespeichert werden.
3. Die Signatur der XML-Datei wird anhand des öffentlichen Schlüssels (public key) des Zertifikats oder anhand des Passworts überprüft. Dabei wird zuerst (i) das XML-Dokument und (ii) der öffentliche Schlüssel des Zertifikats oder das für die Signatur verwendete Passwort verarbeitet und anschließend wird das Ergebnis mit der Signatur verglichen. Wenn die XML-Datei nach der Signierung geändert wurde, schlägt die Überprüfung fehl. Sie können eine Signatur in XMLSpy mit Hilfe des Befehls **XML | XML-Signatur überprüfen** 422 überprüfen.

Eine genaue Anleitung zur Erstellung und Überprüfung von Signaturen in XMLSpy finden Sie in den Unterabschnitten dieses Abschnitts:

- Erstellen von XML-Signaturen 419
- Überprüfen von XML-Signaturen 422

Verwendung von Zertifikaten in XML-Signaturen

Für die Verwendung mit XML-Signaturen müssen Zertifikate einen privaten und einen öffentlichen Schlüssel haben. Der private Schlüssel dient zur Erstellung der XML-Signatur, mit dem öffentlichen Schlüssel wird die XML-Signatur überprüft.

In einem typischen Szenario hat der Absender eines XML-Dokuments Zugriff auf den privaten Schlüssel eines Zertifikats und erstellt damit die XML-Signatur. Der Empfänger des Dokuments hat Zugriff auf den öffentlichen Schlüssel des Zertifikats. Dies kann auf zwei Arten erfolgen: (i) Der Absender sendet den öffentlichen Schlüssel mit der Signatur mit; (ii) der Empfänger hat Zugriff auf die öffentliche Schlüsselversion des vom Absender verwendeten Zertifikats.

Nähere Informationen zu Zertifikaten finden Sie im Unterabschnitt Arbeiten mit Zertifikaten 425.

Anmerkung: Die XML-Signaturfunktion von XMLSpy unterstützt Zertifikate vom Typ RSA-SHA1, DSA-SHA1 und SHA-256. Für passwortbasierte Signaturen wird ab Version XMLSpy 2018 der HMAC-SHA256-Algorithmus verwendet (siehe *Erstellen von XML-Signaturen*[419]).

Gültigkeit eines XML-Dokuments

Wenn eine XML-Signatur in das XML-Dokument eingebettet wird, so wird ein `Signature` Element aus dem Namespace `http://www.w3.org/2000/09/xmldsig#` zum XML-Dokument hinzugefügt. Damit das Dokument dem Schema gemäß gültig bleibt, muss das Schema die entsprechenden Elementdeklarationen enthalten. XMLSpy bettet Signaturen auf zwei Arten ein:

- *Enveloped:* Das Element `Signature` wird als das letzte Child-Element des Root (Dokument)-Elements erstellt.
- *Enveloping:* Das Element `Signature` wird als das Root- (Dokument)-Element erstellt und das ursprüngliche XML-Dokument wird als Child-Element mit dem Namen `Object` in dieses Element eingefügt.

Wenn Sie das Schema des XML-Dokuments nicht ändern möchten, kann die XML-Signatur in einer externen Datei erstellt werden. Nähere Informationen dazu finden Sie in der Beschreibung zu den Platzierungsoptionen im Abschnitt Erstellen von XML-Signaturen[419].

Unterhalb sehen Sie Auszüge aus XML-Schemas, in denen gezeigt wird, wie das `Signature` Element einer "enveloped" Signatur verwendet werden darf.Sie können diese Beispiele als Richtlinien für die Anpassung Ihrer eigenen Schemas verwenden.

In ersten der beiden Beispiele unten wird das XML-Signaturschema in das Benutzerschema importiert. Das XML-Signatur-Schema befindet sich unter der Internetadresse: http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:xsig="http://www.w3.org/2000/09/xmldsig#"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified">
   <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
              schemaLocation="http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd"/>
   <xs:element name="Root">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="FirstChildOfRoot"/>
            <xs:element ref="SecondChildOfRoot" minOccurs="0"/>
            <xs:element ref="ThirdChildOfRoot" minOccurs="0"/>
            <xs:element ref="xsig:Signature" minOccurs="0"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   ...
</xs:schema>
```

Die zweite Option (*Beispiel unten*) ist, ein allgemeines Wildcard-Element hinzuzufügen, das jedes Element aus anderen Namespaces repräsentieren kann. Wenn Sie das Attribut `processContents` auf `lax` setzen, überspringt der Validator dieses Element, da keine übereinstimmende Elementdeklaration gefunden wird. Folglich muss der Benutzer das XML-Signatur-Schema nicht referenzieren. Der Nachteil diese Option ist

© 2016-2022 Altova GmbH

allerdings, dass an der angegebenen Stelle im XML-Dokument jedes beliebige Element (und nicht nur das Element `Signature`) hinzugefügt werden kann, ohne dass das XML-Dokument ungültig wird.

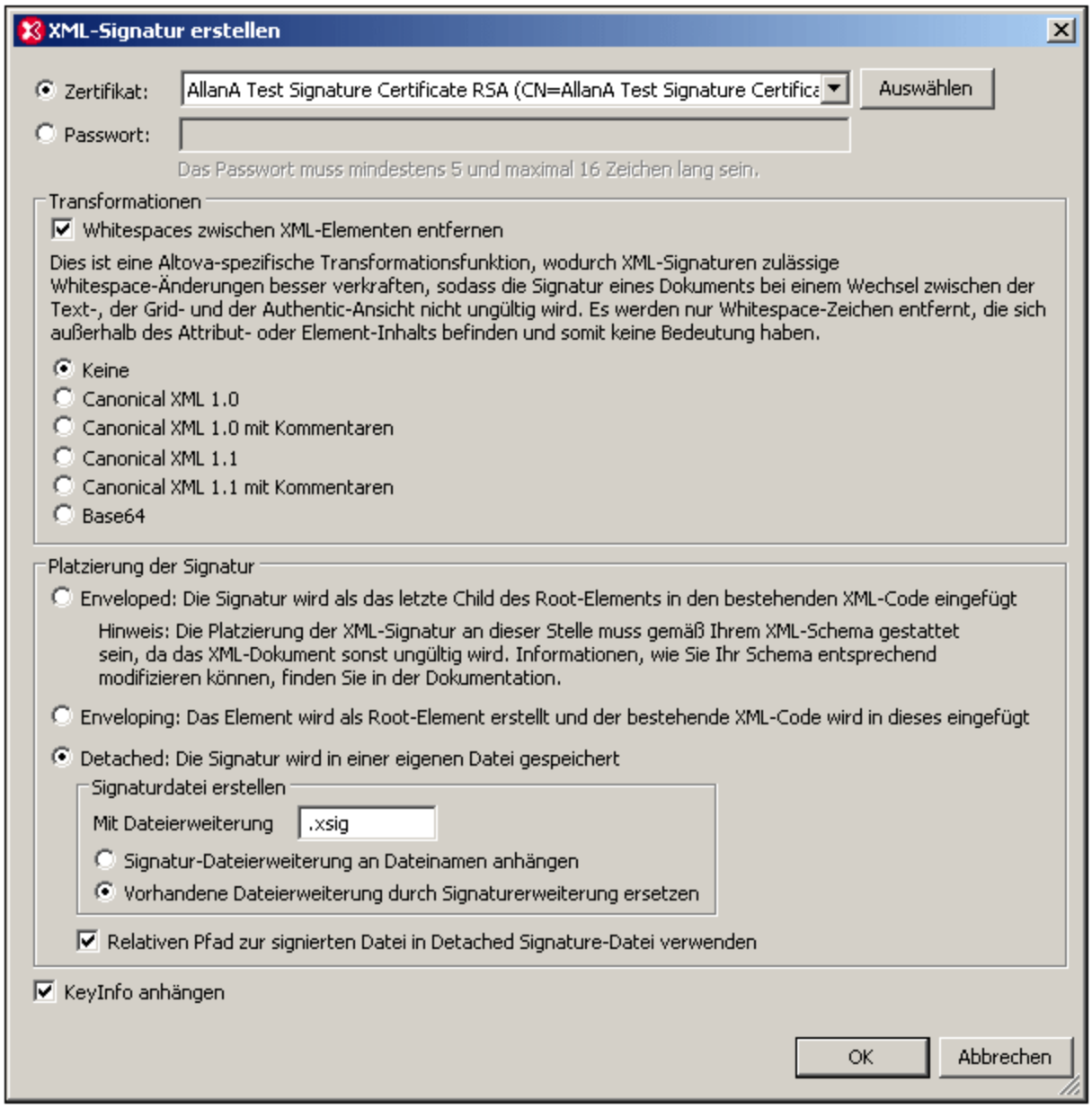
```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified">
   <xs:element name="Root">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="selection"/>
            <xs:element ref="newsitems" minOccurs="0"/>
            <xs:element ref="team" minOccurs="0"/>
            <xs:any namespace="##other" minOccurs="0" processContents="lax"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   ...
</xs:schema>
```

W3C-Spezifikation

Nähere Informationen zu XML-Signaturen finden Sie in der W3C-Spezifikation zu XML-Signaturen unter https://www.w3.org/TR/xmldsig-core1/.

5.13.1 Erstellen von XML-Signaturen

Um eine XML-Signatur für ein XML-Dokument zu erstellen, öffnen Sie das gewünschte XML-Dokument und klicken Sie anschließend auf den Menübefehl **XML | XML-Signatur erstellen**. Daraufhin wird das Dialogfeld "XML-Signatur erstellen" (*Abbildung unten*) geöffnet, dessen Einstellungen im Folgenden erläutert werden.



Authentifizierungsmethode: Zertifikat oder Passwort

Die Signatur kann auf einem Zertifikat oder einem Passwort basieren. Wählen Sie das Optionsfeld für die gewünschte Methode aus.

- *Zertifikat:* Klicken Sie auf die Schaltfläche **Auswählen** und navigieren Sie zum gewünschten Zertifikat. Das ausgewählte Zertifikat muss einen privaten Schlüssel (private key) haben. Die Signatur wird anhand des privaten Schlüssels des Zertifikats generiert. Zur Überprüfung der Signatur muss Zugriff auf das Zertifikat (oder eine Public-Key-Version davon) bestehen. Der öffentliche Schlüssel (public key) des Zertifikats dient zur Überprüfung der Signatur. Nähere Informationen über Zertifikate finden Sie im Abschnitt Arbeiten mit Zertifikaten 425.

 © 2016-2022 Altova GmbH

- *Passwort:* Geben Sie ein Passwort mit einer Länge von fünf bis 16 Zeichen ein. Dieses Passwort wird später benötigt, um die Signatur zu überprüfen.

Anmerkung: Ab XMLSpy 2018 wird zum passwortbasierten Signieren von Dokumenten anstelle von HMAC-SHA1 der HMAC-SHA256-Algorithmus verwendet. Mit XMLSpy 2018 generierte Signaturen, für die Passwörter verwendet werden, können mit früheren Versionen von XMLSpy nicht überprüft werden.

Transformationen

Die XML-Daten werden transformiert und das Ergebnis der Transformation wird zur Erstellung der Signatur verwendet. Sie können den Kanonisierungsalgorithmus, der auf die XML-Daten der Datei (den `SignedInfo`-Inhalt) anzuwenden ist, definieren, bevor die Signatur berechnet wird. Im Folgenden einige wichtige Unterschiede zwischen den Algorithmen:

- *Canonical XML mit oder ohne Kommentare:* Wenn Kommentare bei der Berechnung der Signatur einbezogen werden, so führt jede Änderung an den Kommentaren in den XML-Daten zu einem Fehler bei der Überprüfung der Signatur. Im anderen Fall dürfen Kommentare nach der Signierung des Dokuments geändert oder hinzugefügt werden, ohne dass die Signatur dadurch ungültig wird.
- *Base64:* Das Root- (oder Dokument-) Element des XML-Dokuments wird als Base64-kodiert betrachtet und in seiner Binärform gelesen. Wenn das Root-Element nicht in Base64 kodiert ist, wird ein Fehler zurückgegeben oder das Element als leer gelesen.
- *Keine:* Es wird keine Transformation durchgeführt und die XML-Daten aus der auf der Festplatte gespeicherten Binärdatei werden direkt an die Signaturüberprüfung übergeben. Alle späteren Änderungen an den Daten führen dazu, dass die Signatur nicht verifiziert werden kann. Wenn allerdings das Kontrollkästchen *Whitespaces entfernen* aktiviert ist, so werden alle Whitespace-Zeichen entfernt und Änderungen an Whitespace-Zeichen werden bei der Signaturüberprüfung ignoriert. Ein grundlegender Unterschied zwischen der Option *Keine* und einer der *Canonical*-Optionen ist, dass bei der Kanonisierung ein XML-Datenstrom erzeugt wird, in dem einige Unterschiede wie z.B. die Reihenfolge der Attribute normalisiert wird. Als Folge davon werden Änderungen wie eine andere Attributreihenfolge bei einer Kanonisierungstransformation normalisiert (sodass die Signatur verifiziert werden kann), während eine solche Änderung berücksichtigt wird, wenn keine Transformation durchgeführt wird (d.h. die Signatur kann nicht verifiziert werden). Beachten Sie allerdings, dass standardmäßig eine Kanonisierung erfolgt, wenn die Signatur (mittels enveloped oder enveloping) eingebettet wird. Die XML-Daten werden also in ihrem aktuellen Zustand (d.h. ohne Transformation) verwendet, wenn die Signatur "detached" ist, wenn *Keine* ausgewählt ist und wenn die Option *Whitespaces entfernen* deaktiviert ist.

Platzierung der Signatur

Die Signatur kann in die XML-Datei platziert oder als separate Datei erstellt werden. Es stehen die folgenden Optionen zur Verfügung:

- *Enveloped:* Das `Signature` Element wird als das letzte Child-Element des Root (Dokument)-Elements erstellt.
- *Enveloping:* Das `Signature` Element wird als das Root- (Dokument)-Element erstellt und das XML-Dokument wird als Child-Element in dieses Element eingefügt.
- *Detached:* Die XML-Signatur wird in Form einer separaten Datei erstellt. In diesem Fall können Sie die Dateierweiterung der Signatur definieren und festlegen ob: (i) die Erweiterung an den Namen der XML-Datei angehängt werden soll (z.B. `test.xml.xsig`) oder (ii) ob die Erweiterung die XML-Erweiterung der Datei ersetzen soll (z.B. `test.xsig`). Außerdem können Sie festlegen, ob die Referenz auf die XML-Datei in der Signaturdatei ein relativer oder absoluter Pfad sein soll.

Anmerkung: XML-Signaturen für XML-Schema- (`.xsd`)-Dateien können in der Schema-Ansicht als separate (detached) (nicht eingebettete) Signaturdateien erstellt werden. XML-Signaturen für XBRL-Dateien können in der XBRL-Ansicht als separate (detached) Signaturdateien erstellt werden. XML-Signaturen für WSDL-Dateien können in der WSDL-Ansicht als separate Signaturdateien erstellt werden oder über die Option "Enveloped" in die WSDL-Datei inkludiert werden.

Anmerkung: Wenn die XML-Signatur als separate (detached) Datei erstellt wird, werden die XML-Datei und die Signaturdatei durch eine Referenz in der Signaturdatei miteinander verknüpft. Folglich muss, wenn sich die Signatur in einer externen Datei befindet, bei der Überprüfung der Signatur die Signaturdatei aktiv sein und nicht die XML-Datei.

Schlüsselinformationen anhängen

Die Option *KeyInfo anhängen* steht zur Verfügung, wenn die Signatur auf einem Zertifikat basiert. Wenn die Signatur auf einem Passwort basiert, steht sie nicht zur Verfügung.

Wenn die Option aktiv ist, werden die Daten des öffentlichen Schlüssels in die Signatur platziert. Im anderen Fall werden die Schlüsselinformationen nicht in die Signatur inkludiert. Der Vorteil bei der Inkludierung der Schlüsselinformationen ist, dass das Zertifikat selbst (insbesondere die Daten des öffentlichen Schlüssels) für die Überprüfung nicht benötigt werden (da die Schlüsselinformationen ja in der Signatur vorhanden sind).

5.13.2 Überprüfen von XML-Signaturen

Eine XML-Signatur wird als korrekt bestätigt, wenn die XML-Datei seit der Signierung nicht mehr geändert wurde. Andernfalls schlägt die Überprüfung fehl. XML-Signaturen können in XMLSpy in den folgenden unten beschriebenen Fällen überprüft werden:

- Die XML-Datei enthält eine zertifikatbasierte Signatur, wobei die Schlüsselinformationen in der Signatur inkludiert sind 422
- Die XML-Datei enthält eine zertifikatbasierte Signatur, wobei die Schlüsselinformationen nicht in der Signatur inkludiert sind 423
- Die zertifikatbasierte Signatur befindet sich in einer externen Datei, die Schlüsselinformationen sind in der Signatur enthalten 423
- Die zertifikatbasierte Signatur befindet sich in einer externen Datei, wobei die Schlüsselinformationen nicht in der Signatur enthalten sind 424
- Die XML-Datei enhält eine passwortbasierte Signatur 424
- Die passwortbasierte Signatur befindet sich in einer externen Datei 424

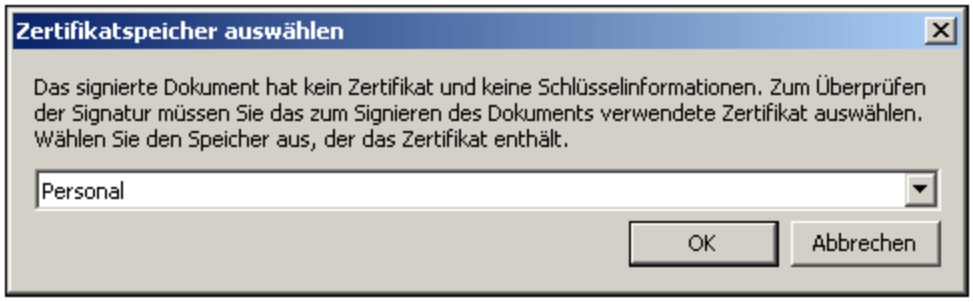
Die XML-Datei enthält eine zertifikatbasierte Signatur, wobei die Schlüsselinformationen in der Signatur inkludiert sind

Um die XML-Signatur in diesem Szenario zu überprüfen, machen Sie die XML-Datei in XMLSpy zu aktiven Datei. Wenn Sie den Befehl **XML | XML-Signatur überprüfen** auswählen, wird die Überprüfung durchgeführt und das Ergebnis wird im Fenster "Meldungen" angezeigt (Überprüfung erfolgreich oder fehlgeschlagen).

© 2016-2022 Altova GmbH

Die XML-Datei enthält eine zertifikatbasierte Signatur, wobei die Schlüsselinformationen nicht in der Signatur inkludiert sind

Falls die Schlüsselinformationen nicht in der zertifikatbasierten Signatur enthalten sind, werden Sie von XMLSpy aufgefordert, das Zertifikat anzugeben, aus dem die Daten des öffentlichen Schlüssels für die Überprüfung ausgelesen werden können. Bei der Überprüfung muss die XML-Datei in XMLSpy aktiv sein. Wenn Sie den Befehl **XML | XML-Signatur überprüfen** auswählen, werden Sie aufgefordert, den Zertifikatspeicher[425], dem das Zertifikat sich befindet, auszuwählen (*Abbildung unten*).



Nach Auswahl eines Zertifikatspeichers[425] und Klicken auf **OK** wird ein Dialogfeld angezeigt, in dem die Zertifikate in diesem Speicher angezeigt werden (*Abbildung unten*). Wählen Sie das gewünschte Zertifikat aus und klicken Sie auf **OK**.



Die Signatur wird überprüft und das Ergebnis wird im Fenster "Meldungen" angezeigt.

Die zertifikatbasierte Signatur befindet sich in einer externen Datei, die Schlüsselinformationen sind in der Signatur enthalten

Wenn sich eine zertifikatbasierte XML-Signatur in einer externen Datei befindet, muss bei der Überprüfung der Signatur die Signaturdatei in XMLSpy aktiv sein. Wenn Sie den Befehl **XML | XML-Signatur überprüfen**

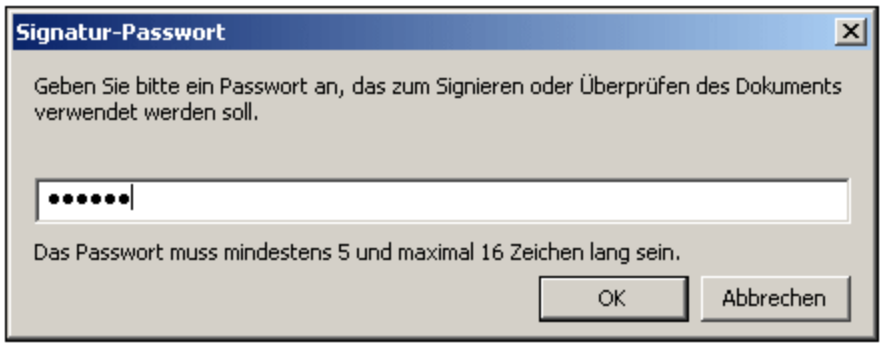
auswählen, wird die Überprüfung durchgeführt und das Ergebnis im Fenster "Meldungen" angezeigt (Überprüfung erfolgreich oder fehlgeschlagen).

Die zertifikatbasierte Signatur befindet sich in einer externen Datei, wobei die Schlüsselinformationen nicht in der Signatur enthalten sind

Wenn sich eine zertifikatbasierte XML-Signatur in einer externen Datei befindet, muss bei der Überprüfung der Signatur die Signaturdatei in XMLSpy aktiv sein. Wenn Sie den Befehl **XML | XML-Signatur überprüfen** auswählen, werden Sie aufgefordert das Zertifikat anzugeben, aus dem die Daten für den öffentlichen Schlüssel für die Überprüfung ausgelesen werden können. Wählen Sie das Zertifikat wie im Abschnitt: *Die XML-Datei enthält eine zertifikatbasierte Signatur, wobei die Schlüsselinformationen nicht in der Signatur inkludiert sind*[423] beschrieben ist, aus. Die Überprüfung wird durchgeführt und das Ergebnis wird im Fenster "Meldungen" angezeigt (Überprüfung erfolgreich oder fehlgeschlagen).

Die XML-Datei enhält eine passwortbasierte Signatur

Wenn die XML-Datei eine passwortbasierte XML-Signatur enthält, muss bei der Überprüfung der Signatur die XML-Datei in XMLSpy aktiv sein. Wenn Sie den Befehl **XML | XML-Signatur überprüfen** auswählen, erscheint ein Dialogfeld, in dem Sie aufgefordert werden, das Passwort einzugeben (*Abbildung unten*).

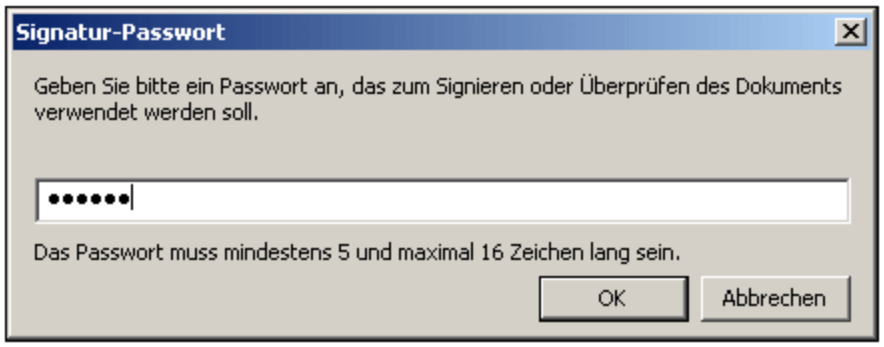


Geben Sie das Passwort ein, das fünf bis sechzehn Zeichen lang sein muss, und klicken Sie auf **OK**. Die Überprüfung wird durchgeführt und das Ergebnis wird im Fenster "Meldungen" angezeigt (Überprüfung erfolgreich oder fehlgeschlagen).

Die passwortbasierte Signatur befindet sich in einer externen Datei

Wenn sich eine passwortbasierte XML-Signatur in einer externen Datei befindet, muss bei der Überprüfung der Signatur die Signaturdatei in XMLSpy aktiv sein. Wenn Sie den Befehl **XML | XML-Signatur überprüfen** auswählen, erscheint ein Dialogfeld, in dem Sie aufgefordert werden, das Passwort einzugeben (*Abbildung unten*).

© 2016-2022 Altova GmbH



Geben Sie das Passwort ein, das fünf bis sechzehn Zeichen lang sein muss und klicken Sie auf **OK**. Die Überprüfung wird durchgeführt und das Ergebnis wird im Fenster "Meldungen" angezeigt (Überprüfung erfolgreich oder fehlgeschlagen).

5.13.3 Arbeiten mit Zertifikaten

Zur Erstellung und Überprüfung von XML-Signaturen werden oft Autorisierungszertifikate verwendet. Dieser Abschnitt enthält Informationen zum Beschaffen, Importieren und Exportieren von Zertifikaten. Er ist in die folgenden Unterabschnitte gegliedert:

- Beschaffung eines Zertifikats mit einem Private-Public-Key-Paar 425
- Importieren eines Private-Public-Key-Zertifikats 425
- Die Zertifikatspeicher auf einem Windows-Rechner 426
- Exportieren eines Public-Key-Zertifikats 427

Beschaffung eines Zertifikats mit einem Private-Public-Key-Paar

Ein Zertifikat kann auf die folgenden Arten eingeholt werden:

- *Von einer Zertifizierungsstelle.* Die Zertifizierungsstelle überprüft die Identität des Zertifikatinhabers. Dies steht im Gegensatz zu selbstsignierten Zertifikaten, die von jeder Person im Besitz eines Zertifkaterstellungstools erzeugt werden können.
- *Durch Erstellung eines selbstsignierten Zertifikats.* Derartige Zertifikate werden nicht von einer Zertifizierungsstelle überprüft und bieten dennoch meist genug Schutz. Eine Reihe von Tools, wie z.B. Microsoft Visual Studio unterstützt die Erstellung von Zertifikaten.

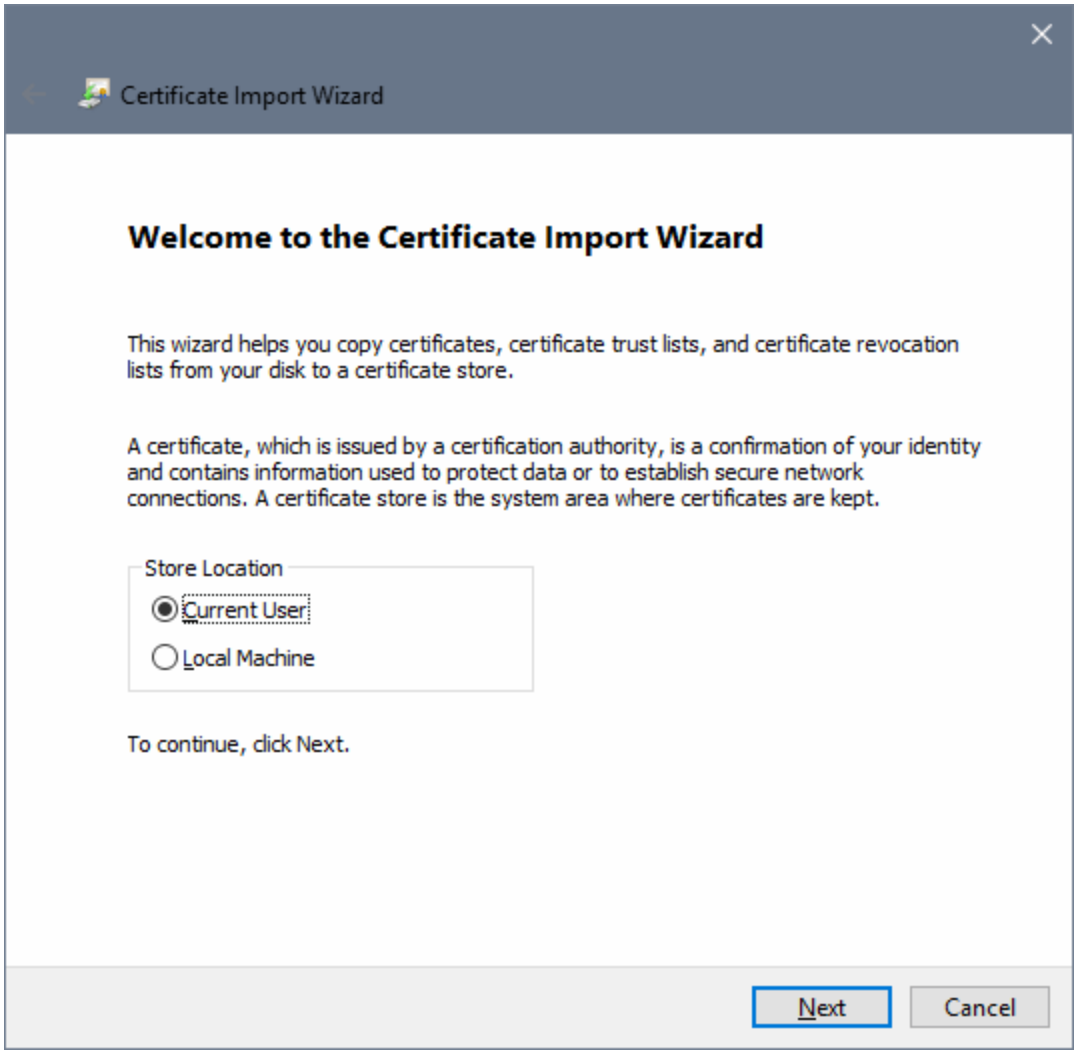
Für die Verwendung mit XML-Signaturen benötigen Sie ein Zertifikat mit einem Private-Public-Key-Paar.

Anmerkung: Die XML-Signaturfunktion von XMLSpy unterstützt Zertifikate vom Typ RSA-SHA1, DSA-SHA1 und SHA-256

Importieren eines Private-Public-Key-Zertifikats

Nach Erhalt eines Private-Public-Key-Zertifikats müssen Sie es in Ihren Windows-Zertifikatspeicher importieren. Gehen Sie dabei folgendermaßen vor:

1. Doppelklicken Sie auf die Zertifikatdatei, um sie im Zertifikatimport-Assistenten zu öffnen (*Abbildung unten*) und klicken Sie auf **Weiter.**



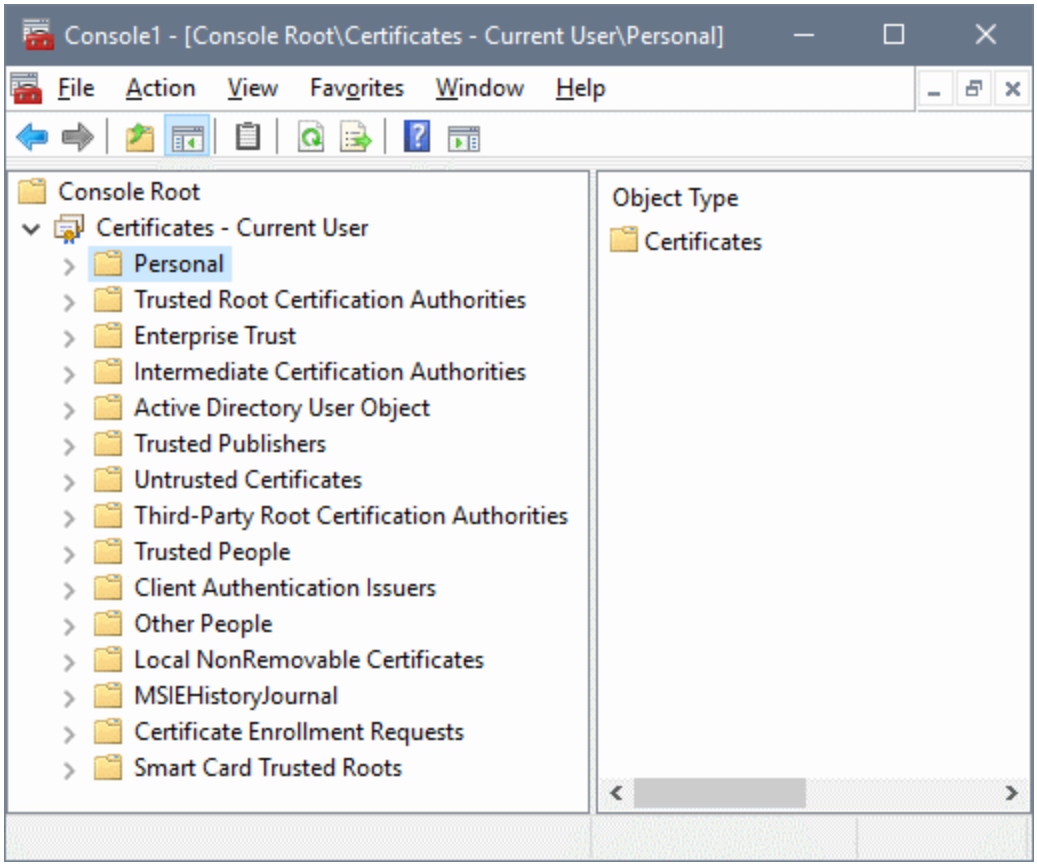
2. Stellen Sie im Fenster "Importdatei" sicher, dass die Zertifikatdatei ausgewählt ist und klicken Sie auf **Weiter.**
3. Geben Sie das Passwort für den privaten Schlüssel ein. Sie müssen das Passwort kennen, wenn Sie vorhaben, den privaten Schlüssel zum Erstellen einer XML-Signatur zu verwenden. Sie erhalten das Passwort für den privaten Schlüssel bei Erhalt des Zertifikats. Klicken Sie nach Eingabe des Passworts auf **Weiter.**
4. Sie können den Assistenten den Speicher, in den das Zertifikat - nach Zertifikatstyp - platziert werden soll, automatisch auswählen lassen oder Sie können den Speicher selbst auswählen. (Es empfiehlt sich, den Speicher selbst auszuwählen, damit Sie wissen, wo das Zertifikat gespeichert ist.) Klicken Sie anschließend auf **Weiter.**
5. Klicken Sie auf **Fertig stellen** um den Vorgang abzuschließen.

Die Zertifikatspeicher auf einem Windows-Rechner

Der Zertifikatspeicher auf einem Windows XP-Betriebssystem kann folgendermaßen aufgerufen werden:

© 2016-2022 Altova GmbH

1. Klicken Sie im im Startmenü auf **Ausführen**.
2. Geben Sie `mmc` ein und klicken Sie auf **OK**. Daraufhin wird ein Konsolenfenster angezeigt (*Abbildung unten*).



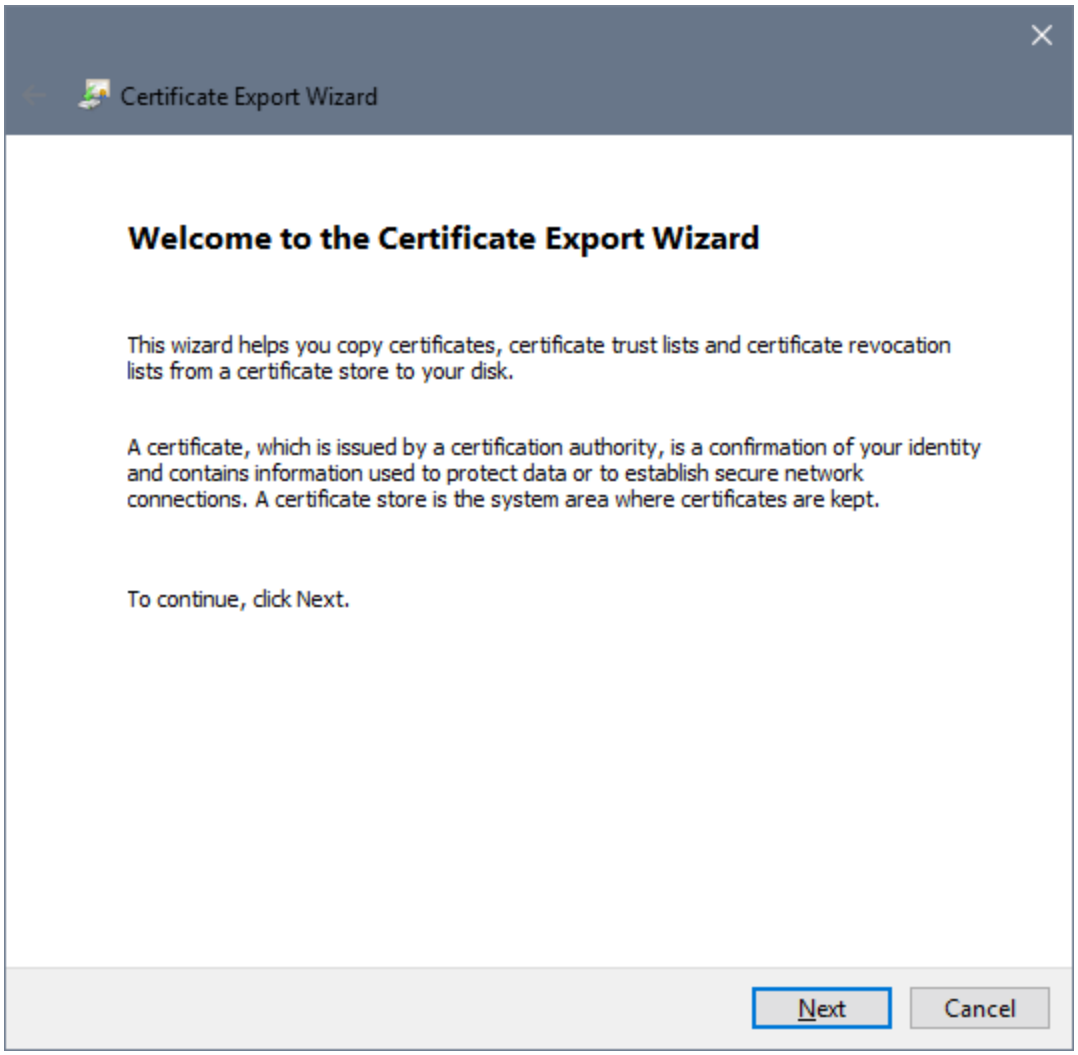
3. Wählen Sie im Konsolenfenster den Befehl **Datei | Snap-In hinzufügen/entfernen**.
4. Klicken Sie im Abschnitt *Eigenständig* des Dialogfelds "Snap-In hinzufügen/entfernen" auf **Hinzufügen.**
5. Wählen Sie im Dialogfeld "Eigenständiges Snap-In hinzufügen" den Eintrag "Zertifikate" und klicken Sie auf **Hinzufügen.**
6. Schließen Sie das Dialogfeld "Eigenständiges Snap-In hinzufügen".
7. Klicken Sie im Dialogfeld "Snap-In hinzufügen/entfernen" auf **OK**.
8. Der Konstolenstamm im Fenster "Konsole" enthält nun einen Zertifikateintrag (*siehe Abbildung oben*). Dieser Zertifikateintrag enthält die Zertifikatspeicher auf Ihrem Rechner.
9. Speichern Sie die Konsole mit dem Befehl **Datei | Speichern** des Konsolenfensters als Microsoft Management Console-Datei (`.msc`-Datei). Später können Sie mit Hilfe dieser MSC-Datei (über den Befehl **Datei | Öffnen** eines Konsolenfensters) auf den Zertifikatspeicher auf Ihrem Rechner zugreifen.

Exportieren eines Public-Key-Zertifikats

Wenn Sie ein Zertifikat mit einem Private-Public-Key haben, sollten Sie dieses eventuell nur mit einem öffentlichen Schlüssel exportieren. Dieses Public-Key-Zertifikat kann anschließend an Empfänger gesendet werden, die damit Signaturen überprüfen können, die mit dem privaten Schlüssel des Zertifikats erstellt wurden.

Ein Public-Key-Zertifikat kann folgendermaßen aus einem vorhandenen Private-Public-Key-Zertifikat exportiert werden:

1. Öffnen Sie die Zertifikatspeicher in einem Konsolenfenster. Gehen Sie dazu folgendermaßen vor: (i) Geben Sie im Startmenü in die Befehlszeile von **Ausführen** `mmc` ein; (ii) wählen Sie im Fenster "Konsole", das daraufhin angezeigt wird, den Befehl **Datei | Öffnen** und wählen Sie die MSC-Datei aus, in der die Zertifikatspeicher gespeichert wurden (*siehe Abschnitt unmittelbar oberhalb dieses Abschnitts*).
2. Navigieren Sie zum Zertifikat, das Sie als Public-Key-Zertifikat exportieren möchten, und klicken Sie mit der rechten Maustaste darauf.
3. Wählen Sie im Kontextmenü, das daraufhin angezeigt wird, den Befehl **Alle Aufgaben | Exportieren**. Daraufhin wird der Zertifikatexport-Assistent angezeigt (*Abbildung unten*).



4. Klicken Sie auf **Weiter**.
5. Wählen Sie im Fenster "Privaten Schlüssel exportieren" die Option *Nein, privaten Schlüssel nicht exportieren* und klicken Sie auf **Weiter.**
6. Wählen Sie im Fenster "Format der zu exportierenden Datei" das gewünschte Format aus (belassen Sie das Standardformat DER, wenn Sie sich nicht sicher sind) und klicken Sie auf **Weiter**.

© 2016-2022 Altova GmbH

7. Navigieren Sie im Fenster "Zu exportierende Datei" zu dem Ordner, in dem die Datei gespeichert werden soll, und geben Sie einen Namen für die Datei an (ohne Angabe einer Dateierweiterung, da diese automatisch angehängt wird). Klicken Sie anschließend auf **Weiter.**
8. Klicken Sie auf **Fertig stellen** um den Export abzuschließen.

Unter dem angegebenen Pfad wird ein Public-Key-Zertifikat erstellt. Dieses Public-Key-Zertifikat kann nun an Empfänger der XML-Datei gesendet werden, die mit dem entsprechenden privaten Schlüssel signiert wurde. Der Empfänger kann diesen öffentlichen Schlüssel dann in einen Zertifikatspeicher auf seinem Rechner importieren und die XML-Datei anhand dieses öffentlichen Schlüssels überprüfen.

5.14 Zusätzliche Funktionen

Im Folgenden werden weitere Funktionen für die Arbeit mit XML-Dateien aufgelistet.

- Kodierung 430
- Generieren von DTDs und XML-Schemas 430
- Suchen und Ersetzen 430
- XPath auswerten 431
- Importieren und Exportieren von Text 431

Kodierung

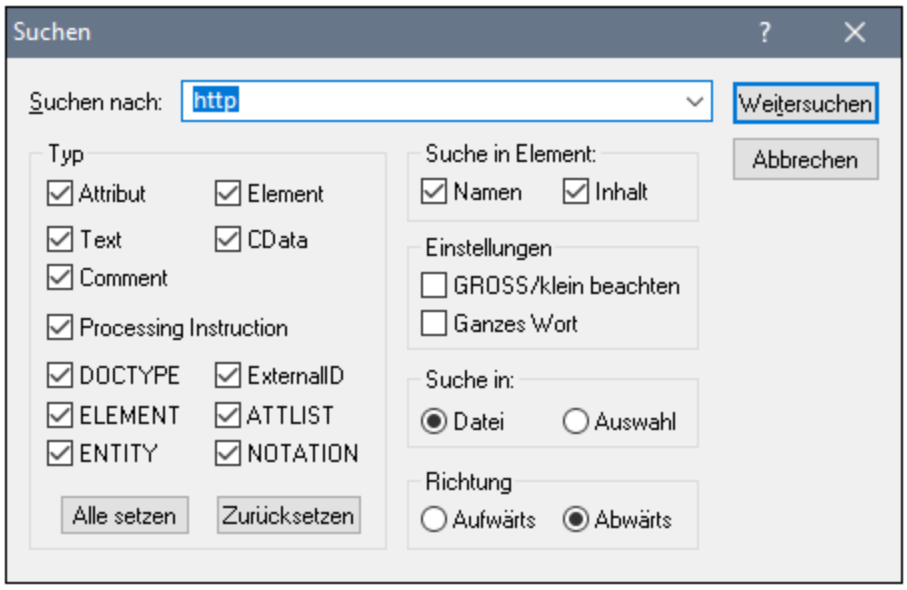
Die Kodierung von XML-Dateien (und anderen Dokumentarten) kann mit Hilfe des Menübefehls **Datei | Kodierung** 1285 gespeichert werden. Die Standardkodierung von XML- und Nicht-XML.-Dateien kann im Abschnitt Optionen | Kodierung 1608 gespeichert werden.

Generieren von DTDs und XML-Schemas

Wenn Sie ein Schema erstellen möchten, in dem die Struktur des XML-Dokuments beschrieben wird, verwenden Sie den Befehl **DTD/Schema | DTD/Schema generieren** 1374. Im Dialogfeld "DTD/Schema generieren", das daraufhin angezeigt wird, können Sie auswählen, ob eine DTD oder ein XML-Schema generiert werden soll. Außerdem stehen bestimmte Schema-Optionen zur Verfügung wie z.B., ob anhand der im XML-Dokument enthaltenen Werte Enumerationen generiert werden sollen.

Suchen und Ersetzen

Die Funktionen Suchen 1307 und Ersetzen 1310 (die über das Menü **Bearbeiten** aufgerufen werden) sind leistungsstarke Suchfunktionen. Zusätzlich können Sie die Suche einschränken, sodass Groß- und Kleinschreibung beachtet werden oder nur ganze Wörter gesucht werden sollen. Zur Suche kann auch eine Regular Expression verwendet werden. Der Suchbereich kann auf eine Auswahl im Dokument und auf bestimmte Node-Typen eingeschränkt werden (*siehe Abbildung unten*).



© 2016-2022 Altova GmbH

In der Abbildung oben sehen Sie das Dialogfeld "Suchen" der Grid-Ansicht. Das Dialogfeld und seine Funktionalitäten sehen je nachdem, welche Ansicht aktiv ist, unterschiedlich aus. Informationen zu den Funktionen "Suchen" und "Ersetzen" finden Sie hier[157].

XPath auswerten

Ein XPath-Ausdruck, den Sie in das Fenster "XPath/XQuery" eingeben, kann gegen das aktive XML-Dokument ausgewertet werden. Die Ergebnisse der Auswertung werden im Fenster "XPath/XQuery" angezeigt. Wenn Sie in den Ergebnissen auf eine Node klicken, wird dieser Node in der Dokumentansicht im Hauptfenster markiert. Beachten Sie, dass Sie das Fenster "XPath/XQuery" durch Klicken auf den Befehl **XML | XPath auswerten** aktivieren können.

Importieren und Exportieren von Text

Textdaten können aus anderen Applikationsformaten importiert und dorthin exportiert werden. Befehle zu diesen Funktionen finden Sie im Menü **Konvertieren**[1470].

6 DTDs und XML Schemas

Altova Website: XML-Schema Editor

Dieser Abschnitt enthält eine Übersicht über das Arbeiten mit DTDs[448] und XML-Schemas[451]. Außerdem wird darin beschrieben, wie man mit SchemaAgent arbeitet[470] und wie man die leistungsstarken In Schemas suchen[482]-Funktionen nutzen kann. Zusätzlich zu den Bearbeitungsfunktionen bietet XMLSpy einige äußerst nützliche DTD/Schema-Funktionen:

Katalog-Mechanismus

Dank der Unterstützung für den OASIS Katalog-Mechanismus[463] können URIs auf lokale Adressen umgeleitet werden, was die Verwendung auf mehreren Arbeitsplätzen erleichtert.

Schemaregeln

Einem XML-Schema kann eine Gruppe zusätzlicher benutzerdefinierter Einschränkungen zugewiesen werden. XMLSpy enthält einen Schemaregel-Editor, in dem Sie eine Schemaregeldatei für ein XML-Schema erstellen und bearbeiten können.

Schemamodule

In der Schema-Ansicht können die Komponenten eines großen Schemas als separate Dateien angelegt werden. Diese kleineren Schemamodule können anschließend in das größere Schema inkludiert werden. Beim umgekehrten Vorgang, der als "Schemanivellierung" bezeichnet wird, werden die Komponenten inkludierter Dateien direkt in das größere Schema platziert. Eine Anleitung zum Generieren von Schemamodulen und zur Nivellierung von Schemas finden Sie im Abschnitt Schemamodule[452].

Konvertieren von DTDs in XML-Schemas und umgekehrt

Mit den Befehlen im Menü **DTD/Schema**[1371] kann eine DTD in ein XML-Schema konvertiert werden und umgekehrt. Außerdem können damit beide Arten von Dokumenten nivelliert werden. Wenn eine DTD nivelliert wird, werden Komponenten in inkludierten/importierten Modulen direkt in der übergeordneten Datei gespeichert und nicht verwendete Komponenten werden gelöscht.

Erzeugen einer XML-Beispieldatei

Mit dem Menübefehl **DTD/Schema | XML-Beispieldatei erzeugen**[1381] wird anhand der aktiven DTD oder XML-Schema-Datei ein XML-Skelettdokument erzeugt. Dieser Befehl eignet sich gut, um anhand eines Schemas schnell eine XML-Datei zu generieren.

Gehe zu Definition

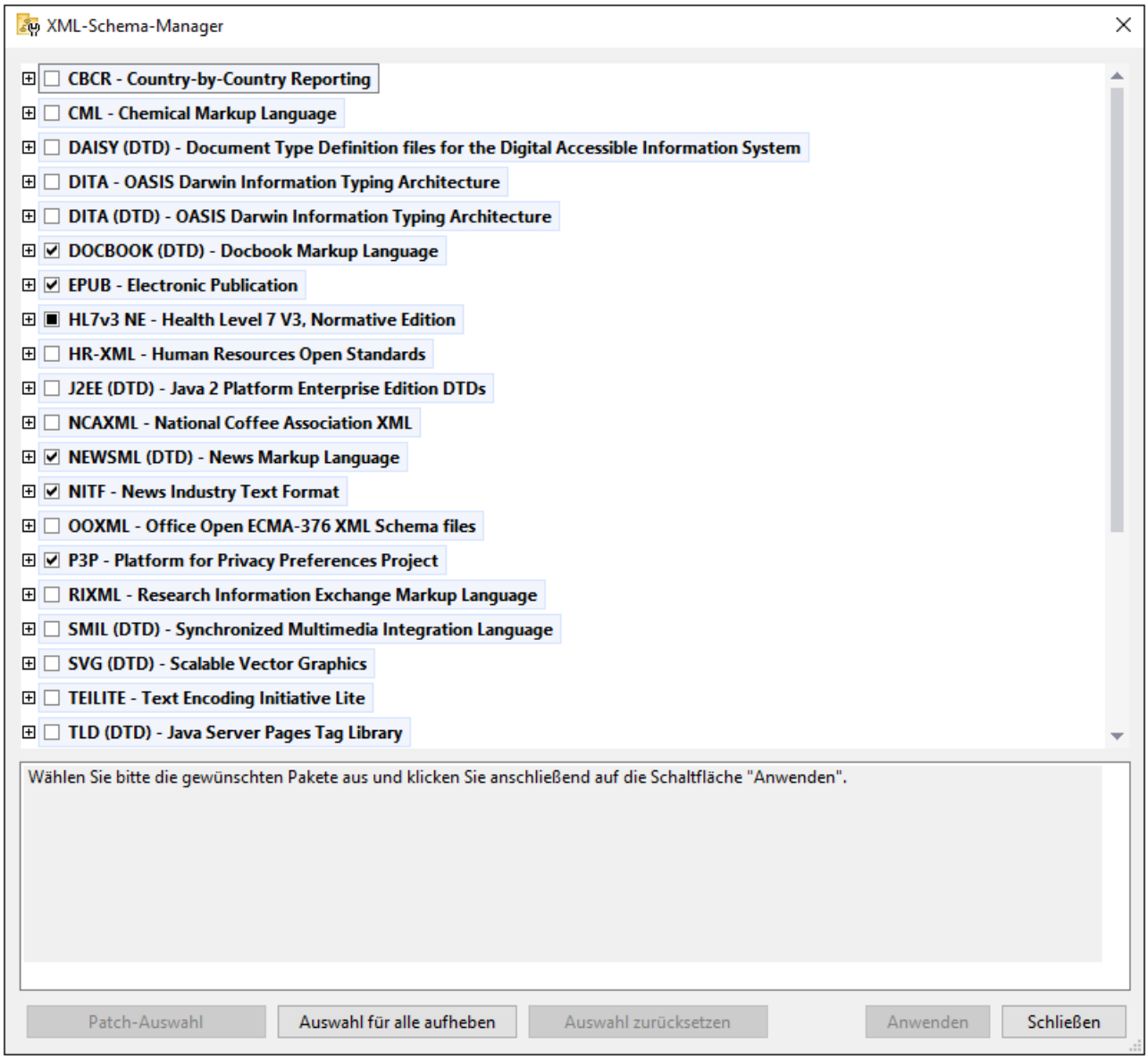
Wenn sich der Cursor in einem XML-Dokument innerhalb eines Node befindet, wird bei Auswahl des Menübefehls **DTD/Schema | Gehe zu Definition**[1373] die Schema-Datei geöffnet und die Definition des ausgewählten XML-Node wird markiert.

© 2016-2022 Altova GmbH

6.1 Schema-Manager

Der XML-Schema-Manager ist ein Altova-Tool, mit dem Sie XML-Schemas (DTDs für XML-Dateien und XML-Schemas) zentral installieren und verwalten können, um diese in allen XML-fähigen Applikationen von Altova einschließlich XMLSpy verwenden zu können.

- Unter Windows hat der Schema-Manager eine grafische Benutzeroberfläche (*siehe Abbildung unten*) und steht auch über die Befehlszeile zur Verfügung. (Die Desktop-Applikationen von Altova stehen nur unter Windows zur Verfügung; *siehe Liste unten*).
- Unter Linux und MacOS steht der Schema-Manager nur über die Befehlszeile zur Verfügung. (Die Server-Applikationen von Altova stehen unter Windows, Linux und macOS zur Verfügung; *siehe Liste unten*).



Altova-Applikationen, die mit Schema-Manager arbeiten:

Desktop-Applikationen (nur Windows)	Server-Applikationen (Windows, Linux, macOS)
XMLSpy (alle Editionen)	RaptorXML Server, RaptorXML+XBRL Server
MapForce (alle Editionen)	StyleVision Server
StyleVision (alle Editionen)	
Authentic Desktop Enterprise Edition	

Installation und Deinstallation des Schema-Managers

Der Schema-Manager wird bei der ersten Installation einer neuen Version der Altova Mission Kit Enterprise Edition oder einer der XML-Schema-fähigen Applikationen von Altova (*siehe Tabelle oben*) automatisch installiert.

Ebenso wird er auch automatisch entfernt, wenn Sie die letzte XML-Schema-fähige Applikation von Altova auf Ihrem Rechner deinstallieren.

Schema-Manager-Funktionalitäten

Im Schema-Manager stehen die folgenden Funktionalitäten zur Verfügung:

- Anzeigen der auf Ihrem Rechner installierten XML-Schemas und Überprüfung, ob neue Versionen zum Download zur Verfügung stehen.
- Download neuer Versionen von XML-Schemas unabhängig vom Altova Produkt-Release-Zyklus. (Die Schemas werden von Altova online bereitgestellt und können über den Schema-Manager heruntergeladen werden).
- Installation oder Deinstallation jeder beliebigen (oder ggf. aller) der zahlreichen Versionen eines bestimmten Schemas.
- Ein XML-Schema kann Abhängigkeiten von anderen Schemas aufweisen. Bei der Installation oder Deinstallation eines bestimmten Schemas informiert Sie der Schema-Manager über davon abhängige Schemas und installiert bzw. entfernt diese ebenfalls automatisch.
- Der Schema-Manager ordnet Schema-Referenzen mit Hilfe des XML-Katalogs lokalen Dateien zu. Dadurch lassen sich große XML-Schemas schneller verarbeiten, als wenn sie sich unter einem entfernten Pfad befinden.
- Alle wichtigen Schemas werden über den Schema-Manager bereitgestellt und regelmäßig auf die jeweils neuesten Version aktualisiert. Dadurch können alle Ihre Schemas zentral verwaltet werden und stehen allen XML-Schema-fähigen Applikationen von Altova jederzeit zur Verfügung.
- Im Schema-Manager vorgenommene Änderungen werden für alle auf dem Rechner installierten Altova-Produkte wirksam.

Funktionsweise

Alle in Altova-Produkten verwendeten XML-Schemas werden von Altova online bereitgestellt. Dieser Speicher wird bei Veröffentlichung neuer Versionen der Schemas aktualisiert. Im Schema-Manager werden sowohl bei Aufruf über die Benutzeroberfläche als auch über das CLI Informationen über die neuesten verfügbaren Schemas angezeigt. Sie können die gewünschten Schemas dann über den Schema-Manager installieren, aktualisieren oder deinstallieren.

© 2016-2022 Altova GmbH

Schemas können vom Schema-Manager auch auf eine weitere Art installiert werden. Sie können ein Schemas und die davon abhängigen Schemas auf der Altova Website (https://www.altova.com/de/schema-manager) auswählen. Daraufhin wird auf der Website eine Datei des Typs **.altova_xmlschemas** mit Informationen über Ihre ausgewählten Schemas zum Download vorbereitet. Bei Doppelklick auf diese Datei oder bei Übergabe an den Schema-Manager über das CLI als Argument des Befehls `install`[443] installiert der Schema-Manager die ausgewählten Schemas.

Lokaler Cache: Überprüfung Ihrer Schemas

Alle Informationen über installierte Schemas werden in einem zentralen Cache-Verzeichnis auf Ihrem Rechner aufgezeichnet. Das Verzeichnis befindet sich hier:

Windows	`C:\ProgramData\Altova\pkgs\.cache`
Linux	`/var/opt/Altova/pkgs\.cache`
macOS	`/var/Altova/pkgs`

Dieses Cache-Verzeichnis wird regelmäßig mit dem neuesten Status der Schemas aus dem Online-Speicher von Altova aktualisiert. Diese Aktualisierungen finden unter den folgenden Bedingungen statt:

- bei jedem Start von Schema-Manager.
- Wenn Sie XMLSpy zum ersten Mal an einem bestimmten Kalendertag starten.
- Wenn XMLSpy länger als 24 Stunden geöffnet ist, findet alle 24 Stunden eine Aktualisierung des Cache statt.
- Sie können den Cache auch durch Ausführung des `update`[446]-Befehls über die Befehlszeilenschnittstelle aktualisieren.

Der Schema-Manager kann somit Ihre installierten Schemas über den Cache ständig anhand der online verfügbaren Schemas auf der Altova Website überprüfen.

Nehmen Sie keine manuellen Änderungen am Cache vor!

Das lokale Cache-Verzeichnis wird automatisch auf Basis der installierten oder deinstallierten Schemas verwaltet; es sollte nicht manuell geändert oder gelöscht werden. Falls Sie den Schema-Manager je in seinen Originalzustand zurücksetzen möchten, (i) führen Sie den CLI-Befehl `reset`[445] der Befehlszeilenschnittstelle und (ii) anschließend den Befehl `initialize`[443] aus. (Führen Sie alternativ dazu den Befehl **reset** mit der Option **-i** aus).

6.1.1 Ausführen des Schema-Managers

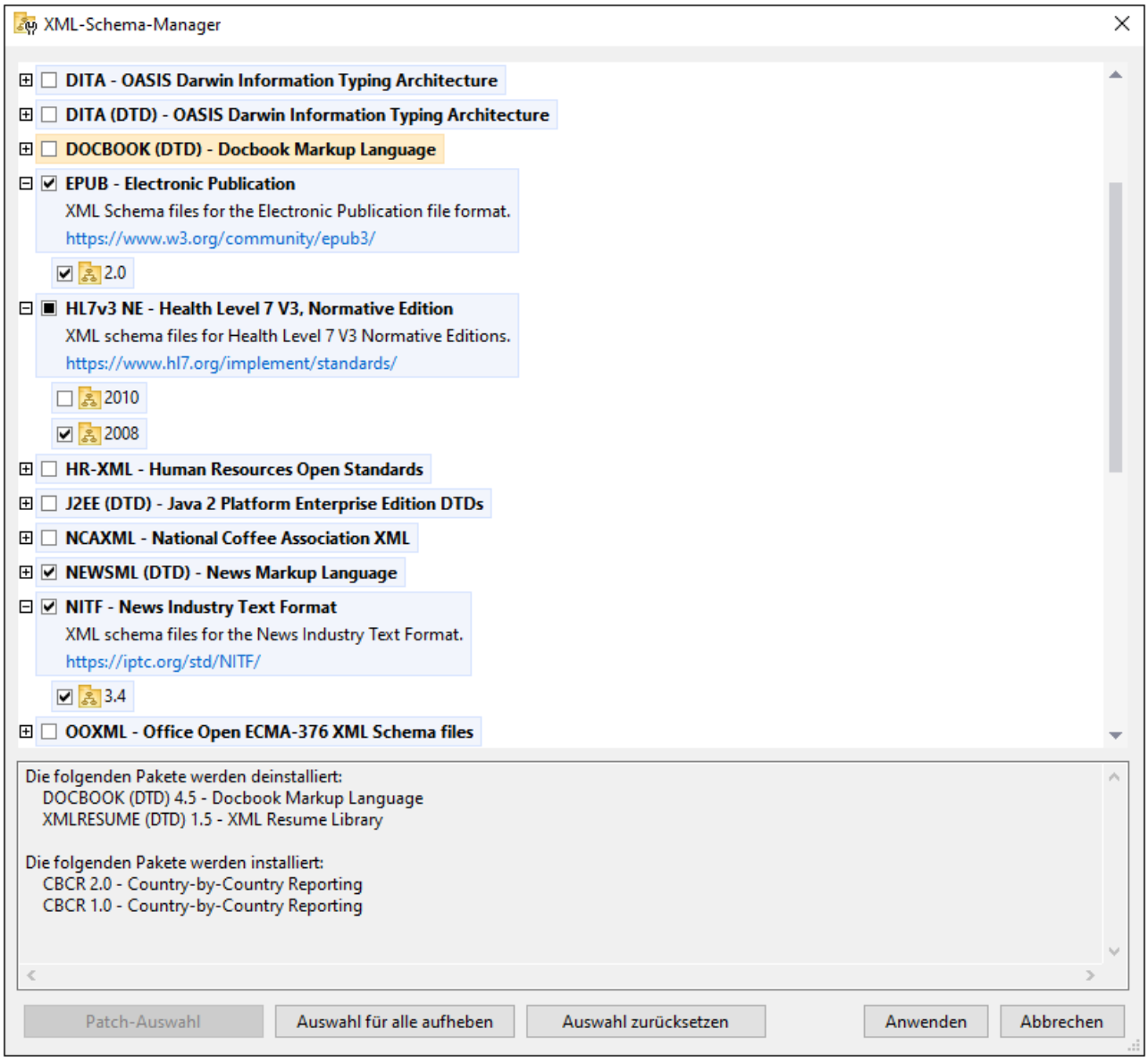
Grafische Benutzeroberfläche

Sie können die Benutzeroberfläche des Schema-Managers auf eine der folgenden Arten aufrufen:

- *Bei der Installationen von XMLSpy:* Aktivieren Sie gegen Ende der Installation das Kontrollkästchen *Altova-Schema-Manager aufrufen*, wodurch Sie die Benutzeroberfläche des Schema-Managers direkt aufrufen können. Auf diese Art können Sie Schemas während der Installation Ihrer Altova-Applikation installieren.

- *Nach der Installation von XMLSpy:* Nachdem die Applikation installiert wurde, können Sie die Benutzeroberfläche des Schema-Managers jederzeit über den Menübefehl **Tools | XML-Schemas-Manager** aufrufen.
- über die von der Altova Webseite heruntergeladene `.altova_xmlschemas`-Datei: Doppelklicken Sie auf die heruntergeladene Datei, um den Schema-Manager zu starten, der daraufhin die (auf der Website) ausgewählten Schemas installiert.

Nachdem die Benutzeroberfläche des Schema-Managers geöffnet wurde (*Abbildung unten*), werden bereits installierte Schemas markiert angezeigt. Wenn ein zusätzliches Schema installiert werden soll, aktivieren Sie dieses. Wenn ein bereits installiertes Schema deinstalliert werden soll, deaktivieren Sie dieses. Nachdem Sie Ihre Auswahl getroffen haben, können Ihre Änderungen angewendet werden. Die Schemas, die installiert bzw. deinstalliert werden, werden markiert und im Fenster "Meldungen" am unteren Rand des Schema-Manager-Fensters (*siehe Abbildung*) erscheint eine Meldung über die bevorstehenden Änderungen.



 © 2016-2022 Altova GmbH

Befehlszeilenschnittstelle

Sie können den Schema-Manager über eine Befehlszeilenschnittstelle starten, indem Sie Befehle an die ausführbare Datei **xmlschemamanager.exe** senden.

Die Datei **xmlschemamanager.exe** steht im folgenden Ordner zur Verfügung:

- *unter Windows:* `C:\ProgramData\Altova\SharedBetweenVersions`
- *Unter Linux oder macOS (nur Server-Applikationen):* `%INSTALLDIR%/bin`, wobei `%INSTALLDIR%` das Installationsverzeichnis des Programms ist.

Anschließend können Sie jeden der im Abschnitt zur CLI-Befehlsreferenz 441 aufgelisteten Befehle verwenden.

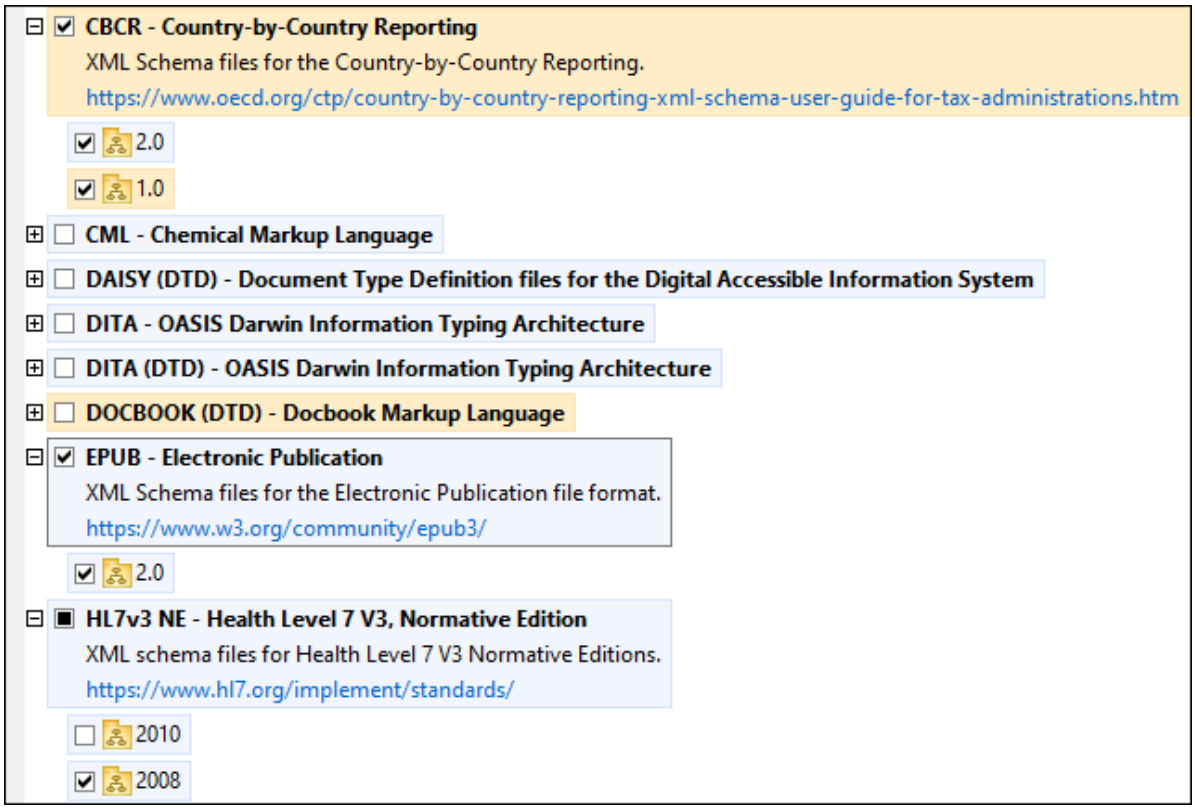
Um die Hilfe zu den Befehlen anzuzeigen, führen Sie den folgenden Befehl aus:

- *unter Windows:* `xmlschemamanager.exe --help`
- *Unter Linux oder macOS (nur Server-Applikationen):* `sudo ./xmlschemamanager --help`

6.1.2 Statuskategorien

Der Schema-Manager unterscheidet folgendermaßen zwischen den von ihm verwalteten Schemas:

- *installierte Schemas.* Diese werden auf der Benutzeroberfläche mit einem Häkchen angezeigt (*in der Abbildung unten sind die mit einem Häkchen versehenen und blau angezeigten Versionen der EPUB- und HL7v3 NE-Schemas installiert*). Wenn alle Versionen eines Schemas ausgewählt sind, wird ein Häkchen angezeigt. Wenn zumindest eine Version nicht ausgewählt ist, wird ein gefülltes Quadrat angezeigt. Sie können das Kontrollkästchen für ein installiertes Schema deaktivieren, um es zu **deinstallieren**; (*in der Abbildung unten ist die DocBook DTD installiert und wurde deaktiviert; sie wird daher für die Deinstallation vorbereitet*).
- *Nicht installierte verfügbare Schemas.* Diese werden auf der Benutzeroberfläche mit einem deaktivierten Kontrollkästchen angezeigt. Sie können die Schemas, die **installiert** werden sollen, auswählen.



- *Schemas, für die ein Upgrade zur Verfügung steht* sind diejenigen, die seit ihrer Installation vom Herausgeber überarbeitet wurden. Sie werden auf der Benutzeroberfläche durch ein -Symbol gekennzeichnet. Sie können für ein installiertes Schema ein **Patch** der verfügbaren überarbeiteten Version installieren.

Wichtige Punkte

- In der Abbildung oben sind beide CBCR-Schemas ausgewählt. Dasjenige mit einem blauen Hintergrund ist bereits installiert. Dasjenige mit dem gelben Hintergrund ist nicht installiert und wurde für die Installation ausgewählt. Beachten Sie, dass das Schema "HL7v3 NE 2010" nicht installiert ist und nicht für die Installation ausgewählt wurde.
- Ein gelber Hintergrund bedeutet, dass das Schema auf irgendeine Art geändert wird, wenn Sie auf die Schaltfläche **Anwenden** klicken. Wenn ein Schema deaktiviert ist und einen gelben Hintergrund hat, bedeutet dies, dass es bei Klick auf die Schaltfläche **Anwenden** deinstalliert wird. In der Abbildung oben ist dies bei der DocBook DTD der Fall.
- Bei Ausführung des Schema-Managers über die Befehlszeile wird der Befehl `list` 444 mit verschiedenen Optionen verwendet, um verschiedene Schemakategorien aufzulisten:

`xmlschemamanager.exe list`	Listet alle installierten und verfügbaren Schemas auf; auch verfügbare Upgrades werden angezeigt.
`xmlschemamanager.exe list -i`	Listet nur installierte Schemas auf; auch verfügbare Upgrades werden angezeigt.

© 2016-2022 Altova GmbH

`xmlschemamanager.exe list -u`	Listet Schemas auf, für die Upgrades zur Verfügung stehen

Anmerkung: Verwenden Sie unter Linux und macOS `sudo ./xmlschemamanager list`

6.1.3 Anwenden eines Patch oder Installation eines Schemas

Anwenden eines Patch auf ein installiertes Schema

Von Zeit zu Zeit werden von den Herausgebern der XML-Schemas Patches (Upgrades oder Überarbeitungen) veröffentlicht. Wenn der Schema-Manager erkennt, dass Patches zur Verfügung stehen, werden diese in der Schemaliste des Schema-Managers angezeigt und Sie können diese Patches schnell installieren.

Über die Benutzeroberfläche

Patches werden mit dem Symbol gekennzeichnet. (*Siehe auch vorhergehendes Kapitel über Statuskategorien*[437]*).* Falls Patches zur Verfügung stehen, ist die Schaltfläche **Patch-Auswahl** aktiv. Klicken Sie darauf, um alle Patches für die Installation auszuwählen und vorzubereiten. Auf der Benutzeroberfläche ändert sich das Symbol von Schemas, für die ein Patch installiert wird von in , und im Fenster "Meldungen" am unteren Rand des Dialogfelds werden die Patches, die angewendet werden, aufgelistet. Sobald Sie mit der Auswahl fertig sind, klicken Sie auf **Anwenden**. Alle Patches werden gemeinsam angewendet. Beachten Sie, dass ein für die Installation eines Patch markiertes Schema deinstalliert wird, wenn Sie die Auswahl aufheben.

Über das CLI

So wenden Sie einen Patch über die Befehlszeilenschnittstelle an:

1. Führen Sie den Befehl `list -u`[444] aus. Daraufhin werden alle Schemas, für die Upgrades zur Verfügung stehen, aufgelistet.
2. Führen Sie den Befehl `upgrade`[447] aus, um alle Patches zu installieren.

Installieren eines verfügbaren Schemas

Sie können Schemas entweder über die Benutzeroberfläche des Schema-Managers oder durch Senden der Schema-Manager-Installationsbefehle über die Befehlszeile installieren.

Anmerkung: Wenn das aktuelle Schema andere Schemas referenziert, werden auch die referenzierten Schemas installiert.

Über die Benutzeroberfläche

Um Schemas über die Benutzeroberfläche des Schema-Managers zu installieren, wählen Sie die gewünschten Schemas aus und klicken Sie auf **Anwenden**.

Sie können die gewünschten Schemas auch auf der Altova Website auswählen und eine herunterladbare `.altova_xmlschemas`-Datei generieren. Bei Doppelklick auf diese Datei wird der Schema-Manager aufgerufen, in dem die gewünschten Schemas bereits vorausgewählt sind. Sie müssen nur mehr auf **Anwenden** klicken.

Über das CLI

Um Schemas über die Befehlszeile zu installieren, rufen Sie den Befehl `install`[443] auf:

```
xmlschemamanager.exe install [options] Schema+
```

wobei es sich bei **Schema** um das/die gewünschte(n) Schema(s) bzw. eine **.altova_xmlschemas**-Datei handelt. Ein Schema wird von einem Identifier im Format **<name>-<version>** referenziert. (Die Identifier von Schemas werden angezeigt, wenn Sie den Befehl list[444] ausführen.) Sie können beliebig viele Schemas eingeben. Nähere Informationen dazu finden Sie unter der Beschreibung des Befehls install[443].

Anmerkung: Verwenden Sie unter Linux oder macOS den Befehl **sudo ./xmlschemamanager**.

Installation eines benötigten Schemas
Wenn Sie einen XML-Schema-Befehl in XMLSpy ausführen und XMLSpy erkennt, dass ein zur Ausführung des Befehls erforderliches Schema nicht vorhanden oder unvollständig ist, wird der Schema-Manager mit Informationen über das/die fehlende(n) Schema(s) aufgerufen. Sie können die gewünschten Schemas dann über den Schema-Manager direkt installieren.

Alle bereits installierten Schemas können jederzeit durch Aufruf des Schema-Managers über **Extras | Schema-Manager** über die Benutzeroberfläche des Schema-Managers angezeigt werden.

6.1.4 Deinstallieren eines Schemas, Zurücksetzen

Deinstallieren eines Schemas

Sie können Schemas entweder über die Benutzeroberfläche des Schema-Managers oder durch Senden der Schema-Manager-Deinstallationsbefehle über die Befehlszeile deinstallieren.

Anmerkung: Wenn das gewünschte Schema andere Schemas referenziert, so werden auch die referenzierten Schemas deinstalliert.

Über die Benutzeroberfläche
Um Schemas über die Benutzeroberfläche des Schema-Managers zu deinstallieren, deaktivieren Sie die Kontrollkästchen der entsprechenden Schemas und klicken Sie auf **Anwenden**. Daraufhin werden die ausgewählten Schemas und die davon referenzierten Schemas deinstalliert.

Um alle Schemas zu deinstallieren, klicken Sie auf **Auswahl für alle aufheben** und anschließend auf **Anwenden**.

Über das CLI
Um Schemas über die Befehlszeile zu deinstallieren, rufen Sie den Befehl uninstall[445] auf:

```
xmlschemamanager.exe uninstall [options] Schema+
```

wobei es sich beim Argument **Schema** ein zu deinstallierendes Schema oder eine **.altova_xmlschemas**-Datei handelt. Ein Schema wird von einem Identifier im Format **<name>-<version>** definiert. (Die Identifier von Schemas werden angezeigt, wenn Sie den Befehl list[444] ausführen.) Sie können beliebig viele Schemas eingeben. Nähere Informationen dazu finden Sie unter der Beschreibung des Befehls uninstall[445].

© 2016-2022 Altova GmbH

Anmerkung: Verwenden Sie unter Linux oder macOS den Befehl **`sudo ./xmlschemamanager`**.

Zurücksetzen des Schema-Managers

Sie können den Schema-Manager zurücksetzen. Damit werden alle installierten Schemas und das Cache-Verzeichnis entfernt.

- Klicken Sie auf der Benutzeroberfläche auf **Auswahl zurücksetzen**.
- Führen Sie über die Benutzeroberfläche den Befehl `reset`[445] aus.

Nachdem Sie diesen Befehl ausgeführt haben, muss der Befehl `initialize`[443] ausgeführt werden, um das Cache-Verzeichnis neu zu erstellen. Führen Sie alternativ dazu den Befehl `reset`[445] mit der Option `-i` aus.

Beachten Sie, dass mit `reset-i`[445] die Originalinstallation des Produkts wiederhergestellt wird, daher wird empfohlen, nach dem Zurücksetzen auch den Befehl `update`[446] auszuführen. Führen Sie alternativ dazu den Befehl `reset`[445] mit den Optionen `-i` und `-u` aus.

6.1.5 Befehlszeilenschnittstelle (CLI)

Um den Schema-Manager über die Befehlszeile aufzurufen, müssen Sie den Pfad zur ausführbaren Datei kennen. Standardmäßig befindet sich die ausführbare Schema-Manager-Datei hier:

```
C:\ProgramData\Altova\SharedBetweenVersions\XMLSchemaManager.exe
```

Anmerkung: Nachdem Sie auf Linux- und macOS-Systemen das Verzeichnis in dasjenige, das die ausführbare Datei enthält, geändert haben, können Sie die ausführbare Datei mit **`sudo ./xmlschemamanager`** aufrufen. Das Präfix **`./`** gibt an, dass sich die ausführbare Datei im aktuellen Verzeichnis befindet. Das Präfix **`sudo`** gibt an, dass der Befehl mit Root-Rechten ausgeführt werden muss.

Befehlszeilensyntax

Die allgemeine Syntax zur Verwendung der Befehlszeile lautet folgendermaßen:

```
<exec> -h | --help | --version | <command> [options] [arguments]
```

Der senkrechte Balken **`|`** im Codefragment oben trennt eine Gruppe einander gegenseitig ausschließender Elemente. Optionale Elemente stehen innerhalb von eckigen Klammern **`[]`**. Im Prinzip können Sie den Pfad zur ausführbaren Datei, gefolgt von entweder `--h`, `--help` oder `--version`-Optionen oder gefolgt von einem Befehl eingeben. Jeder Befehl kann Optionen und Argumente haben. Die Liste der Befehle wird in den folgenden Abschnitten beschrieben.

6.1.5.1 help

Mit diesem Befehl erhalten Sie Hilfe zu Befehlen zur ausführbaren Schema-Manager-Datei.

Syntax

```
<exec> help [Befehl]
```

`[Befehl]` ist hierbei ein optionales Argument zur Angabe jedes beliebigen gültigen Befehlsnamens.

Beachten Sie dazu Folgendes:

- Sie können die Hilfe zu einem Befehl auch durch Eingabe des Befehls, gefolgt von **`-h`** oder **`--help`** aufrufen, z.B: **`<exec> list -h`**
- Wenn Sie **`-h`** oder **`--help`** direkt nach dem Namen der ausführbaren Datei und vor einem Befehl eingeben, wird die allgemeine Hilfe (und nicht die Hilfe zu einem bestimmten Befehl) angezeigt, z.B: **`<exec> -h list`**

Beispiel

Mit dem folgenden Befehl wird Hilfe zum Befehl `list` angezeigt:

```
xmlschemamanager help list
```

6.1.5.2 info

Mit diesem Befehl werden ausführliche Informationen über die einzelnen als `Schema`-Argument angegebenen Schemas angezeigt. Darin enthalten sind Titel, Version, Beschreibung, Herausgeber der jeweils angegebenen Schemas und davon referenzierte Schemas sowie die Information, ob das Schema installiert ist oder nicht.

Syntax

```
<exec> info [options] Schema+
```

- Das Argument **`Schema`** ist der Name eines Schemas oder Teil eines Schemanamens. (Die Paket-ID eines Schemas und detaillierte Informationen über ihren Installationsstatus erhalten Sie mit dem Befehl list 444.)
- Mit **`<exec> info -h`** können Sie die Hilfe zum Befehl anzeigen.

Beispiel

Mit dem folgenden Befehl werden Informationen über das jeweils neueste **`DocBook-DTD`**- und **`NITF`**-Schemas angezeigt.

```
xmlschemamanager info doc nitf
```


© 2016-2022 Altova GmbH

6.1.5.3 initialize

Mit diesem Befehl wird die Schema-Manager-Umgebung initialisiert. Sie erstellen damit ein Cache-Verzeichnis, in dem Informationen über alle Schemas lokal gespeichert werden. Die Initialisierung erfolgt automatisch bei der ersten Installation einer Schema-fähigen Altova-Applikation. Normalerweise muss dieser Befehl nicht ausgeführt werden. Nach Ausführung des **reset**-Befehls ist dies allerdings erforderlich.

Syntax

```
<exec> initialize | init [options]
```

Optionen
Für den Befehl **initialize** stehen die folgenden Optionen zur Verfügung:

`--silent, --s`	Nur Fehlermeldungen anzeigen. Der Standardwert ist **false**.
`--verbose, --v`	Anzeige detaillierter Informationen während der Ausführung. Der Standardwert ist **false**.
`--help, --h`	Anzeige der Hilfe zum Befehl.

Beispiel

Mit dem folgenden Befehl wird der Schema-Manager initialisiert:

```
xmlschemamanager initialize
```

6.1.5.4 install

Mit diesem Befehl installieren Sie ein oder mehrere Schemas.

Syntax

```
<exec> install [options] Schema+
```

Um mehrere Schemas zu installieren, fügen Sie das Argument **Schema** mehrmals hinzu.

Als **Schema**-Argument kann eines der folgenden verwendet werden:

- Ein Schema-Identifier (im Format **<name>-<version>**, z.B: **cbcr-2.0**). Um die Schema-Identifier der gewünschten Schemas zu eruieren, führen Sie den Befehl list[444] aus. Sie können auch einen abgekürzten Identifier verwenden, sofern dieser eindeutig ist, z.B. **docbook**. Falls Sie einen abgekürzten Identifier verwenden, wird die neueste Version dieses Schemas installiert.
- Der Pfad zu einer von der Altova-Website heruntergeladenen **.altova_xmlschemas**-Datei. Informationen zu diesen Dateien finden Sie in der *Einführung zu Schema-Manager: Funktionsweise*[433].

Optionen
Für den Befehl **install** stehen die folgenden Optionen zur Verfügung:

`--silent, --s`	Nur Fehlermeldungen anzeigen. Der Standardwert ist **`false`**.
`--verbose, --v`	Anzeige detaillierter Informationen während der Ausführung. Der Standardwert ist **`false`**.
`--help, --h`	Anzeige der Hilfe zum Befehl.

Beispiel

Mit dem folgenden Befehl werden das CBCR 2.0 (Country-By-Country Reporting)-Schema und die neueste DocBook-DTD installiert:

```
xmlschemamanager install cbcr-2.0 docbook
```

6.1.5.5 list

Mit diesem Befehl werden vom Schema-Manager verwaltete Schemas aufgelistet. In der Liste wird eine der folgenden Informationen angezeigt:

- alle verfügbaren Schemas
- Schemas, die im Namen den im Argument **`Schema`** angegebenen String enthalten
- nur installierte Schemas
- Nur Schemas, für die ein Upgrade installiert werden kann

Syntax

```
<exec> list | ls [options] Schema?
```

Wenn kein **`Schema`**-Argument angegeben wird, werden alle verfügbaren Schemas aufgelistet. Andernfalls werden die durch die angegebenen Optionen definierten Schemas aufgelistet (*siehe Beispiel unten*). Beachten Sie, dass Sie das Argument **`Schema`** mehrfach angeben können.

Optionen
Für den Befehl **`list`** stehen die folgenden Optionen zur Verfügung:

`--installed, --i`	Auflisten nur der installierten Schemas. Der Standardwert ist **`false`**.
`--upgradeable, --u`	Auflisten nur derjenigen Schemas, für die Upgrades (Patches) zur Verfügung stehen. Der Standardwert ist **`false`**.
`--help, --h`	Anzeige der Hilfe zum Befehl.

Beispiele

- Um alle verfügbaren Schemas aufzulisten, führen Sie den folgenden Befehl aus: **`xmlschemamanager list`**
- Um nur installierte Schemas aufzulisten, führen Sie **`xmlschemamanager list -i`** aus.

© 2016-2022 Altova GmbH

- Um Schemas, die in ihrem Namen entweder "doc" oder "nitf" enthalten, aufzulisten, führen Sie **xmlschemamanager list doc nitf** aus.

6.1.5.6 reset

Mit diesem Befehl werden alle installierten Schemas und das Cache-Verzeichnis entfernt. Ihre Schemaumgebung wird vollständig zurückgesetzt. Nachdem Sie diesen Befehl ausgeführt haben, muss der Befehl initialize[443] ausgeführt werden, um das Cache-Verzeichnis neu zu erstellen. Führen Sie alternativ dazu den Befehl **reset** mit der Option **-i** aus. Da mit **reset-i** die Originalinstallation des Produkts wiederhergestellt wird, wird empfohlen, nach dem Zurücksetzen und Initialisieren auch den Befehl update[446] auszuführen. Führen Sie alternativ dazu den Befehl **reset** mit den Optionen **-i** und **-u** aus.

Syntax

```
<exec> reset [Optionen]
```

Optionen
Für den Befehl **reset** stehen die folgenden Optionen zur Verfügung:

`--init, --i`	Initialisierung des Schema-Managers nach dem Zurücksetzen. Der Standardwert ist **false**.
`--update, --u`	Aktualisiert die Liste der verfügbaren Schemas im Cache. Der Standardwert ist **false**.
`--silent, --s`	Nur Fehlermeldungen anzeigen. Der Standardwert ist **false**.
`--verbose, --v`	Anzeige detaillierter Informationen während der Ausführung. Der Standardwert ist **false**.
`--help, --h`	Anzeige der Hilfe zum Befehl.

Beispiele

- Um den Schema-Manager zurückzusetzen, führen Sie den folgenden Befehl aus: **xmlschemamanager reset**
- Um den Schema-Manager zurückzusetzen und ihn zu initialisieren, führen Sie **xmlschemamanager reset -i** aus.
- Um den Schema-Manager zurückzusetzen, ihn zu initialisieren und seine Schemaliste zu aktualisieren, führen Sie **xmlschemamanager reset -i -u** aus.

6.1.5.7 uninstall

Mit diesem Befehl deinstallieren Sie ein oder mehrere Schemas. Standardmäßig werden auch alle Schemas, die vom der aktuellen Schema referenziert werden, deinstalliert. Um nur das aktuelle Schema zu deinstallieren und die referenzierten Schemas beizubehalten, setzen Sie die Option `--k`.

Syntax

```
<exec> uninstall [options] Schema+
```

Um mehrere Schemas zu deinstallieren, fügen Sie das Argument **Schema** mehrmals hinzu.

Als **Schema**-Argument kann eines der folgenden verwendet werden:

- Ein Schema-Identifier (im Format **<name>-<version>**, z.B: **cbcr-2.0**). Um die Schema-Identifier der installierten Schemas zu eruieren, führen Sie den Befehl list -i[444] aus. Sie können auch einen abgekürzten Schemanamen verwenden, sofern dieser eindeutig ist, z.B. **docbook**. Falls Sie einen abgekürzten Namen verwenden, werden alle Schemas, die die Abkürzung in ihrem Namen enthalten, deinstalliert.
- Der Pfad zu einer von der Altova-Website heruntergeladenen **.altova_xmlschemas**-Datei. Informationen zu diesen Dateien finden Sie in der *Einführung zu Schema-Manager: Funktionsweise*[433].

Optionen
Für den Befehl **uninstall** stehen die folgenden Optionen zur Verfügung:

--keep-references, --k	Definieren Sie diese Option, um referenzierte Schemas beizubehalten. Der Standardwert ist **false**.
--silent, --s	Nur Fehlermeldungen anzeigen. Der Standardwert ist **false**.
--verbose, --v	Anzeige detaillierter Informationen während der Ausführung. Der Standardwert ist **false**.
--help, --h	Anzeige der Hilfe zum Befehl.

Beispiel

Mit dem folgenden Befehl werden die Schemas CBCR 2.0 und EPUB 2.0 und deren Abhängigkeiten deinstalliert:

```
xmlschemamanager uninstall cbcr-2.0 epub-2.0
```

Mit dem folgenden Befehl wird das **eba-2.10**-Schema, nicht aber die davon referenzierten Schemas deinstalliert:

```
xmlschemamanager uninstall --k cbcr-2.0
```

6.1.5.8 update

Mit diesem Befehl wird die Liste der über den Online-Speicher verfügbaren Schemas abgefragt und das lokale Cache-Verzeichnis wird aktualisiert. Normalerweise muss dieser Befehl nur ausgeführt werden, wenn Sie reset[445] und initialize[443] ausgeführt haben.

Syntax

```
<exec> update [options]
```


© 2016-2022 Altova GmbH

Optionen
Für den Befehl **update** stehen die folgenden Optionen zur Verfügung:

`--silent, --s`	Nur Fehlermeldungen anzeigen. Der Standardwert ist **false**.
`--verbose, --v`	Anzeige detaillierter Informationen während der Ausführung. Der Standardwert ist **false**.
`--help, --h`	Anzeige der Hilfe zum Befehl.

Beispiel

Mit dem folgenden Befehl wird der lokale Cache mit der Liste der neuesten Schemas aktualisiert:

```
xmlschemamanager update
```

6.1.5.9 upgrade

Mit diesem Befehl werden alle installierten Schemas, für die ein Upgrade installiert werden kann, auf die neueste verfügbare *Patch*-Version aktualisiert. Um herauszufinden, welche Schemas aktualisiert werden können, starten Sie den Befehl `list-u`[444].

Anmerkung: Der Befehl **upgrade** entfernt ein veraltetes Schema, falls keine neuere Version zur Verfügung steht.

Syntax

```
<exec> upgrade [Optionen]
```

Optionen
Für den Befehl **upgrade** stehen die folgenden Optionen zur Verfügung:

`--silent, --s`	Nur Fehlermeldungen anzeigen. Der Standardwert ist **false**.
`--verbose, --v`	Anzeige detaillierter Informationen während der Ausführung. Der Standardwert ist **false**.
`--help, --h`	Anzeige der Hilfe zum Befehl.

6.2 DTDs

Ein DTD-Dokument kann in der Text- und der Grid-Ansicht bearbeitet werden. Sie können die Standardansicht im Dialogfeld "Optionen" im Abschnitt "Dateiarten" 1606 festlegen.

Textansicht

In der Textansicht wird das Dokument mit Syntaxfärbung angezeigt und muss eingetippt werden. Unten sehen Sie ein Beispiel für ein DTD-Fragment.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Element declarations-->
<!ELEMENT document (header, para+, img+, link+)>
<!ELEMENT header (#PCDATA)>
<!ELEMENT img EMPTY>
  <!ATTLIST img
    src CDATA #REQUIRED
  >

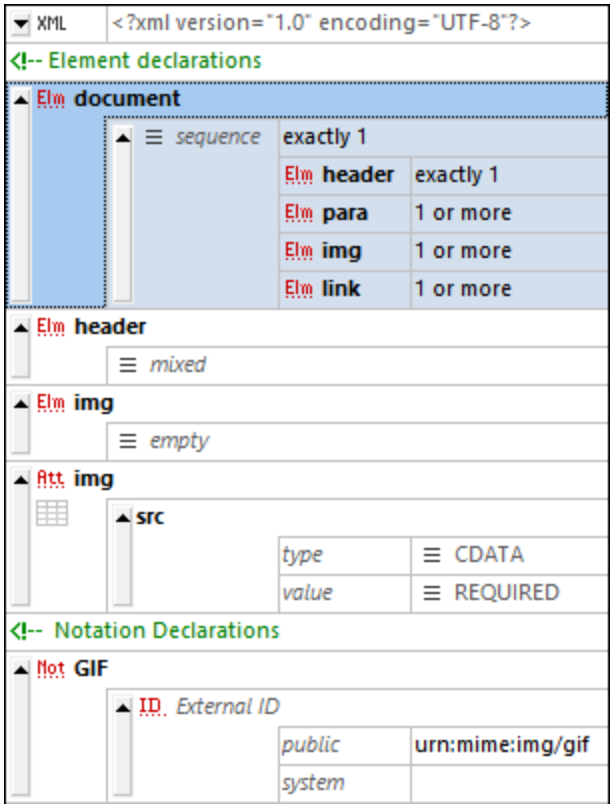
<!-- Notation Declarations -->
<!NOTATION GIF PUBLIC "urn:mime:img/gif">
```

Die Einrückung wird durch Einrücklinien gekennzeichnet. Am besten verwendet man dazu die Tabulatortaste. Der Abstand für die Einrückung kann im Dialogfeld Einstellungen für Textansicht 148 festgelegt werden.

Grid-Ansicht

In der Grid-Ansicht wird das DTD-Dokument als Tabelle dargestellt. In der Abbildung unten sehen Sie, wie die oben aufgelistete DTD in der Grid-Ansicht angezeigt wird.

© 2016-2022 Altova GmbH



Bearbeiten der DTD-Struktur

- Wenn sich der Cursor in einer Zelle befindet, können Sie Nodes einfügen oder anhängen oder einen Child-Node über das Kontextmenü oder das Menü **XML** hinzufügen.
- Klicken Sie links oben in der Zelle auf das Typsymbol des Node, um den Node-Typ zu ändern.
- Durch Klick auf das entsprechende Symbol und Auswahl der gewünschten Option können Sie das Inhaltsmodell (*sequence, mixed, empty,* usw.) und den Occurrence Modifier (*exactly 1, 1 or more,* usw.) eines Node ändern.
- Sie können Nodes auch mit Drag-and-Drop an eine andere Stelle im Dokument verschieben und Nodes an eine neue Stelle kopieren.

Bearbeiten von DTD-Werten

- Um Werte wie Element- und Attributnamen und Kommentare zu bearbeiten, doppelklicken Sie in die Zelle und bearbeiten Sie sie.

Symbolleiste der Grid-Ansicht
Über die Symbolleiste der Grid-Ansicht können Sie das Einstellungsdialogfeld der Ansicht sowie Befehle zum Einstellen des Zoom-Faktors und des Zeilenumbruchs aufrufen.

DTD-Funktionen in XMLSpy

XMLSpy bietet die folgenden äußerst nützlichen Funktionen:

- *Konvertieren einer DTD in ein XML-Schema:* Mit dem Befehl **DTD/Schema | DTD in Schema konvertieren**[1376] können DTDs in XML-Schemas konvertiert werden.
- *Generieren einer XML-Beispieldatei anhand einer DTD:* Mit dem Befehl **DTD/Schema | XML/JSON-Beispieldatei generieren**[1381] kann auf Basis der aktiven DTD ein XML-Dokument generiert werden.

© 2016-2022 Altova GmbH

6.3 XML-Schemas

XML-Dokumente können in der Textansicht, der Grid-Ansicht und der Schema-Ansicht bearbeitet werden. Sie können die Standardansicht, in der XML-Schemas geöffnet werden, im Dialogfeld "Optionen" im Abschnitt „Dateiarten" 1606 festlegen. Sie können während der Bearbeitung zwischen den Ansichten wechseln und die jeweils am besten geeignete Ansicht wählen. XML-Schema-Dokumente werden normalerweise mit der Erweiterung `.xsd` oder `.xs` gespeichert.

Bearbeitung in der Textansicht

In der Textansicht wird ein XML-Schema als XML-Dokument bearbeitet; die Bearbeitungsfunktionen, die für XML-Dokumente zur Verfügung stehen 293, sind auch für XML-Schemas verfügbar. Wie bei allen XML-Dokumenten, bei denen das Schema erkannt wird und zugänglich ist, finden Sie in den Eingabehilfen der Textansicht 165 die Elemente, die an der Cursorposition hinzugefügt werden können.

Bearbeiten in der Grid-Ansicht

In der Grid-Ansicht wird ein XML-Schema als XML-Dokument bearbeitet; die Bearbeitungsfunktionen, die für XML-Dokumente zur Verfügung stehen 169, sind auch für XML-Schemas verfügbar. Wenn ein Element in der Grid-Ansicht ausgewählt ist, finden Sie in den Eingabehilfen der Grid-Ansicht 302 die Elemente, die an der Cursorposition hinzugefügt werden können.

Bearbeiten in der Schema-Ansicht

Die Schema-Ansicht ist eine grafische Ansicht zur Erstellung von Schemas. Während Sie das Schema in der Schema-Ansicht erstellen/bearbeiten, generiert XMLSpy im Hintergrund des entsprechende Textdokument. Eine Anleitung zum Erstellen und Bearbeiten von XML-Schema-Dokumenten in der Schema-Ansicht finden Sie im Abschnitt Bearbeitungsansichten | Schema-Ansicht 171.

Altova Website: XML Schema Editor

XML-Schemafunktionen in XMLSpy

XMLSpy bietet zusätzlich dazu die folgenden nützlichen Funktionen:

- *Konvertieren von XML-Schemas in DTDs:* Mit dem Befehl **DTD/Schema | DTD/Schema konvertieren** 1379 **kann** ein XML-Schema in eine DTD konvertiert werden.
- *Generieren einer XML-Beispieldatei anhand eines XML-Schemas:* Mit dem **Befehl DTD/Schema | XML-Beispieldatei erzeugen** 1381 kann ein XML-Dokument generiert werden, das auf dem aktiven XML-Schema basiert. Beispielwerte können auch für Elemente und Attribute in der XML-Beispieldatei definiert werden.
- XML-Signaturen 417 für XML-Schemadateien (`.xsd`) in der Schema-Ansicht können als externe Signaturdateien erstellt werden. Eine Anleitung zum Arbeiten mit Signaturen finden Sie im Abschnitt XML-Signaturen 417.

6.4 Schemamodule

Sie können eine oder mehrere Komponenten eines XML-Schemas in Form einer separaten Schema-Datei, eines so genannten Schemamoduls speichern. Der Vorteil der Verwendung kleinerer Schemamodule bei der Erstellung des größeren Schemas (mittels Include) liegt darin, dass die kleineren Dateien übersichtlicher zu verwalten sind als ein einziges vollständiges Schema.

Ein mögliches Arbeitsszenario in der Schema-Ansicht, bei dem verschiedene Aspekte der Schemamodulfunktion beschrieben wird, ist das folgende:

1. Erstellen Sie ein Schemamodul, das eine oder mehrere Komponenten des aktiven Schemas enthält. Eine Beschreibung dazu finden Sie weiter unten 452.
2. Erstellen Sie je nach Bedarf weitere Schemamodule.
3. Inkludieren Sie das/die neu erstellten Schemamodul(e), um daraus das größere Schema zu bilden. Dies geschieht, indem Sie im Fenster "Schemaübersicht" 177 für jedes Schemamodul eine Include-Komponente anhängen oder einfügen und die neu erstellte Schemamoduldatei auswählen.
4. Löschen Sie alle Komponenten, die im ursprünglichen vollständigen Schema vorhanden waren, die aber jetzt aufgrund der inkludierten Module doppelt vorhanden sind.

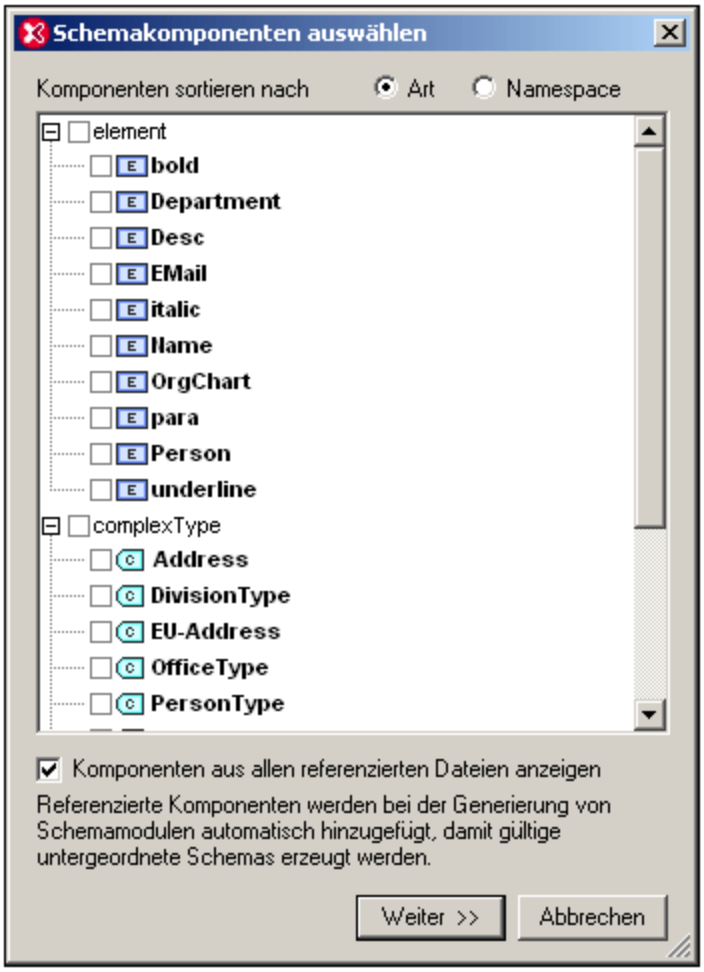
Sie können in der Schema-Ansicht auch umgekehrt vorgehen und die inkludierten Schemamodule auf eine Ebene bringen, sodass (i) die in den Schemamodulen enthaltenen Komponenten direkt zum Hauptschema hinzugefügt werden und (ii) die inkludierten Schemamodule aus dem Hauptschema gelöscht werden. Eine Anleitung zum Nivellieren eines Schemas finden Sie weiter unten 454.

Erstellen von Schemamodulen

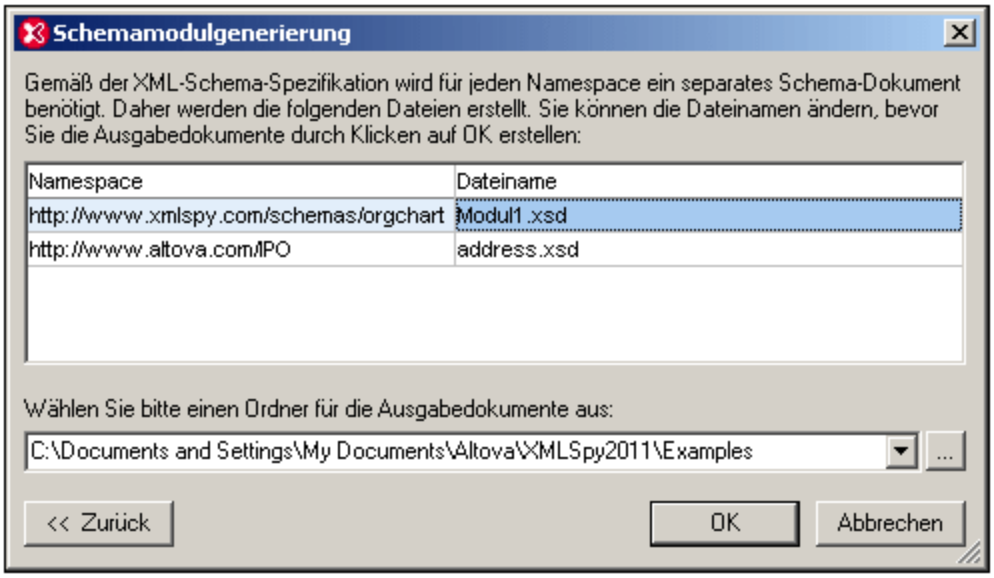
Um ein Schemamodul zu erstellen, gehen Sie folgendermaßen vor:

1. Wählen Sie, während das gewünschte XML-Schema in der Schema-Ansicht aktiv ist, den Befehl **Schema-Design | Schemamodul erstellen**. Daraufhin wird das Dialogfeld "Schemakomponenten auswählen" (*Abbildung unten*) angezeigt.
2. Wählen Sie im Dialogfeld die Komponente bzw. Komponenten aus, die als einzelne Schemamodule angelegt werden sollen und klicken Sie auf **Weiter**. (Beachten Sie das Kontrollkästchen unterhalb des Fensters, über das auch Komponenten aus allen referenzierten Dateien für die Auswahl aufgelistet werden können.)

© 2016-2022 Altova GmbH



3. Geben Sie im Dialogfeld "Schemamodulgenerierung" (*Abbildung unten*), das nun angezeigt wird, den/die Namen ein, den/die die einzelnen Dateien der Schemamodulpakete erhalten sollen. Außerdem müssen Sie den Ordner angeben, in dem die neuen Schemamoduldateien gespeichert werden sollen. Ein Schemamodulpaket könnte mehrere Dateien enthalten, wenn es sich bei einer oder mehreren der erstellten Komponenten im Originalschema um importierte Komponenten handelt. Für jeden Namespace im Schemamodul wird eine eigene Schemadatei angelegt. Die Dateinamen, die im Dialogfeld angezeigt werden, sind standardmäßig die Namen der Originaldateien. Da Sie die Originaldateien aber nicht überschreiben dürfen, verwenden Sie neue Dateinamen, wenn Sie die Dateien im selben Ordner wie die Originaldateien speichern möchten.



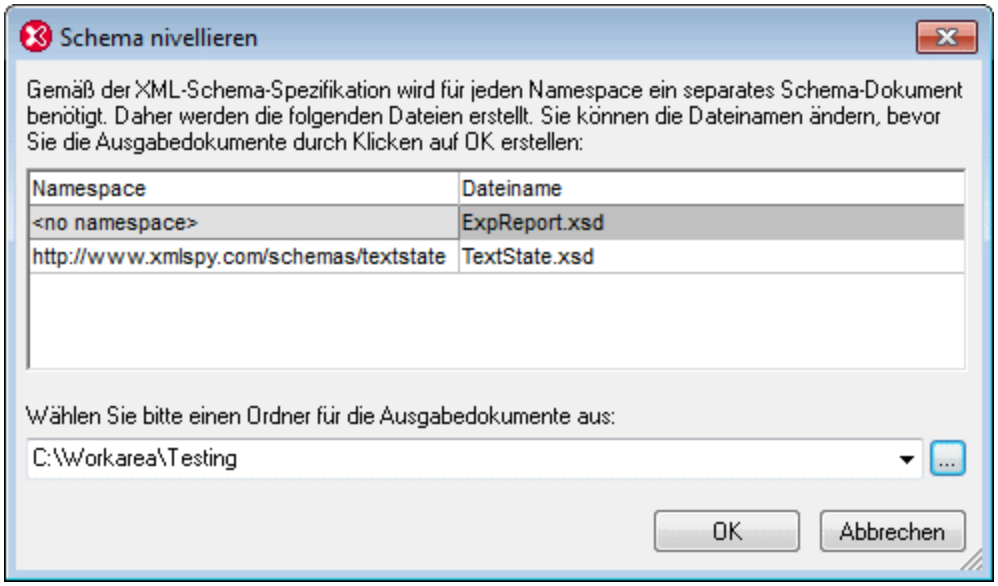
4. Wenn Sie auf **OK** klicken, wird die Schemamoduldatei mit dem Namespace, der dem der aktiven Datei entspricht, in der Schema-Ansicht geöffnet. Alle anderen Dateien im Paket werden erstellt, aber nicht in der Schema-Ansicht geöffnet.

Nivellieren eines Schemas

Beim Nivellieren des aktiven Schemas in der Schema-Ansicht (i) werden die Komponenten aller inkludierten Schemas als globale Komponenten des aktiven Schemas hinzugefügt und (ii) die inkludierten Schemas gelöscht.

Um das aktive Schema zu nivellieren, wählen Sie den Befehl **Schema-Design | Schema nivellieren**. Daraufhin wird das Dialogfeld "Schema nivellieren" (*Abbildung unten*) angezeigt, in dem die Namen separater Dateien - eine für jeden Namespace, der im nivellierten Schema enthalten sein wird - aufgelistet sind. Diese Standardnamen sind mit den Namen der Originaldateien identisch. Da die Originaldateien aber nicht überschrieben werden dürfen, müssen die Dateinamen geändert werden, wenn Sie die Dateien im selben Ordner wie die aktive Datei speichern möchten. Sie können zu einem Ordner navigieren, in dem das nivellierte Schema und die damit verknüpften Dateien gespeichert werden sollen.

© 2016-2022 Altova GmbH



Wenn Sie auf **OK** klicken, wird die nivellierte Schema-Datei in der Schema-Ansicht geöffnet.

6.5 Schemaregeln

Bei Schemaregeln handelt es sich um eine Gruppe von Regeln zum Validieren eines Schemas. So kann eine Regel z.B: definieren, dass im XML-Schema definierte Attributnamen mit einem Kleinbuchstaben beginnen müssen oder dass ein bestimmter complexType nur von einem bestimmte Typ aus erweitert werden kann.

Eine Gruppe von Schemaregeln wird in einer Schemaregeldatei, einer XML-Datei (`.xml`) gespeichert. XMLSpy enthält einen Schemaregel-Editor[458], in dem Sie Schemaregeln grafisch bearbeiten können. Anschließend kann eine Schemaregeldatei in XMLSpy einem XML-Schema zugewiesen werden[456]. Bei Auswahl des Befehls **XML | Validieren (F8)** wird das XML-Schema in der Schema-Ansicht anhand der zugewiesenen Regeln validiert.

6.5.1 Verwalten von Schemaregeln

Einem aktiven XML-Schema (`.xsd`-Datei) können über das Register "Schema" des Fensters "Info" eine oder mehrere Schemaregeldateien zugewiesen werden. (*Abbildung unten*).

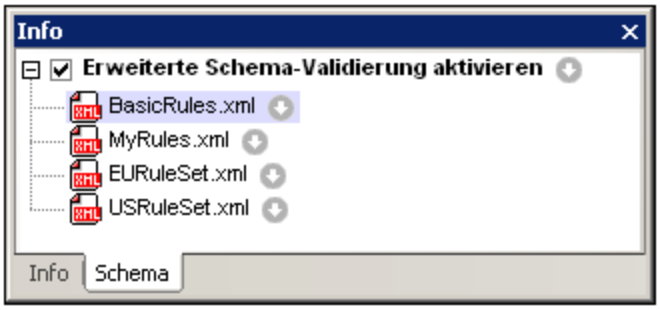


Hinzufügen von Schemaregeln für die erweiterte Validierung

Um eine Schemaregeldatei hinzuzufügen, klicken Sie auf die Schaltfläche "Kontextmenü" . Dadurch wird ein Menü aufgerufen (*siehe Abbildung oben*), in dem Sie auswählen können, wie die Schemaregeldateien zum XML-Schema hinzugefügt werden können. Es stehen die folgenden Optionen zur Verfügung:

- *Vordefinierte Regeln hinzufügen:* Sie können aus einer Liste vordefinierter Schemaregeldateien, die mit XMLSpy installiert wurden, auswählen. Diese Schemaregeldateien sind im Ordner `Extended Schema Validation` des XMLSpy Applikationsordners gespeichert. Alle Regeldateien, die zu diesem Ordner hinzugefügt werden, werden im Dialogfeld "Vordefinierten Regeln hinzufügen" angezeigt und können hinzugefügt werden.
- *Vorhandene Regeln suchen:* Hier können Sie zu einer nicht vordefinierten Schemaregeldatei navigieren.
- *Neue Regeln erstellen:* Ruft den Schemaregel-Editor auf, in dem Sie die Schemaregeln in einer Schemaregeldatei bearbeiten können. Eine Anleitung zum Arbeiten mit dem Schemaregel-Editor finden Sie im Abschnitt Definieren von Schemaregeln[458]. Nachdem Sie eine mit diesem Befehl erstellte Schemaregeldatei gespeichert haben, wird sie zur Liste für das aktive XML-Schema hinzugefügt (*siehe Abbildung unten*).

© 2016-2022 Altova GmbH



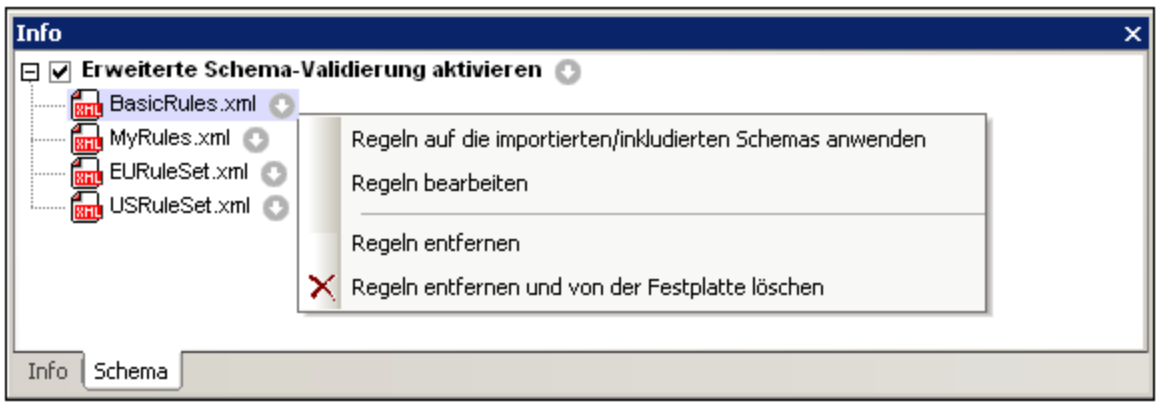
Sie können beliebig viele Schemaregeldateien hinzufügen (*siehe Abbildung oben*). Wenn einem XML-Schema mehrere Schemaregeldateien zugewiesen sind, werden alle hinzugefügten Schemaregeldateien bei der Validierung des XML-Schemas **(XML | Validieren)** in der Schema-Ansicht verwendet.

Aktivieren und Deaktivieren der erweiterten Schemavalidierung

Die erweiterte Schemavalidierung kann durch Anklicken des Kontrollkästchens "Erweiterte Schemavalidierung aktivieren" aktiviert bzw. deaktiviert werden.

Bearbeiten und Entfernen von Schemaregeldateien

Die einzelnen Regeldateien, die einem XML-Schema zugewiesen sind, können über das Kontextmenü, das durch Klicken auf die Schaltfläche "Kontextmenü"  aufgerufen wird (*Abbildung unten*), verwaltet werden.



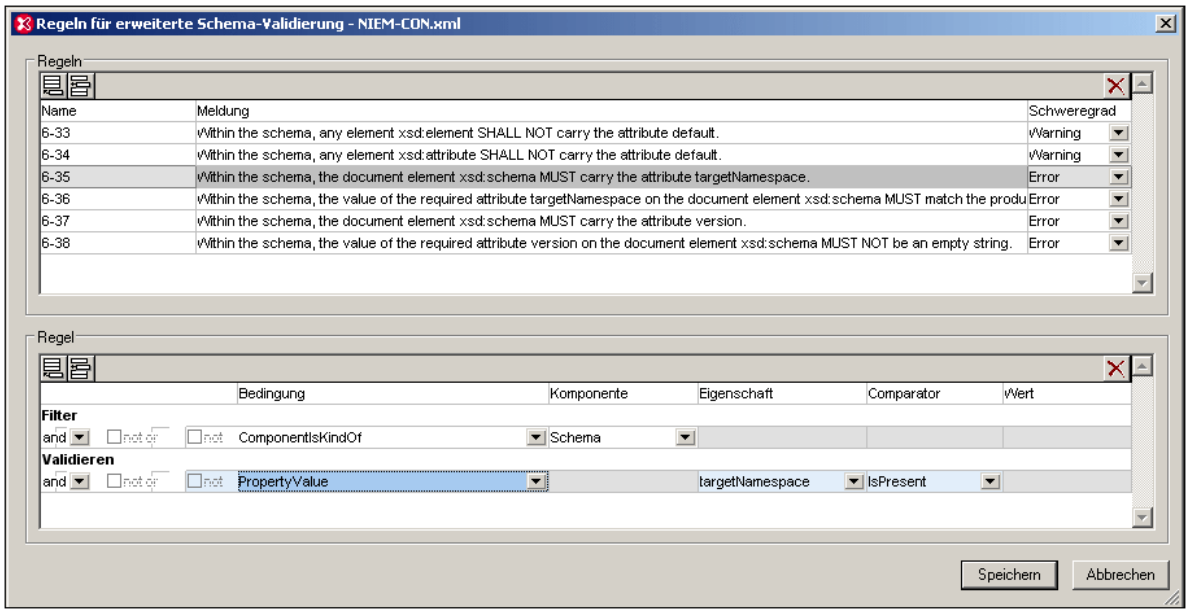
Es stehen die folgenden Optionen zur Verfügung:

- *Regeln auf die importierten/inkludierten Schemas anwenden:* Wenn eine Regeldatei angewendet wird, werden die Regeln darin für alle Schemas verwendet, die in das Hauptschema importiert oder inkludiert werden.
- *Regel bearbeiten:* Öffnet die Schemaregeldatei im Schemaregel-Editor.
- *Regeln entfernen:* Entfernt die Regeldatei aus der Liste der hinzugefügten Regeldateien.
- *Regeln entfernen und von der Festplatte löschen:* Dieser Befehl ist bei allen nicht vordefinierten Schemaregeldateien aktiv. Er entfernt die Regeldatei aus der Liste der hinzugefügten Regeldateien und löscht die Regeldatei selbst.

6.5.2 Definieren von Schemaregeln

Eine Schemaregeldatei kann zur Bearbeitung im Schemaregel-Editor (*Abbildung unten*) geöffnet werden. Anschließend können Sie die Schemaregeln in dieser Schemaregeldatei erstellen, bearbeiten und löschen. Um eine Schemaregeldatei im Schemaregel-Editor zu öffnen, gehen Sie folgendermaßen vor:

1. Wählen Sie die Schemaregeln in der Liste der Schemaregeln im Fenster "Info" aus.
2. Klicken Sie auf die Schaltfläche "Kontextmenü" dieser Regeldatei.
3. Wählen Sie nun im angezeigten Kontextmenü den Befehl "Regel bearbeiten".



Das Dialogfeld "Schemaregel-Editor" hat zwei Bereiche:

- Einen Bereich "Regeln" (im oberen Bereich des Editors), in dem Sie Regeln hinzufügen und löschen können. Durch Klicken auf die Schaltfläche **Anhängen** oder **Einfügen** in der linken oberen Ecke des Bereichs können Sie eine leere Zeile für eine Regel anhängen bzw. einfügen. Um eine Regel zu löschen, wählen Sie sie aus und klicken Sie rechts oben im Bereich auf die Schaltfläche **Löschen**. Jede Regel in diesem Bereich hat einen Namen, einen beschreibenden Text und einen Schweregrad (bei einem Verstoß gegen die Regel, kann festgelegt werden, dass bei der Validierung ein Fehler oder eine Warnmeldung zurückgegeben wird).
- einem Bereich "Regel" (im unteren Bereich des Editors). In diesem Bereich werden die Einzelheiten zu der Regel, die im oberen Bereich "Regeln" ausgewählt wurde, angezeigt. Hier können Sie die Einzelheiten der Regel bearbeiten. Nähere Informationen zum Definieren von Regeln finden Sie weiter unten im Abschnitt "Definieren einer Regel 458".

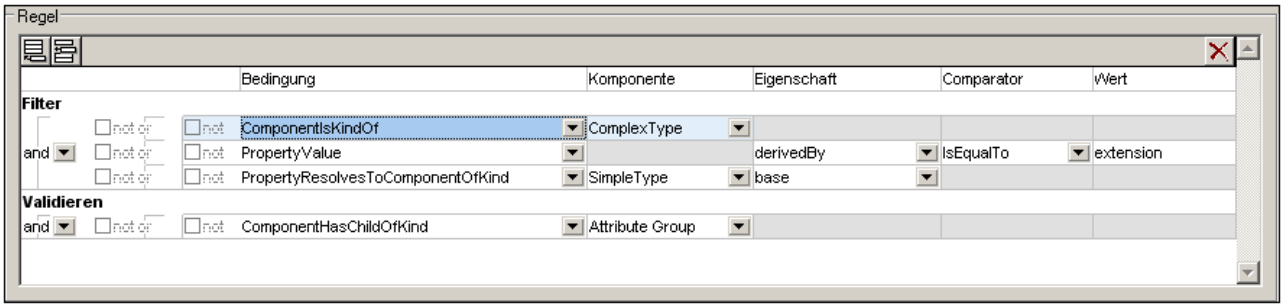
Wenn Sie die Regeln in einer Schemaregeldatei bearbeitet haben, klicken Sie auf **Speichern**, um die Regeln in der Schemaregeldatei zu speichern.

Definieren einer Regel

Um eine Regel zu definieren oder zu bearbeiten, wählen Sie die Regel in der Liste im oberen Bereich "Regeln" aus. Die Definition der Regel wird im Bereich "Regel" angezeigt, wo sie auch bearbeitet werden kann. In der

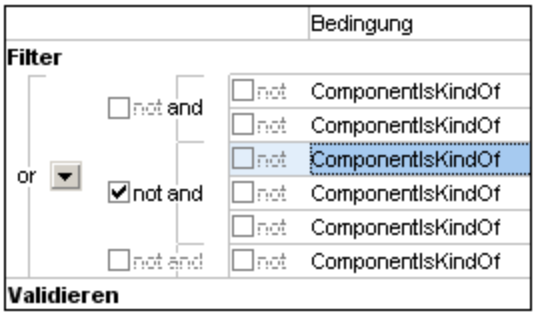
 © 2016-2022 Altova GmbH

Abbildung unten sehen Sie eine Regel, die folgendermaßen definiert werden kann: *Wenn ein complexType eine Erweiterung eines simpleType ist, muss er ein Child der Art AttributeGroup haben.*



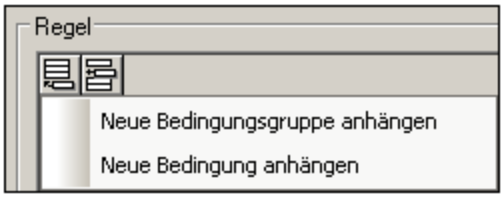
Validierungsbedingungen und Filterbedingungen:

- Jede Regel enthält Validierungsbedingungen und Filterbedingungen (*siehe erste Spalte in der Abbildung oben*).
- Alle Filterbedingungen müssen "true" sein, damit die Validierungsbedingung überprüft werden kann.
- Jede Bedingung (ob Validierungsbedingung oder Filterbedingung) besteht aus einer oder mehreren Bedingungsgruppen, wobei jede Bedingungsgruppe eine oder mehrere Bedingungen enthält. Die Validierungsbedingung in der Abbildung oben enthält eine Bedingungsgruppe mit einer Bedingung und die Filterbedingung enthält drei Bedingungsgruppen mit je einer Bedingung. Die Filterbedingung in der Abbildung unten enthält drei Bedingungsgruppen: Die erste enthält zwei Bedingungen, die zweite drei und die dritte eine.



- Jede einzelne Bedingung kann durch Aktivieren des Kontrollkästchens *Not* (links neben der Bedingung) negiert werden.
- In einer Bedingungsgruppe geben die logischen Verbindungen and oder or an, ob alle Bedingungen in der Gruppe oder nur eine Bedingung in der Bedingungsgruppe "true" sein muss, damit die gesamte Bedingungsgruppe als "true" ausgewertet wird. Auf der Benutzeroberfläche befinden sich diese logischen Operatoren innerhalb zweier Spalten mit logischen Operatoren.
- Jede einzelne Bedingungsgruppe kann durch Aktivieren des Kontrollkästchens *Not* (links neben dem logischen Operator der Bedingungsgruppe) negiert werden.
- Der äußere logische Konnektor and oder or gibt an, ob alle Bedingungsgruppen in der Gruppe (Validierungsbedingung oder Filterbedingung) oder nur eine Bedingungsgruppe "true" sein muss, damit die gesamte Gruppe (Validierung oder Filter) als "true" ausgewertet wird.
- Logische Konnektoren können durch Auswahl der entsprechenden Option in der Auswahlliste für den äußeren logischen Konnektor (den Konnektor der Bedingungsgruppe) geändert werden. Die Werte der inneren logischen Konnektoren (der Konnektoren für die Bedingungen innerhalb einer Bedingungsgruppe) werden alle auf den dem äußeren logischen Konnektor entgegengesetzten Wert gesetzt.

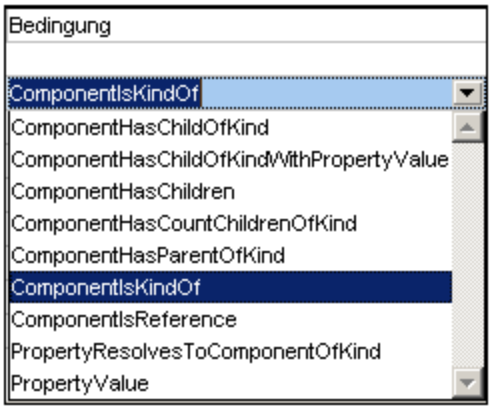
- Eine Bedingungsgruppe oder Bedingung kann auf folgende Arten an die ausgewählte Bedingung angehängt oder in diese eingefügt werden: Wählen Sie die Bedingung aus und klicken Sie auf die Schaltfläche **Anhängen** bzw. **Einfügen** (links oben im Bereich). Wählen Sie anschließend das Gewünschte aus dem Menü aus (Bedingungsgruppe oder Bedingung). (*siehe Abbildung unten*).



Arten von Bedingungen

Eine Bedingung kann einer von drei Gruppen angehören (*siehe Abbildung unten*):

- der Art "Komponente" (die Arten, die in der Dropdown-Liste mit *Component* beginnen*; siehe Abbildung unten*)
- der Art "Eigenschaft" (*Property Value*)
- einer Kombination von Komponenten- und Eigenschaftsarten (Arten mit *Property* und *Component* im Namen)

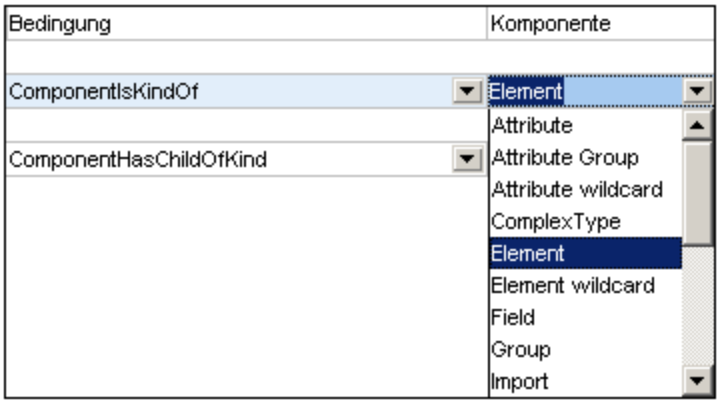


Die Art der Unterbedingung wird aus der Dropdown-Liste in der Spalte *Bedingung* der Bedingung ausgewählt (*Abbildung oben*). Jede der drei Gruppen von Bedingungen wird im Folgenden beschrieben.

Unterbedingungen der Art "Komponente"

Bei Unterbedingungen der Art "Komponente" (Arten, die mit *Component* beginnen) muss die Komponente anschließend in der Spalte *Komponente* (*siehe Abbildung unten*) definiert werden. Die Komponente wird aus der Dropdown-Liste im Komponentenfeld einer Unterbedingung ausgewählt. Da für Bedingungen der Art "Komponente" kein anderes Feld (Eigenschaft, Comparator Wert) definiert sein muss, sind alle anderen Felder ausgegraut.

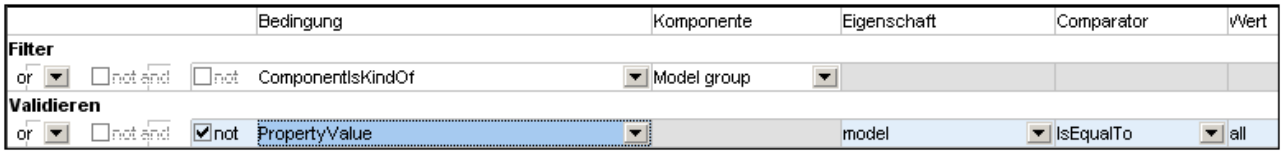
© 2016-2022 Altova GmbH



In der Abbildung oben definiert die Filterbedingung, dass die Regel sich auf Komponenten der Art *Schema* bezieht. Wenn in der Validierungsbedingung definiert ist, dass die Komponente ein Child der Art *Notation* haben muss, so lautet die komplette Regel dass: eine *Elementkomponente* ein Child der Art *Notation* haben muss. Wenn bei der Validierungsbedingung die Option `NOT` aktiviert ist, so würde die Regel lauten: Eine *Elementkomponente* darf kein Child der Art *Notation* haben.

Bedingungen der Art "Eigenschaft"

Die Bedingung der Art "Eigenschaft" ist *PropertyValue* (*siehe Abbildung unten*). Bei dieser Art von Bedingung wird der Charakter einer Eigenschaft definiert. Daher müssen in diesem Fall die Spalten *Eigenschaft* und *Comparator* und optional dazu auch die Spalte *Wert* ausgefüllt werden. In der Spalte *Komponente* ist kein Eintrag erforderlich, daher ist die Spalte ausgegraut. Eigenschaften, die in den Dropdown-Listen der Spalte *Eigenschaft* aufgelistet sind, enthalten nicht nur XML-Attribute (wie z.B. `default` und `maxOccurs`) sondern auch die logischen Eigenschaften von Komponenten (wie z.B. `derivedBy`).



In der Abbildung oben sehen Sie eine Regel, in der die Eigenschaft *Model* einen Wert hat, der gleich *All* ist, und der (über das Kontrollkästchen `NOT`) negiert ist. Zusammen mit dem Filter der Komponente *Model Group* besagt diese Regel ganz einfach, dass ein Schema keine `xsd:all` Elemente enthalten darf.

Anmerkung: Beachten Sie bitte die folgenden Punkte:

- Wenn Sie den Comparator `IsQNameEqualTo` verwenden, muss der entsprechende Wert in folgender Form geschrieben sein: `{URI}localName`. Der Wert könnte z.B.: `{http://www.w3.org/2001/XMLSchema}NOTATION` lauten.
- Die Eigenschaft `default` kann vorhanden und leer sein (`<element name default=""/>`) oder auch fehlen (`<element name/>`).

Bedingungen, in denen die Arten "Komponente" und "Art" kombiniert sind

Bedingungen, die aus einer Kombination der Arten "Komponente" und "Eigenschaft" bestehen sind:

- `ComponentHasChildOfKindWithPropertyValue`: Definiert die Art der Komponente eines Child-Elements und einen Eigenschaftswert dieses Child-Elements.
- `PropertyResolvesToComponentOfKind`: Eine Eigenschaft ist definiert, die zu einer Komponentenart aufgelöst wird. Die Spalten *Comparator* und *Value* sind leer.

Negieren einer Bedingung

Eine Bedingung wird durch Aktivieren des Kontrollkästchens *NOT* direkt links von der Bedingung (innere *Not*-Kontrollkästchen) negiert. Eine Bedingungsgruppe wird durch Aktivieren des *Not*-Kontrollkästchens links vom logischen Konnektor für Bedingungen in dieser Bedingungsgruppe negiert.

© 2016-2022 Altova GmbH

6.6 Kataloge in XMLSpy

XMLSpy unterstützt einen Teil der OASIS XML-Kataloge. Mit Hilfe von Katalogen kann XMLSpy gebräuchliche Schemas (sowie Stylesheets und andere Dateien) aus lokalen Benutzerordnern abrufen, wodurch die Verarbeitungsgeschwindigkeit erhöht wird, Benutzer auch offline arbeiten können (d.h. ohne mit einem Netzwerk verbunden zu sein) und Dokumente leichter übertragbar werden (da die URIs dann nur in den Katalogdateien geändert werden müssten).

Der Katalogmechanismus in XMLSpy funktioniert, wie in diesem Abschnitt beschrieben:

- Funktionsweise von Katalogen 463
- Katalogstruktur in XMLSpy 464
- Anpassen von Katalogen 466
- Umgebungsvariablen 467

Nähere Informationen zu Katalogen finden Sie in der XML-Katalogspezifikation.

6.6.1 Funktionsweise von Katalogen

Mit Hilfe von Katalogen können Umleitungen sowohl zu DTDs als auch XML-Schemas definiert werden. Das Prinzip des Mechanismus ist in beiden Fällen dasselbe, doch unterscheidet er sich in einigen weiter unten beschriebenen Details.

DTDs

Kataloge dienen normalerweise dazu, einen Aufruf von einer DTD an eine lokale URI umzuleiten. Dies geschieht in der Katalogdatei durch Mappen von Public und System Identifiern auf die gewünschte lokale URI. Wenn also die **DOCTYPE**-Deklaration in einer XML-Datei gelesen wird, findet ihr Public oder System Identifier über das Katalogdatei-Mapping die gewünschte lokale Ressource.

Für gebräuchliche Schemas ist der **PUBLIC** Identifier normalerweise vordefiniert, sodass für die URI in der Katalogdatei nur der **PUBLIC** Identifier auf den korrekten lokalen Identifier gemappt werden muss. Wenn das XML-Dokument geparst wird, wird der **PUBLIC** Identifier darin gelesen. Wenn dieser Identifier in einer Katalogdatei gefunden wird, wird die entsprechende URL in der Katalogdatei nachgeschlagen und das Schema wird von dort aus gelesen. Wenn also die folgende SVG-Datei in XMLSpy geöffnet wird:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg width="20" height="20" xml:space="preserve">
  <g style="fill:red; stroke:#000000">
     <rect x="0" y="0" width="15" height="15"/>
     <rect x="5" y="5" width="15" height="15"/>
  </g>
</svg>
```

Der Katalog wird nach dem **PUBLIC** Identifier dieser SVG-Datei durchsucht. Angenommen, die Katalogdatei enthält den folgenden Eintrag:

```
<catalog>
  ...
    <public publicId="-//W3C//DTD SVG 1.1//EN" uri="schemas/svg/svg11.dtd"/>
  ...
</catalog>
```

In diesem Fall gibt es einen Treffer für den **PUBLIC** Identifier, sodass der Lookup-Mechanismus für die SVG DTD an die URL **schemas/svg/svg11.dtd** umgeleitet wird; (dieser Pfad ist relativ zur Katalogdatei). Diese lokale Datei wird dann als DTD für die SVG-Datei verwendet. Wenn im Katalog kein passender Treffer für den **Public** Identifier gefunden wird, wird die URL im XML-Dokument verwendet (im SVG Beispiel oben ist dies die Internet URL **http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd**).

XML-Schemas

In XMLSpy können Sie eine **Umleitung zu einem XML-Schema** auch mit Hilfe von Katalogen durchführen. In der XML-Instanzdatei erfolgt die Referenz zum Schema im **xsi:schemaLocation** Attribut des Elements der obersten Ebene des XML-Dokuments. Beispiel:

```
xsi:schemaLocation="http://www.xmlspy.com/schemas/orgchart OrgChart.xsd"
```

Der Wert des **xsi:schemaLocation**-Attributs besteht aus zwei Teilen: einem Namespace-Teil (oben grün) und einem URI-Teil (markiert). Anhand des Namespace-Teils erfolgt im Katalog das Mapping auf die alternative Ressource. So erfolgt etwa im folgenden Katalogeintrag eine Umleitung der Schemareferenz oben auf ein Schema unter einem anderen Pfad.

```
<uri name="http://www.xmlspy.com/schemas/orgchart" uri="C:\MySchemas\OrgChart.xsd"/>
```

Normalerweise ist der URI-Teil des Attributwerts von **xsi:schemaLocation** ein Pfad zum aktuellen Schema. Wenn das Schema jedoch über einen Katalog referenziert wird, muss der URI-Teil nicht auf ein tatsächliches XML-Schema verweisen, muss aber vorhanden sein, damit das Attribut **xsi:schemaLocation** lexikalisch gültig ist. So würde z.B. der Wert `foo` als URI-Teil des Attributwerts genügen, um gültig zu sein.

6.6.2 Katalogstruktur in XMLSpy

XMLSpy lädt beim Start eine Datei namens **RootCatalog.xml**(*Struktur siehe unten*), die eine Liste von Katalogdateien enthält, die durchsucht werden. Sie können diese Datei bearbeiten und beliebig viele Katalogdateien definieren, die durchsucht werden sollen. Jede davon wird in einem **nextCatalog** referenziert. Diese Katalogdateien werden durchsucht und die URIs darin werden entsprechend ihren Mappings aufgelöst.

Codefragment von RootCatalog.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog xmlns="urn:oasis:names:tc:entity:xmlns:xml:catalog"
   xmlns:spy="http://www.altova.com/catalog_ext"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="urn:oasis:names:tc:entity:xmlns:xml:catalog Catalog.xsd">
  <nextCatalog catalog="%PersonalFolder%/Altova/%AppAndVersionName%/CustomCatalog.xml"/>
  <!-- Include all catalogs under common schemas folder on the first directory level -->
  <nextCatalog spy:recurseFrom="%CommonSchemasFolder%" catalog="catalog.xml"
```

 © 2016-2022 Altova GmbH

```
spy:depth="1"/>
  <nextCatalog spy:recurseFrom="%ApplicationWritableDataFolder%/pkgs/.cache"
catalog="remapping.xml" spy:depth="0"/>
  <nextCatalog catalog="CoreCatalog.xml"/>
</catalog>
```

Im obigen Codefragment wird ein benutzerdefinierter Katalog namens **CustomCatalog.xml** und eine Gruppe von Katalogen, die gebräuchliche Schemas (wie z.B. W3C XML-Schemas udn das SVG-Schema) referenzieren, referenziert.

- **CustomCatalog.xml** befindet sich in Ihrem persönlichen Ordner (der über die Variable **%PersonalFolder%**) gefunden wird. erstellen. Diese Datei ist eine Skelettdatei, in der Sie Ihre eigenen Mappings erstellen können. Sie können für jedes gewünschte Schema, das nicht in den Katalogdateien im Altova Ordner "Common Schemas" enthalten ist, Mappings zu **CustomCatalog.xml** hinzufügen. Verwenden Sie dazu die unterstützten Elemente des OASIS-Katalogmechanismus (*siehe nächster Abschnitt*).
- Der durch die Variable **%CommonSchemasFolder%** definierte Ordner "Common Schemas" enthält eine Reihe gebräuchlicher Schemas. Innerhalb dieser einzelnen Schema-Ordner befindet sich eine **catalog.xml**-Datei, die Public und/oder System Identifier auf URIs mappt, die auf lokal gespeicherte Kopien des jeweiligen Schemas verweisen.
- Schemas, die im Zusammenhang mit XBRL-Dateien und verschiedenen XBRL-Taxonomien stehen, sind sehr umfangreich und werden bei Bedarf mit Hilfe des Altova Taxonomie-Managers lokal installiert. Die einzelnen Schemas und Taxonomien werden im Katalog **remapping.xml**, der sich im Unterordner **pkgs/.cache** des Ordners Program Data befindet (dessen Pfad durch die Variable **%ApplicationWritableDatFolder%** angegeben wird), gemappt. **Bitte bearbeiten Sie diese Datei nicht;** der kleinste Fehler hätte schwerwiegende Folgen für große Gruppen von Referenzen.
- **CoreCatalog.xml** befindet sich im XMLSpy-Applikationsordner und dient zum Auffinden von Schemas und Stylesheets, die von XMLSpy-spezifischen Prozessen wie z.B. StyleVision Power Stylesheets, anhand derer die Altova-Authentic-Ansicht von XML-Dokumenten generiert wird.

Pfadvariablen
Die in **RootCatalog.xml** verwendeten Variablen (*Codefragment oben*) haben die folgenden Werte:

%PersonalFolder%	Der persönliche Ordner des aktuellen Benutzers, z.B. C:\Benutzer\<Name>\Dokumente
%CommonSchemasFolder%	C:\ProgramData\Altova\Common2023\Schemas
%ApplicationWritableDataFolder%	C:\ProgramData\Altova

Speicherpfad von Katalogdateien und Schemas
Beachten Sie die Pfade der verschiedenen Katalogdateien.

- Die Dateien **RootCatalog.xml** und **CoreCatalog.xml** befinden sich im XMLSpy Applikationsordner.
- **CustomCatalog.xml** befindet sich in Ihrem Ordner **EigeneDokumente\Altova\XMLSpy**.
- Die **catalog.xml**-Dateien befinden sich jeweils in einem eigenen Schemaordner, wobei sich diese Schemaordner innerhalb des Ordners "Common Schemas" befinden:

6.6.3 Anpassen von Katalogen

Wenn Sie Einträge für **CustomCatalog.xml** (oder jede andere Katalogdatei, die von XMLSpy gelesen werden soll) erstellen, verwenden Sie nur die folgenden Elemente der OASIS-Katalogspezifikation. Jedes der unten angeführten Elemente wird mit einer Erläuterung der Attributwerte aufgelistet. Eine ausführlichere Beschreibung finden Sie in der XML Catalogs Specification. Beachten Sie, dass jedes Element das Attribut **xml:base**, mit dem die Basis-URI dieses Elements definiert wird, erhalten kann.

- <public publicId="PublicID of Resource" uri="URL of local file"/>
- <system systemId="SystemID of Resource" uri="URL of local file"/>
- <uri name="filename" uri="URL of file identified by filename"/>
- <rewriteURI uriStartString="StartString of URI to rewrite" rewritePrefix="String to replace StartString"/>
- <rewriteSystem systemIdStartString="StartString of SystemID" rewritePrefix="Replacement string to locate resource locally"/>

Beachten Sie die folgenden Punkte:

- In Fällen, in denen es keinen Public Identifier gibt, kann der System Identifier einer URL wie z.B. bei den meisten Stylesheets über das **system** Element direkt auf eine URL gemappt werden.
- Eine URI kann über das uri Element auf eine andere URI gemappt werden.
- Mit Hilfe der Elemente rewriteURI und rewriteSystem kann der Anfangsteil einer URI bzw. eines System Identifiers neu geschrieben werden. Dadurch kann der Anfang eines Dateipfads ersetzt werden, sodass ein anderes Verzeichnis als Ziel gewählt werden kann. Nähere Informationen zu diesen Elementen finden Sie in der XML Catalogs Specification.

Ab Release 2014 entspricht XMLSpy weitgehend der XML Catalogs Specification (OASIS Standard V1.1, 7 Oktober 2005). In dieser Spezifikation wird streng zwischen externen Identifier Look-ups (jenen mit einer öffentlichen ID oder einer System-ID) und URI Look-ups (URIs, die keine öffentlichen IDS oder System-IDs sind) getrennt. Namespace URIs müssen daher einfach als URIs - und nicht Public IDs oder System-IDs - behandelt werden und folglich als URI Look-ups anstelle von externen Identifier Look-ups verwendet werden. In XMLSpy Versionen vor Version 2014 wurden Schema Namespace URIs über **<public>** Mappings übersetzt. Ab Version 2014 müssen **<uri>** Mappings verwendet werden.

Vor v2014: <public publicID="http://www.MyMapping.com/ref"
uri="file:///C:/MyDocs/Catalog/test.xsd"/>
Ab V-2014: <uri name="http://www.MyMapping.com/ref"
uri="file:///C:/MyDocs/Catalog/test.xsd"/>

Wie findet XMLSpy ein referenziertes Schema

Ein Schema wird in einem XML-Dokument über das Attribut **xsi:schemaLocation** (*siehe unten*) referenziert. Der Wert des **xsi:schemaLocation**-Attributs besteht aus zwei Teilen: einem Namespace-Teil (grün) und einem URI-Teil (markiert).

```
xsi:schemaLocation="http://www.xmlspy.com/schemas/orgchart OrgChart.xsd"
```

Im Folgenden wird Schritt für Schritt beschrieben, wie XMLSpy vorgeht, um ein referenziertes Schema zu finden: Das Schema wird beim ersten erfolgreichen Schritt geladen.

1. Durchsuchen des Katalogs nach dem URI-Teil des **xsi:schemaLocation**-Werts. Wenn ein Mapping

© 2016-2022 Altova GmbH

gefunden wird - darunter auch in **rewriteURI** Mappings wird die erzeugte URI verwendet, um das Schema zu laden.
2. Durchsuchen des Katalogs nach dem Namespace-Teil des **xsi:schemaLocation**-Werts. Wenn ein Mapping gefunden wird - darunter auch in **rewriteURI** Mappings wird die erzeugte URI verwendet, um das Schema zu laden.
3. Der URI-Teil des **xsi:schemaLocation**-Werts wird zum Laden des Schemas verwendet.

Dateierweiterungen und intelligente schemagemäße Bearbeitung

Mittels Katalogdateien können Sie auch festlegen, dass auf Dokumente mit einer bestimmten Dateierweiterung die intelligenten Bearbeitungsfunktionen von XMLSpy gemäß den Regeln des von Ihnen angegebenen Schemas angewendet werden. Wenn Sie z.B. eine benutzerdefinierte Dateierweiterung `.myhtml` für (HTML)-Dateien erstellen, die gemäß der HTML-DTD gültig sein sollen, können Sie die intelligente Editierung für Dateien mit dieser Erweiterung aktivieren, indem Sie das folgende Textelement als Child-Element des **<catalog>** Elements zu **CustomCatalog.xml** hinzufügen.

```
</catalog>
  ...
    <spy:fileExtHelper ext="myhtml" uri="schemas/xhtml/xhtml1-transitional.dtd"/>
  ...
</catalog>
```

Damit werden die intelligenten Bearbeitungsfunktionen (Autokomplettierung, Eingabehilfen usw.) von **.myhtml**-Dateien in XMLSpy gemäß der XHTML 1.0 Transitional DTD aktiviert. Sehen Sie in der Datei **catalog.xml** im Ordner **%AltovaCommonSchemasFolder%\Schemas\xhtml**, welche ähnliche Einträge enthält, nach.

XML-Schema-Spezifikationen

Die XML-Schemaspezifikationsinformationen sind in XMLSpy integriert und die Gültigkeit von XML-Schema-(`.xsd`)-Dokumenten wird anhand dieser internen Informationen überprüft. Daher sollte in einem XML-Schema-Dokument kein Verweis auf ein Schema, das die XML-Schema-Spezifikation definiert, vorgenommen werden.

Die Datei **catalog.xml** im Ordner `%AltovaCommonSchemasFolder%\Schemas\schema` enthält Referenzen auf DTDs, die ältere XML-Schema-Spezifikationen implementieren. Sie sollten Ihre XML-Schema-Dokumente nicht anhand dieser Schemas validieren. Zweck dieser beiden DTDs ist es einzig und allein, für die Eingabehilfen von XMLSpy zu Bearbeitungszwecken Informationen bereitzustellen, falls Sie Dateien gemäß diesen älteren Empfehlungen erstellen wollen.

6.6.4 Umgebungsvariablen

Shell-Umgebungsvariablen können im `nextCatalog` Element verwendet werden, um den Pfad zu Systemordnern zu definieren (*siehe RootCatalog.xml-Liste oben*). Es werden die folgenden Shell-Umgebungsvariablen unterstützt:

%PersonalFolder%	Vollständiger Pfad zum persönlichen Ordner des aktuellen Benutzers, z.B. `C:\Benutzer\<Name>\Dokumente`
%CommonSchemasFol	C:\ProgramData\Altova\Common2023\Schemas

der%	
%ApplicationWritableDataFolder%	C:\ProgramData\Altova
%AltovaCommonFolder%	C:\Programme\Altova\Common2023
%DesktopFolder%	Vollständiger Pfad des Desktop-Ordners des aktuellen Benutzers.
%ProgramMenuFolder%	Vollständiger Pfad zum Ordner 'Programme" des aktuellen Benutzers.
%StartMenuFolder%	Vollständiger Pfad zum Startmenüordner des aktuellen Benutzers.
%StartUpFolder%	Vollständiger Pfad zum Startordner des aktuellen Benutzers.
%TemplateFolder%	Vollständiger Pfad des Template-Ordners des aktuellen Benutzers.
%AdminToolsFolder%	Vollständiger Pfad zum Dateisystemverzeichnis, in dem Verwaltungstools des aktuellen Benutzers gespeichert sind.
%AppDataFolder%	Vollständiger Pfad zum Ordner "Anwendungsdaten" des aktuellen Benutzers.
%CommonAppDataFolder%	Vollständiger Pfad zum Dateisystem, das die Applikationsdaten aller Benutzer enthält.
%FavoritesFolder%	Vollständiger Pfad zum Ordner 'Favoriten' des aktuellen Benutzers.
%PersonalFolder%	Vollständiger Pfad zum Ordner "Personal" des aktuellen Benutzers.
%SendToFolder%	Vollständiger Pfad zum SendTo-Ordner des aktuellen Benutzers.
%FontsFolder%	Vollständiger Pfad zum Ordner "Systemschriftarten".
%ProgramFilesFolder%	Vollständiger Pfad zum Ordner 'Programme" des aktuellen Benutzers.
%CommonFilesFolder%	Vollständiger Pfad zum Ordner "Gemeinsame Dateien" des aktuellen Benutzers.
%WindowsFolder%	Vollständiger Pfad zum Ordner "Windows" des aktuellen Benutzers.
%SystemFolder%	Vollständiger Pfad zum Ordner "System" des aktuellen Benutzers.
%LocalAppDataFolder%	Vollständiger Pfad zum Dateisystemverzeichnis, das als Datenspeicher für lokale (nicht-Roaming) Applikationen dient.
%MyPicturesFolder%	Vollständiger Pfad zum Ordner "Meine Bilder".

© 2016-2022 Altova GmbH

6.7 Arbeiten mit SchemaAgent

XMLSpy kann für die Verwendung mit Altova SchemaAgent konfiguriert werden.

SchemaAgent Technologie

SchemaAgent gestattet Benutzern, Beziehungen zwischen mehreren Schemas zu erstellen und zu bearbeiten. SchemaAgent besteht aus:

- Einem SchemaAgent Server, der Informationen über die Beziehungen zwischen Schemas in einem oder mehreren von Ihnen definierten Suchpfaden (Ordnern auf dem Netzwerk) enthält.
- Einem SchemaAgent Client, dem Altova Produkt SchemaAgent 2023, der anhand der Schema-Informationen vom SchemaAgent, mit dem er verbunden ist, (i) Beziehungen zwischen diesen Schemas erstellt und (ii) diese Schemas verwaltet (durch Umbenennen, Verschieben, Löschen von Schemas usw.);

Es stehen zwei Arten von SchemaAgent Servern zur Verfügung:

- Altova SchemaAgent Server 2023, der auf einem Netzwerk installiert werden kann, und auf den Sie über das Netzwerk zugreifen können und
- Altova SchemaAgent 2023. Dies ist das SchemaAgent Client-Produkt. Es enthält eine Leichtversion des Servers, den so genannten LocalServer, der nur auf demselben Rechner verwendet werden kann, auf dem auch SchemaAgent installiert ist.

XMLSpy verwendet die SchemaAgent-Technologie zum direkten Editieren von Schemas in der Schema-Ansicht, wobei das Programm Informationen über andere Schemas verwendet, die es von einem SchemaAgent Server erhält. In dieser Konfiguration ist XMLSpy mit einem SchemaAgent Server verbunden und sendet über SchemaAgent Client Requests an SchemaAgent Server. Wenn XMLSpy für die Arbeit mit SchemaAgent konfiguriert wurde, enthalten die Eingabehilfen in der Schema-Ansicht nicht nur Komponenten aus dem derzeit in der Schema-Ansicht gerade aktiven Schema sondern auch Komponenten aus anderen Schemas in den Suchpfaden des SchemaAgent Servers, mit dem das Programm gerade verbunden ist. Auf diese Art haben Sie direkten Zugriff auf diese Komponenten. Sie können das Content Model einer zu einem anderen Schema gehörenden Komponente in der Schema-Ansicht anzeigen und sie mit oder ohne Änderungen wiederverwenden. Außerdem können Sie Beziehungen zwischen Schemas erstellen und dadurch komplexe Schemas direkt in XMLSpy in Module aufteilen und verwalten.

Installation von SchemaAgent und SchemaAgent Server

Nähere Informationen zu SchemaAgent und SchemaAgent Server und zum Konfigurieren von Suchpfaden auf Servern finden Sie im Benutzerhandbuch zu SchemaAgent.

Einrichten von XMLSpy als SchemaAgent Client

Um XMLSpy für die Arbeit als SchemaAgent Client zu konfigurieren, gehen Sie folgendermaßen vor:

- Downloaden Sie SchemaAgent von der Altova Website und installieren Sie ihn auf Ihrem Netzwerk. Sie können nun über den LocalServer von SchemaAgent auf Schemas zugreifen. Informationen zum Konfigurieren von Suchpfaden in LocalServer finden Sie im SchemaAgent-Benutzerhandbuch. **Beachten Sie**, dass Sie für SchemaAgent eine gültige Lizenz benötigen, die Sie nach Ablauf des

© 2016-2022 Altova GmbH

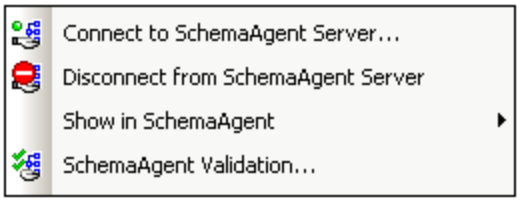
Testzeitraums erwerben müssen, um das Produkt weiterverwenden zu können. Beachten Sie, dass das Altova Altova MissionKit Enterprise Edition-Paket das Produkt SchemaAgent sowie einen Lizenzschlüssel dazu enthält. (Die SchemaAgent Server-Applikation ist jedoch nicht in den Altova MissionKit-Paketen enthalten.)

- Zusätzlich sollten Sie bei Bedarf den netzwerkbasierten SchemaAgent Server von der Altova Website herunterladen und installieren.
- Definieren Sie den/die Suchpfad(e) für den SchemaAgent Server (Konfigurieren von SchemaAgent Server). Eine ausführliche Erläuterung dazu finden Sie in der Dokumentation zu SchemaAgent. (Ein Suchpfad ist ein Pfad zu dem Ordner, der die XML-Schemas enthält, die miteinander in Zusammenhang stehen und gemappt werden).
- Stellen Sie eine Verbindung zwischen XMLSpy und dem SchemaAgent server.

Wichtig: Alle SchemaAgent-Produkte und alle mit SchemaAgent in Zusammenhang stehenden Produkte von Altova (einschließlich XMLSpy) beginnend mit Version 2005 Release 3 sind **nicht** mit früheren Versionen von SchemaAgent oder mit SchemaAgent in Zusammenhang stehenden Produkten kompatibel.

SchemaAgent-Befehle in XMLSpy

Die SchemaAgent-Funktion in XMLSpy steht nur in der Schema-Ansicht zur Verfügung und wird über die Menübefehle im Schema-Design-Menü *(siehe Abbildung unten),* die Symbolleiste und die Eingabehilfen in der Schema-Ansicht aufgerufen.



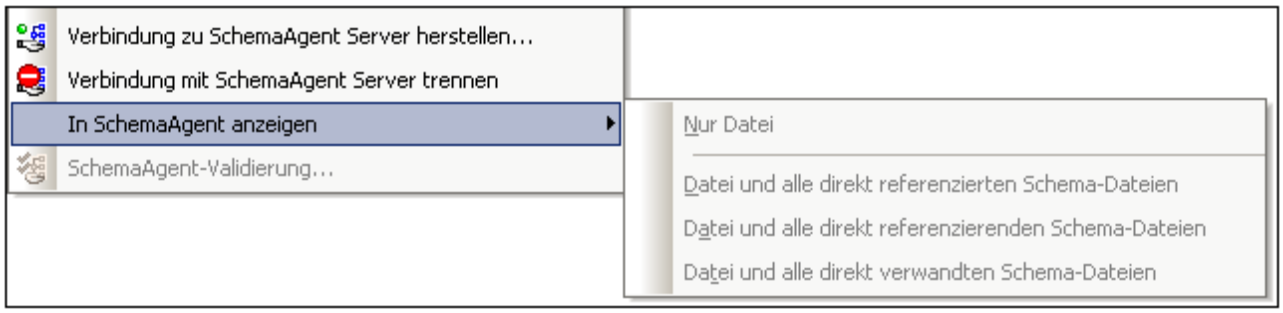
Über die Menübefehle stehen Ihnen allgemeine Verwaltungsfunktionen zur Verfügung. Die Eingabehilfen (und die Standardfunktion der Benutzeroberfläche wie z.B. Drag & Drop) dienen zum Bearbeiten von Schemas.

In diesem Abschnitt wird beschrieben, wie man die in der Schema-Ansicht zur Verfügung stehenden SchemaAgent-Funktionen verwendet.

6.7.1 Herstellen einer Verbindung mit SchemaAgent Server

Anmerkung: Um eine Verbindung herstellen zu können, muss SchemaAgent Client installiert sein.

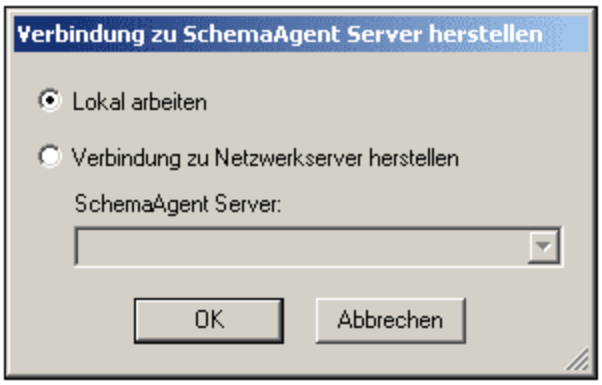
Bevor Sie eine Verbindung zu SchemaAgent Server hergestellt haben, ist im Menü **Schema-Design** nur der Befehl **Verbindung zu SchemaAgent Server herstellen,** aktiv. Alle anderen SchemaAgent-Befehle im Menü **Schema-Design** sind deaktiviert (*siehe Abbildung unten*). Die anderen Menüeinträge werden aktiv, sobald eine Verbindung zu SchemaAgent Server besteht.



Schritte beim Herstellen einer Verbindung

So stellen Sie eine Verbindung zu einem SchemaAgent Server her:

1. Klicken Sie auf des Symbol "Verbindung zu SchemaAgent Server herstellen"  **(Schema-Design | Verbindung zu SchemaAgent Server herstellen).** Daraufhin wird das gleichnamige Dialogfeld angezeigt (*siehe Screenshot unten*).



2. Sie können entweder mit dem lokalen Server arbeiten (dem SchemaAgent, Server, der mit Altova SchemaAgent 2023 mitgeliefert wird). oder mit einem Netzwerk-Server (dem Altova SchemaAgent Server, der kostenlos erhältlich ist). Wenn Sie die Option "Lokal arbeiten" auswählen, wird der lokale Server von SchemaAgent gestartet, wenn Sie auf OK klicken und es wird eine Verbindung mit diesem Server hergestellt. Wenn Sie die Option "Verbindung zu Netzwerk-Server herstellen" wählen, muss der ausgewählte SchemaAgent Server ausgeführt werden, damit eine Verbindung hergestellt werden kann.

Bitte beachten Sie zu Servern, die unter Windows XP SP2 laufen
Wenn der Name des SchemaAgent Servers im Dialogfeld "Verbindung zu SchemaAgent Server herstellen" aufgelistet ist, Sie jedoch keine Verbindung herstellen können, liegt dies möglicherweise daran, dass Ihr Server nicht an der Namensauflösung in Ihrem Netzwerk teilnimmt. Die Namensauflösung wird durch die Standardeinstellungen der Windows XP SP2 Firewall verhindert.

Um die Verbindung zu einem derartigen Server herzustellen, haben Sie folgende Möglichkeiten:

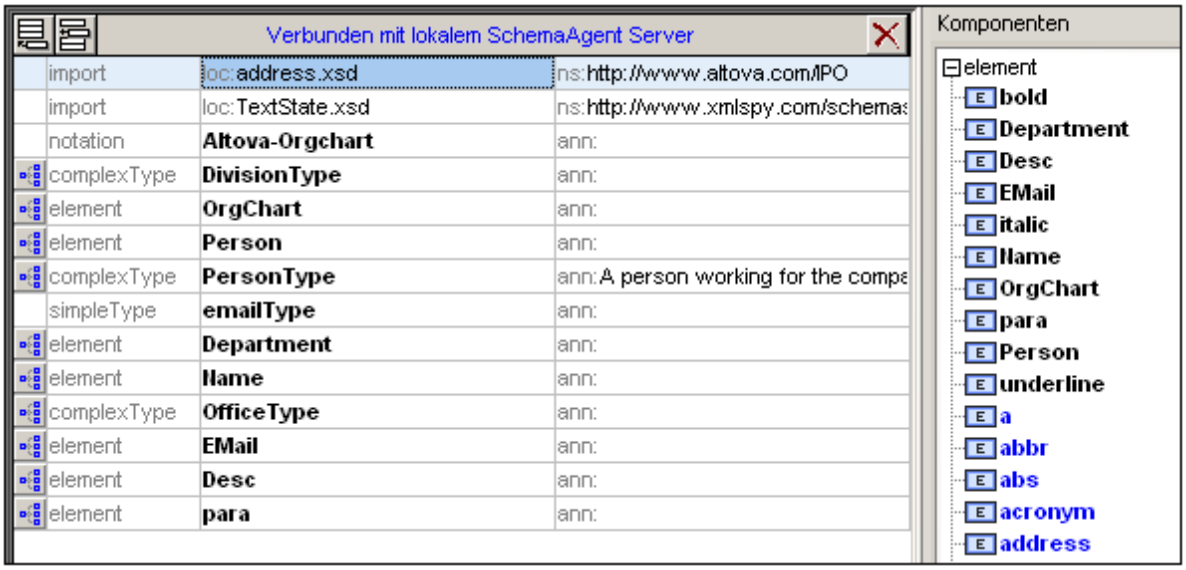
© 2016-2022 Altova GmbH

- Ändern Sie die Servereinstellungen, um die Namensauflösung zuzulassen oder
- Geben Sie im Bearbeitungsfeld des Dialogfelds zur Verbindungsherstellung die IP-Adresse des Servers ein.

Sie müssen diese Einstellungen nur einmal vornehmen, da SchemaAgent Client den Connection String der letzten funktionierenden Verbindung speichert.

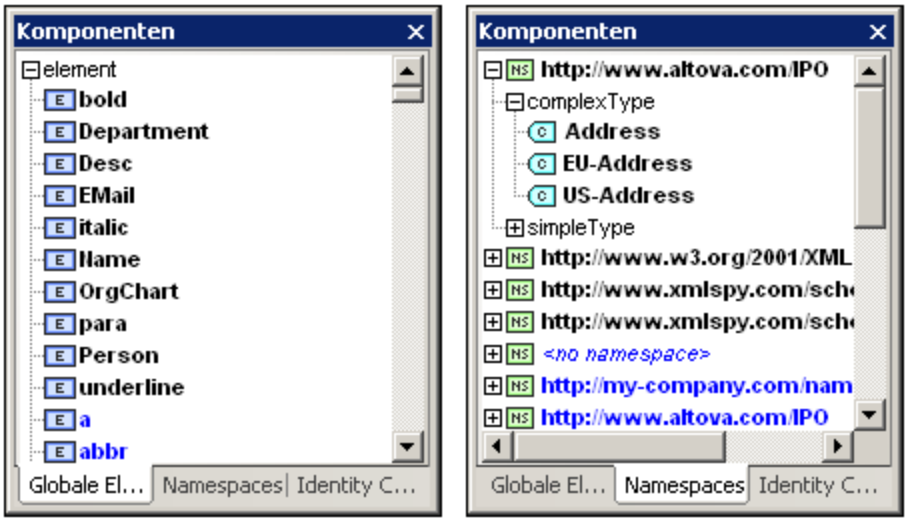
Schema-Ansicht nach Herstellen der Verbindung mit SchemaAgent Server

Nachdem Sie eine Verbindung zu einem SchemaAgent Server hergestellt haben, sieht die Schema-Ansicht in etwa folgendermaßen aus:



Beachten Sie bitte Folgendes:

- In der Schema-Ansicht werden nun in der Ansicht der globalen Elemente der Text "Verbunden mit SchemaAgent Server ..." mit Angabe des Servers, mit dem Sie eine Verbindung hergestellt haben.
- Sie haben nun Zugriff auf alle Schemas und Schema-Konstrukte, die unter dem Server-Suchpfad zur Verfügung stehen. SchemaAgent Schema-Konstrukte: Globale Elemente, complexTypes, simpleTypes usw. werden unterhalb der Konstrukte des aktiven Schemas **(schwarz, fett) blau und fett** angezeigt.

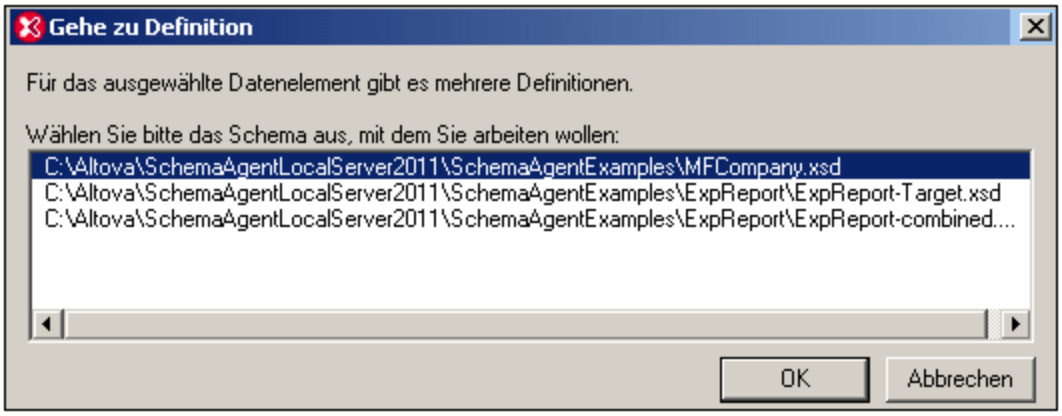


Schema-Konstrukte können in der Komponenten-Eingabehilfe auf den entsprechenden Registern nach Typ (Globale Elemente), nach Namespace oder nach Identity Constraint angezeigt werden.

6.7.2 Öffnen von Schemas aus dem Suchpfad

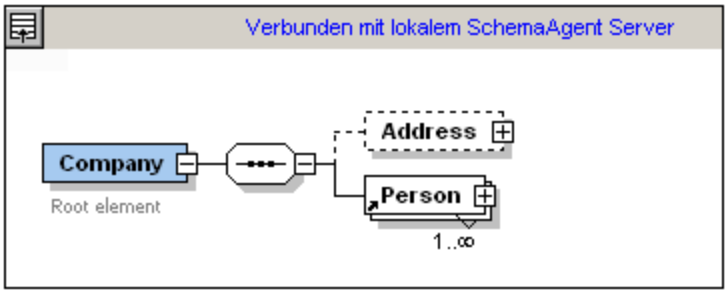
In diesem Beispiel wird gezeigt, wie man ein Schema öffnet, das unter einem in SchemaAgent Server definierten Suchpfad gespeichert ist. Als aktives Schema wird die Datei `DB2schema.xsd` aus dem Ordner `..\Tutorial` verwendet. Das Register `Globale Elemente` der Komponenten-Eingabehilfe ist das aktive Register.

1. Scrollen Sie in der Komponenten-Eingabehilfe hinunter zum blauen Eintrag `Company` in der Komponente "Element" und doppelklicken Sie darauf. Daraufhin wird das Dialogfeld "Gehe zu Definition" geöffnet.



2. Klicken Sie auf den Eintrag `Addresslast.xsd` und bestätigen Sie mit OK. Das Schema `addresslast.xsd` wird geöffnet und das Content Model des Elements `Company` wird angezeigt.

 © 2016-2022 Altova GmbH



Anmerkung: Wenn Sie auf ein SchemaAgent Schema-Konstrukt, wie z.B. ein Element, einen complexType oder einen simpleType doppelklicken, wird das dazugehörige Schema (sowie alle anderen inkludierten Schemas) in XMLSpy geöffnet.

6.7.3 Verwenden von IIR-Beziehungen

XML-Schema bietet drei grundlegende Methoden, um Schemas in Module aufzugliedern: Import, Include und Redefine (IIR). Bei jeder Methode gelten andere Namespace-Anforderungen. Diese werden jedoch automatisch beim Erstellen von IIRs von SchemaAgent Client und XMLSpy überprüft.

Import-, Include- und Redefine-Anweisungen (IIR)

Es gibt unterschiedliche Arten, um ein Schema-Konstrukt "einzufügen":

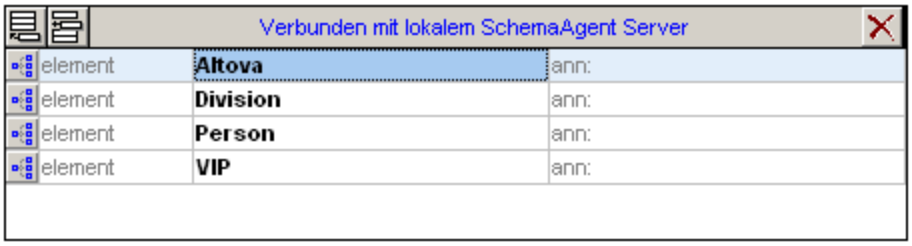
- Sie können globale Elemente direkt aus der Komponenten-Eingabehilfe (in der Schema-Ansicht) in das Content Model einer Schemakomponente ziehen
- Komponenten, wie z.B. complexTypes und simpleTypes können aus einem Listenfeld ausgewählt werden, das automatisch geöffnet wird, wenn Sie ein neues Element/Attribut usw. definieren.
- Komponenten wie z.B. complexTypes können bei der Erstellung/Aktualisierung dieser Konstruktart in der Details-Eingabehilfe ausgewählt werden.

Integrieren von Schema-Komponenten

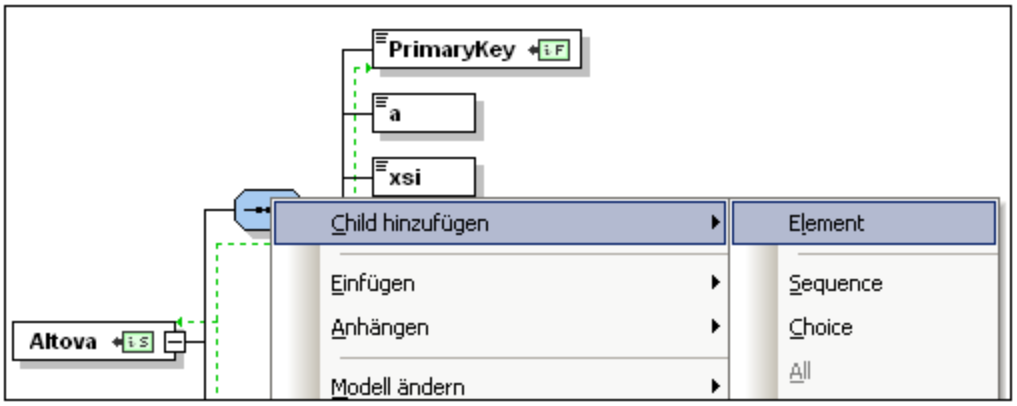
In diesem Beispiel wird die Datei `DB2schema.xsd` aus dem Ordner `..\Tutorial` als aktives Schema verwendet; das Register `Globale Elemente` der Komponenten-Eingabehilfe ist aktiv.

So verwenden Sie Schema-Konstrukte aus SchemaAgent Server Schemas:

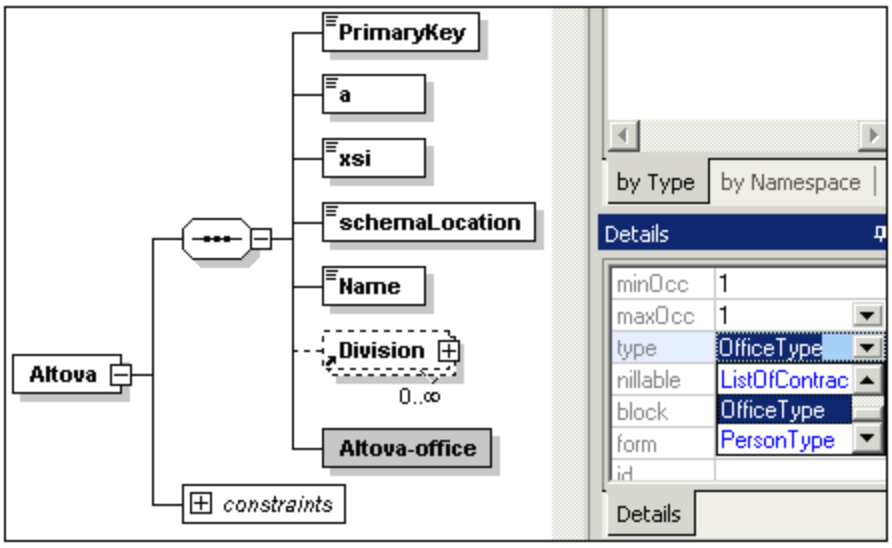
1. Stellen Sie sicher, dass eine Verbindung mit einem SchemaAgent Server hergestellt ist (siehe Herstellen einer Verbindung mit SchemaAgent Server 471).
2. Öffnen Sie die Datei `DB2Schema.xsd` und benennen Sie sie für dieses Beispiel in `Altova-office` um.



3. Klicken Sie in der Schema-Übersicht auf das Element `Altova`, um dessen Content Model aufzurufen.
4. Rechtsklicken Sie auf den `Altova` sequence-Kompositor und wählen Sie die Menüoption **Child hinzufügen | Element**. Beachten Sie, dass nun automatisch ein Listenfeld mit allen globalen Elementen aus dem Serverpfad geöffnet wird. Wenn Sie eines auswählen, wird dieses Element integriert.

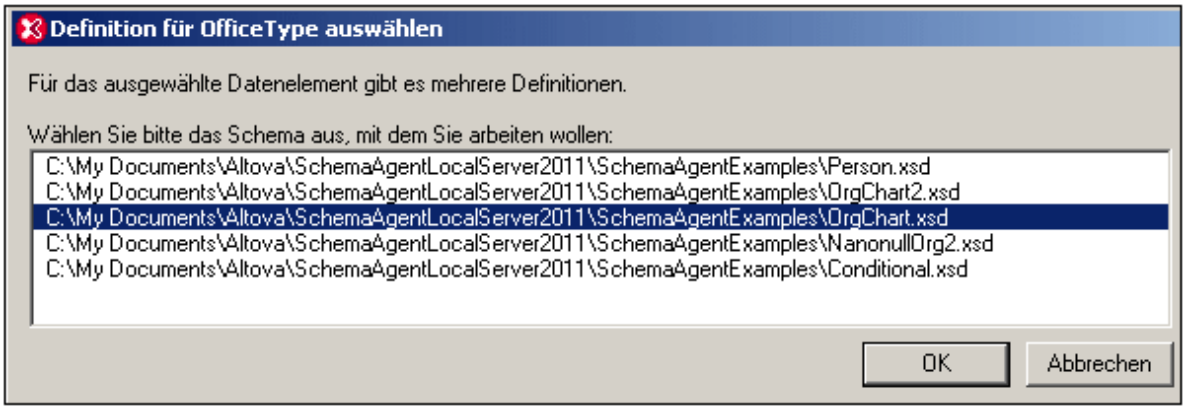


5. Geben Sie `Altova-office` als Namen für dieses neue Element ein und drücken Sie die Eingabetaste.
6. Klicken Sie in der Details-Eingabehilfe auf das Listenfeld `type` und wählen Sie den Eintrag `OfficeType` aus.

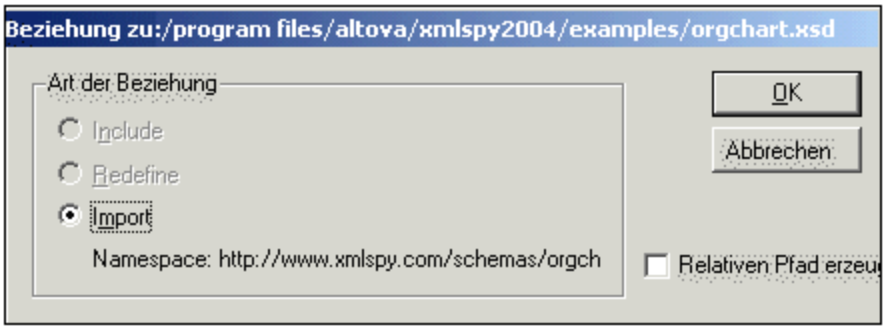


Daraufhin wird das Dialogfeld "Definition für OfficeType auswählen" geöffnet.

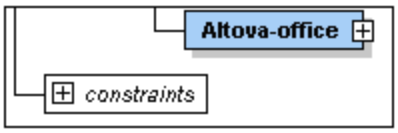
 © 2016-2022 Altova GmbH



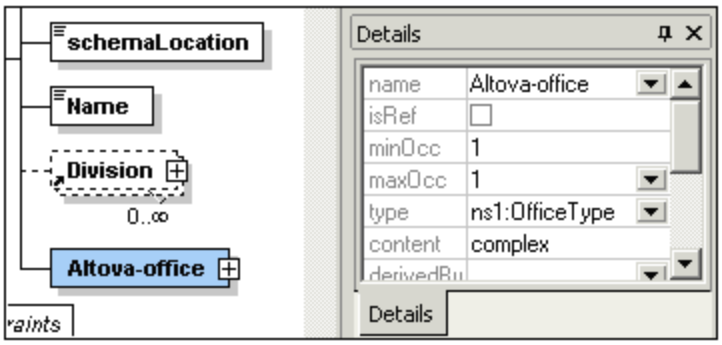
7. Wählen Sie `Orgchart.xsd` und klicken Sie auf OK.



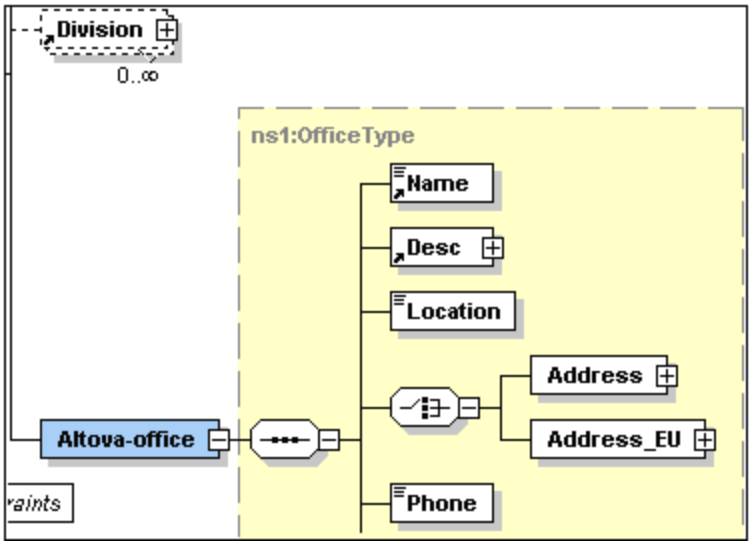
8. Klicken Sie im nächsten Dialogfeld auf OK. Der Befehl "Import" wurde automatisch ausgewählt. Das Element `Altova-office` weist nun ein Plus-Zeichen zum Erweitern auf.



Bitte beachten Sie, dass sich der Eintrag `type` entry in der Details-Eingabehilfe geändert hat; er wird nun als `ns1:OfficeType` angezeigt, da die Schemadatei `Orgchart.xsd` importiert wurde und die beiden Schemas unterschiedliche Target Namespaces haben müssen. Außerdem wurde zum Schema ein Import-Befehl hinzugefügt.



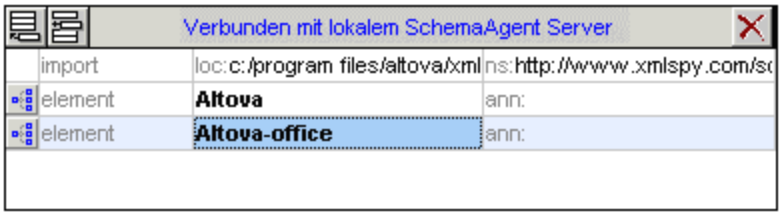
9. Klicken Sie auf das Erweiterungssymbol, um das Content Model von `OfficeType` anzuzeigen.



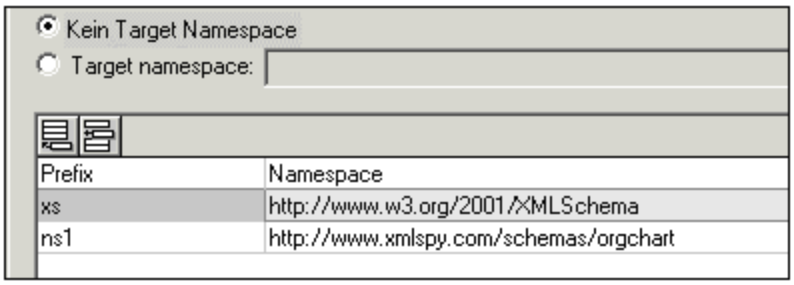
10. Drücken Sie F8, um das Schema zu validieren. Daraufhin sollte die Meldung, dass das Schema gültig ist, erscheinen.

Aufräumen des Schemas:

1. Löschen Sie das Element `Division` im Content Model.
2. Klicken Sie auf das Symbol zum Anzeigen der globalen Elemente, um in die Schema-Übersicht zu wechseln.
3. Löschen Sie die folgenden globalen Elemente: `Division`, `Person` und `VIP`.

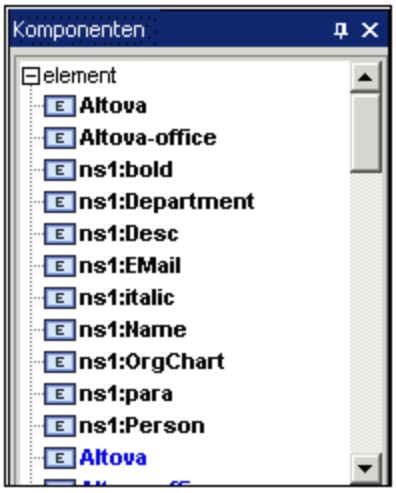


4. Wählen Sie die Menüoption **Schema-Design | Schema-Einstellungen**, um zu sehen, inwiefern sich die Namespace-Einstellungen geändert haben.



Das `ns1`-Präfix wurde automatisch zum Namespace `www.xmlspy.com/schemas/orgchart` hinzugefügt. Die Komponente- und die Details-Eingabehilfe zeigen alle importierten Konstrukte mit dem ns1: namespace Präfix an.

 © 2016-2022 Altova GmbH



Anmerkung:

- Änderungen, die in XMLSpy unter Verwendung von SchemaAgent vorgenommen wurden, werden automatisch in allen anderen Schemas im SchemaAgent Serverpfad, der das geänderte Schema referenziert, übernommen.
- Unter Umständen werden Konstrukte wie Elemente, simpleTypes usw. in den Eingabenhilfen doppelt angezeigt (in Schwarz und Blau), wenn sich das Schema, mit dem Sie gerade arbeiten, auch unter dem SchemaAgent Serverpfad befindet.

6.7.4 Anzeigen von Schemas in SchemaAgent

Um mit dem aktiven Schema und den damit verknüpften Schemas in SchemaAgent zu arbeiten, klicken Sie auf die Menüoption **Schema-Design | In SchemaAgent anzeigen | *Schema und alle direkt verwandten*** *(siehe Abbildung unten).*
Sie haben die Option, entweder nur das aktive Schema in SchemaAgent zu öffnen (Befehl **Nur Datei**) oder das aktive Schema zusammen mit entweder (i) allen direkt referenzierten Schemas oder (ii) allen direkt referenzierenden Schema oder (iii) allen direkt verwandten Schemas zu öffnen.

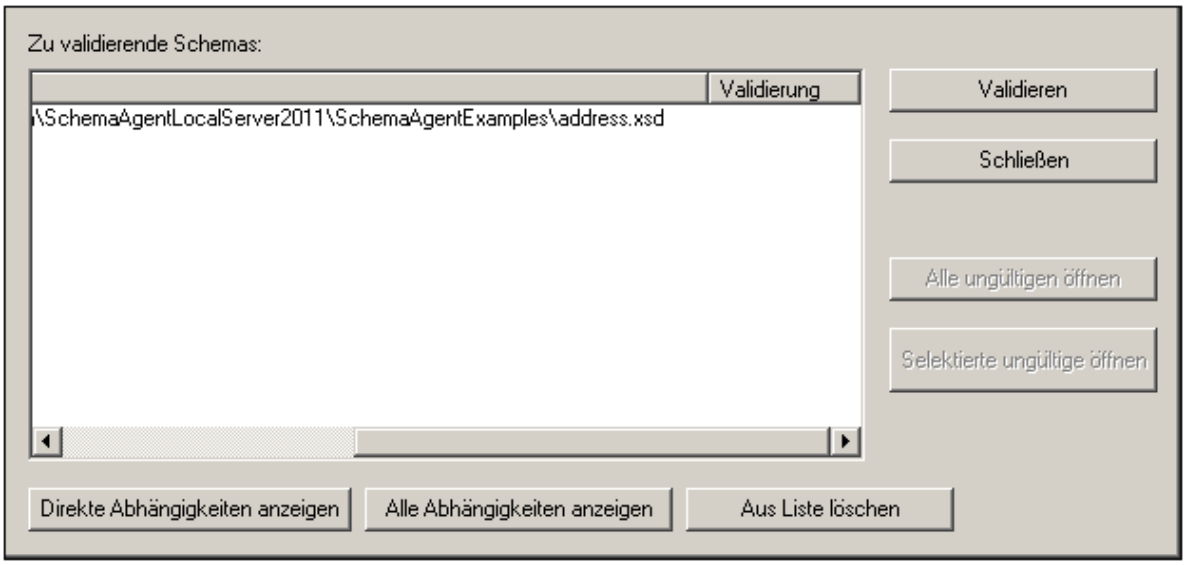
6.7.5 SchemAgent-Validierung

XMLSpy gestattet Ihnen in Verbindung mit SchemaAgent nicht nur das aktive Schema zu validieren, sondern auch Schemas, die mit dem aktiven Schema in Zusammenhang stehen. Wir bezeichnen dies als SchemaAgent-Validierung. SchemaAgent unterscheidet bei der erweiterten Validierung zwischen: (i) direkt abhängigen Schemas (direkt referenzierten und direkt referenzierenden Schemas) und (ii) allen abhängigen Schemas (zusätzlich zu den direkten Abhängigkeiten gehören dazu indirekte Abhängigkeiten, d.h. die Gruppe von Schemas, die über ein Zwischenschema mit einem anderen Schema in Zusammenhang stehen.

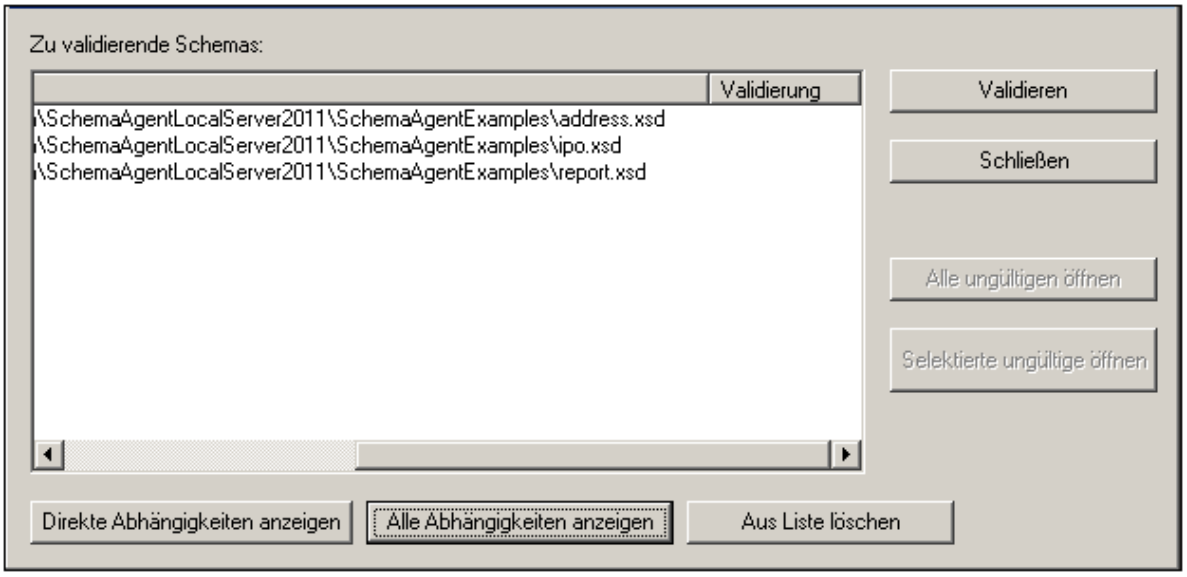
Die Funktionsweise der SchemaAgent-Validierung wird im Folgenden anhand eines Beispiels beschrieben. In diesem Beispiel wird als aktive Schemadatei in der Schema-Ansicht von XMLSpy `address.xsd` verwendet. Damit der Befehl **SchemaAgent-Validierung** aktiviert ist, vergewissern Sie sich, dass die Suchpfade in

SchemaAgent Server die aktive Datei und einige davon abhängige Dateien enthalten. Gehen Sie anschließend folgendermaßen vor:

1. Klicken Sie in der Symbolleiste auf das Symbol "'SchemaAgent-Validierung"  oder wählen Sie den Menüeintrag **Schema-Design | SchemaAgent-Validierung** Daraufhin wird das Dialogfeld "SchemaAgent-Validierung für..." (*Abbildung unten*) geöffnet. Hier können Sie angeben, ob nur das aktive Schema oder auch ein oder mehrere damit verknüpfte Schemas validiert werden sollen.

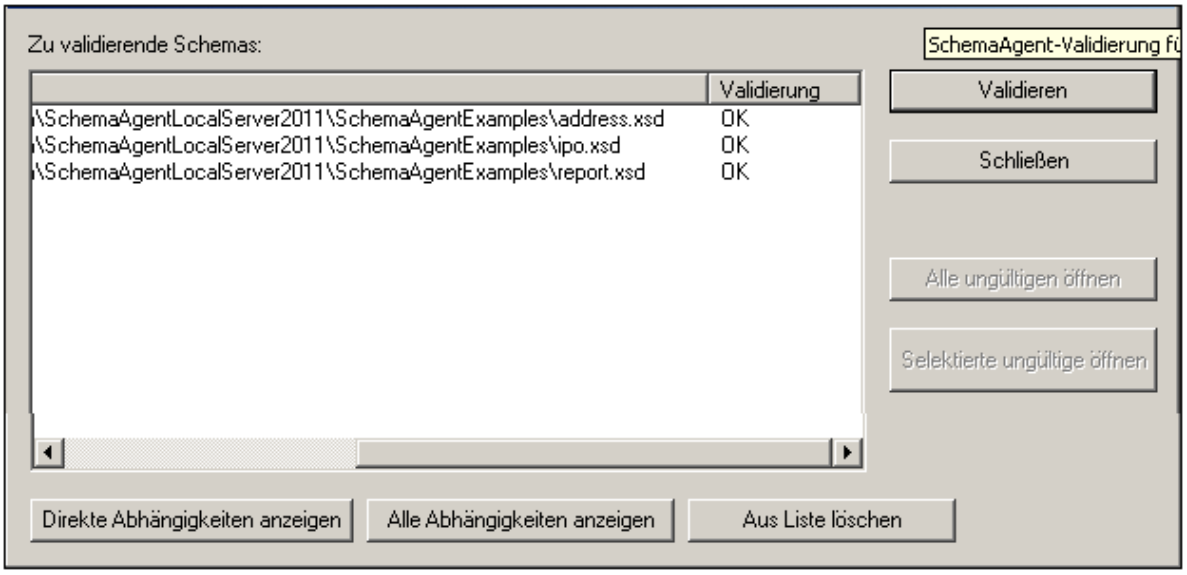


2. Um Schemas zur Liste hinzuzufügen, klicken Sie je nach Bedarf entweder auf **Direkte Abhängigkeiten anzeigen** oder auf **Alle Abhängigkeiten anzeigen**. In diesem Beispiel haben wir die Option **Alle Abhängigkeiten anzeigen** gewählt, woraufhin alle direkt oder indirekt referenzierten Dateien in die Liste übernommen werden.



   Sie können zu diesem Zeitpunkt bei Bedarf ein Schema aus der Liste entfernen ("Aus Liste löschen"').
3. Klicken Sie auf die Schaltfläche "Validieren", um alle Schemas im Listenfeld zu validieren.

 © 2016-2022 Altova GmbH



In der Spalte "Validieren" sehen Sie, ob die Validierung erfolgreich war oder fehlgeschlagen ist.

Sie können nun alle ungültigen Schemas oder nur eine Gruppe von ausgewählten ungültigen Schemas in XMLSpy öffnen.

6.8 In Schemas suchen

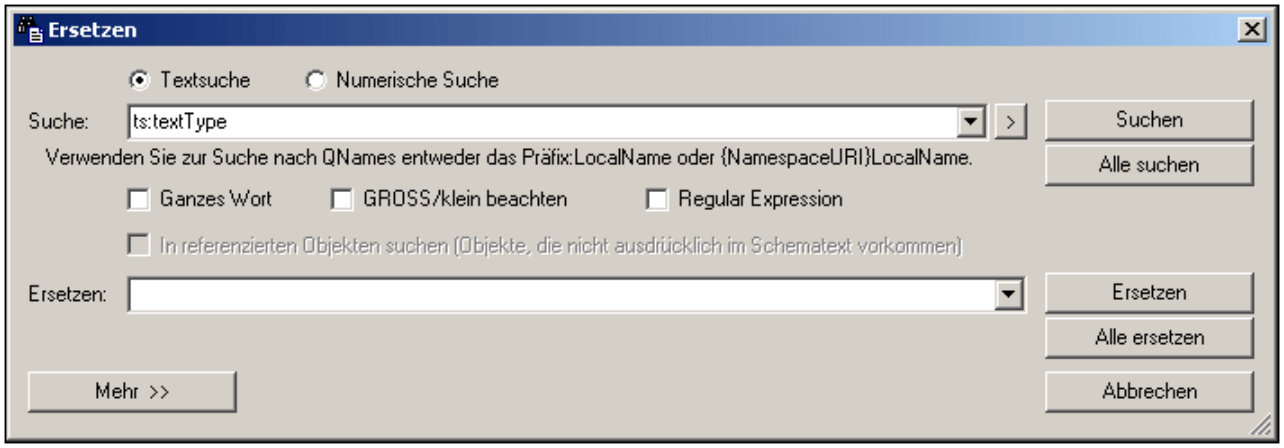
In der Schema-Ansicht können XML-Schemas mit Hilfe der XMLSpy-Funktion zum Suchen und Ersetzen in der Schema-Ansicht gesucht werden.

Die Funktion zum Suchen und Ersetzen in der Schema-Ansicht ist aktiviert, wenn in der Schema-Ansicht ein Schema aktiv ist. Es gibt zwei Möglichkeiten, diese aufzurufen:

- über die Menübefehle **Bearbeiten | Suchen** und **Bearbeiten | Ersetzen**.
- über die Schaltflächen **Suchen** und **Ersetzen** im Fenster "In Schemas suchen".

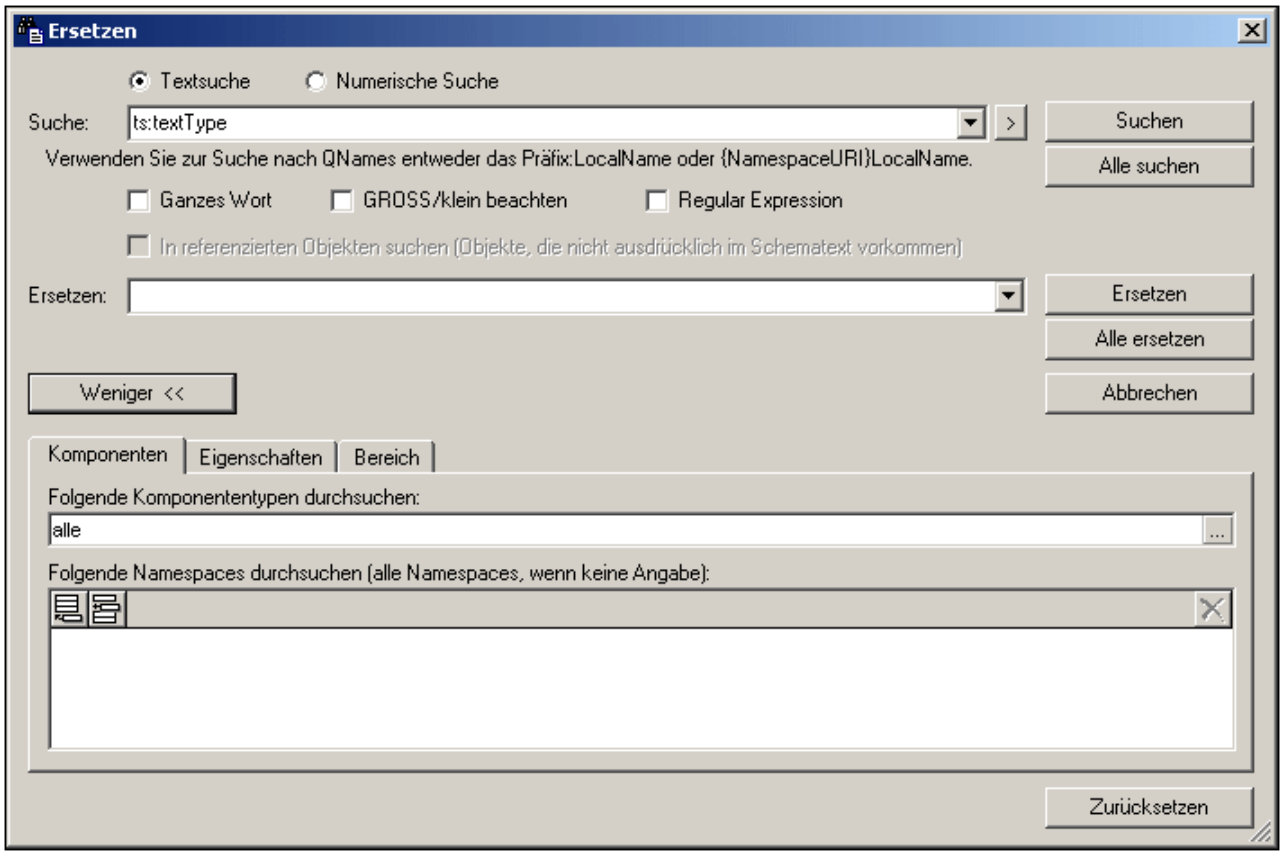
Bei Aufruf eines der beiden Befehle wird das Dialogfeld "Suchen" bzw. "Ersetzen" aufgerufen. Das Dialogfeld "Ersetzen" (*Abbildung unten*) hat im Gegensatz zum Dialogfeld "Suchen" ein Texteingabefeld für den Ersetzungsbegriff.

Das Standardersetzungsdialogfeld sieht folgendermaßen aus:



Wenn Sie auf die Schaltfläche **Mehr** klicken, wird das Dialogfeld erweitert und Sie sehen zusätzliche Suchkriterien (*Abbildung unten*).

© 2016-2022 Altova GmbH



Die Verwendungsweise ist die folgende:

- Geben Sie in die Textfelder "Suchen" und "Ersetzen" den Such- und den Ersetzungsbegriff ein[484].
- Geben Sie im Abschnitt "Komponenten" die gesuchte Schemakomponente ein[485].
- Definieren Sie die Eigenschaften der zu suchenden Komponenten[488]; auf diese Art können Sie die Suche eingrenzen
- Geben Sie den Bereich für die Suche[491] für das aktuelle Dokument oder Projekt ein oder geben Sie an, welcher Ordner durchsucht werden soll
- Führen Sie den Befehl aus[492]
- Verwenden Sie das Fenster "In Schemas suchen"[494], um schnell zu einer Komponente zu navigieren.

Mit der Schaltfläche **Zurücksetzen** um unteren Rand des Dialogfelds werden die folgenden Originaleinstellungen wiederhergestellt:

- Kein Suchbegriff, kein Ersetzungsbegriff
- Komponenten: `alle`
- Namespaces: keiner definiert
- Eigenschaftseinschränkungen: `überall`
- Zusätzliche Eigenschaftseinschränkungen: keine
- Bereich: aktuelle Datei

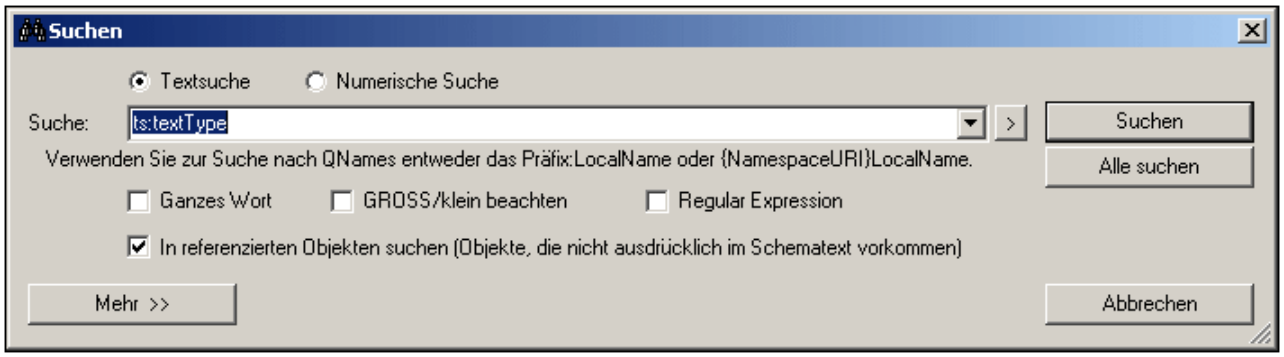
Anmerkung: Regular Expressions werden im Feld "Ersetzen" nicht unterstützt.

6.8.1 Suchbegriff

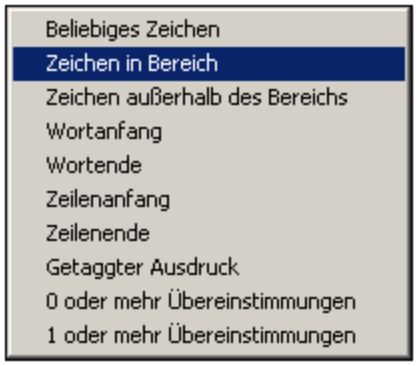
Der Suchbegriff kann als String (durch Auswahl des Optionsfelds **String**) oder als Zahl (durch Auswahl des Optionsfelds **Numerisch**) eingegeben werden.

String-Suche

In einer String-Suche (*Abbildung unten*) kann es sich beim Eintrag um (i) Text; (ii) einen QName; oder (iii) eine Regular Expression handeln. In QName-Suchen wird der Namespace entweder auf Basis des im Dokument verwendeten Präfix oder durch die Namespace URI festgelegt. Eine dieser beiden Optionen muss eingegeben werden. In der Abbildung unten ist das `ts:` Präfix das im Dokument zur Identifizierung eines bestimmten Namespace verwendete Präfix.



Zur Suche mittels einer Regular Expression aktivieren Sie das Kontrollkästchen "Regular Expression" und geben Sie dann die Regular Expression ein. Eingabehilfen für Regular Expressions stehen in einem Menü zur Verfügung, das durch Klicken auf den Rechtspfeil rechts vom Eingabefeld "Suche" (*Abbildung unten*) aufgerufen wird.



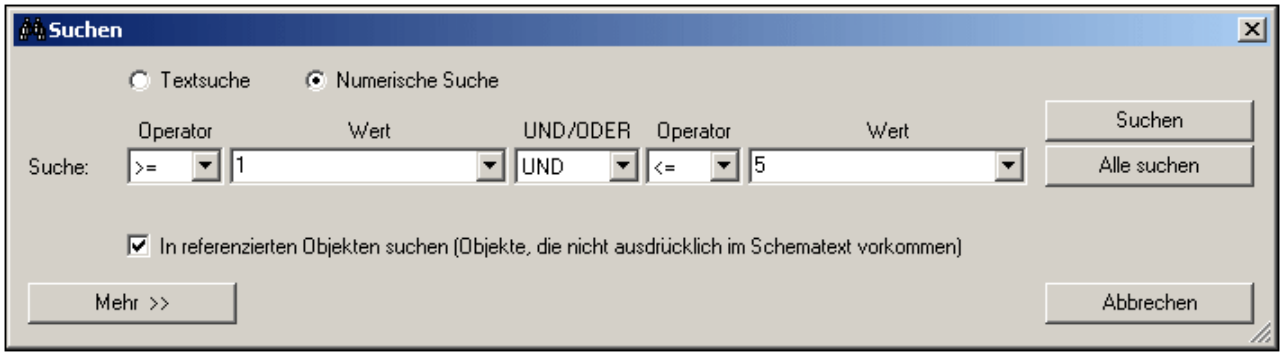
Sie können außerdem auswählen, ob ein Suchbegriff mit einem ganzen Wort im Dokument übereinstimmen muss und/oder ob die Groß- und Kleinschreibung im Dokument berücksichtigt werden soll. Verwenden Sie die Kontrollkästchen unterhalb des Texteingabefelds, um diese Optionen zu definieren.

Wenn in referenzierten Objekten (wie z.B. einer complexType-Definition oder einem globales Element) gesucht werden soll, aktivieren Sie das Kontrollkästchen "In referenzierten Objekten suchen". Diese Option steht nur im Dialogfeld "Suchen" zur Verfügung; im Dialogfeld "Ersetzen" ist sie deaktiviert.

© 2016-2022 Altova GmbH

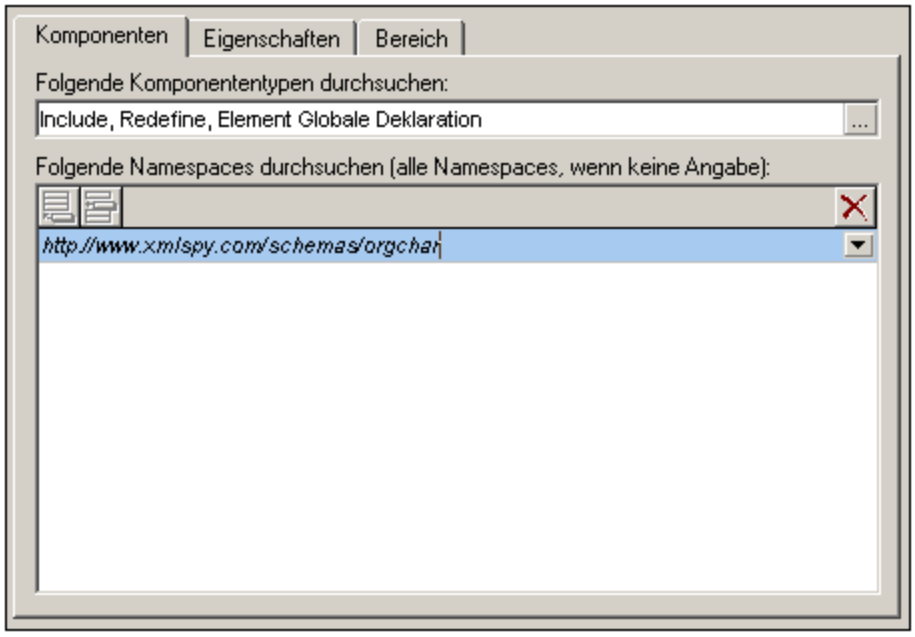
Numerische Suche

Wenn das Optionsfeld "Numerische Suche" ausgewählt ist, kann es sich beim Suchbegriff um einen einzigen Operator-mit-Zahl-Suchparameter handeln oder um eine Gruppe von zwei solchen Operator-mit-Zahl-Suchparameter, die durch die logische Verbindung AND oder OR miteinander verbunden sind. In der Abbildung unten sehen Sie zwei Suchbegriffsparameter für alle Ganzzahlen ab 1 und bis einschließlich 5.



6.8.2 Komponenten

Die Suche kann auf eine oder mehrere Komponententypen und auf einen oder mehrere Target Namespaces eingeschränkt werden. Diese Optionen stehen auf dem Register "Komponenten" zur Verfügung. Erweitern Sie das Dialogfeld "Suchen" oder "Ersetzen" durch Klicken auf die Schaltfläche **Mehr.** Daraufhin werden die Register zum Eingrenzen der Suche angezeigt. Eines dieser Register ist das Register "Komponenten" (*Abbildung unten*).

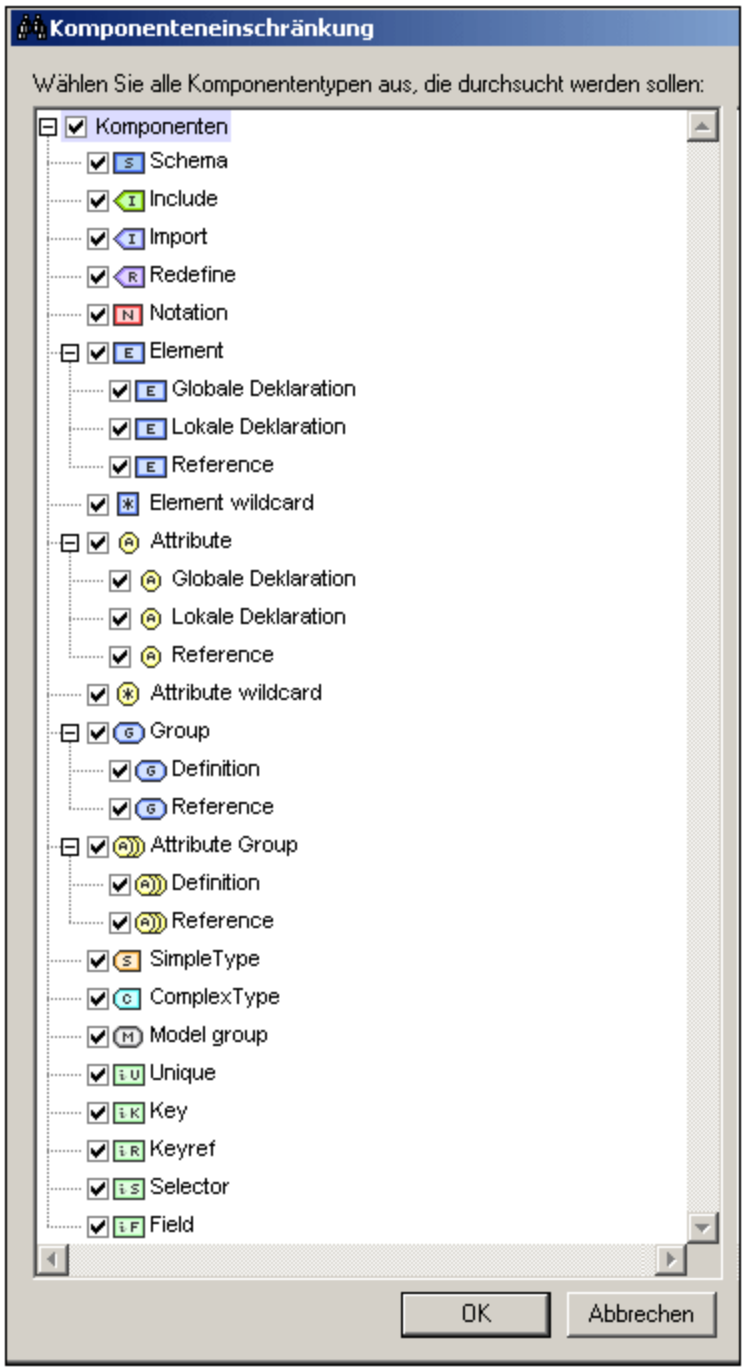


Das Register "Komponenten" besteht aus zwei Teilen: (i) einem Bereich zur Auswahl der zu durchsuchenden Komponententypen und (ii) einem Bereich zur Auswahl der zu durchsuchenden Target Namespaces.

Komponentenauswahl

Durch Klicken auf die Schaltfläche **Hinzufügen**  rechts vom Textfeld (*siehe Abbildung oben*) können Sie die zu durchsuchenden Komponententypen angeben. Daraufhin wird das Dialogfeld "Komponenteneinschränkung" (*Abbildung unten*) aufgerufen, in dem Sie die zu durchsuchenden Komponenten durch Anhaken auswählen können. Wenn Sie den Eintrag `Komponenten` ganz oben in der Liste aktivieren, werden alle Komponenten (Texteintrag: `alle`) ausgewählt. Wenn Sie den Eintrag deaktivieren, werden alle Komponenten - einschließlich aller einzeln ausgewählten Komponenten - deaktiviert (Texteintrag: `keine`). Einzelne Komponenten können daher nur ausgewählt werden, wenn der Eintrag `Komponenten` deaktiviert ist. Die ausgewählten Komponenten werden in das Textfeld als kommagetrennte Liste eingegeben (*siehe Abbildung oben*).

© 2016-2022 Altova GmbH



Anmerkung: Jedes Mal, wenn das Register "Komponenten" oder das Dialogfeld "Suchen/Ersetzen" geöffnet wird, wird die vorherige Komponentenauswahl beibehalten.

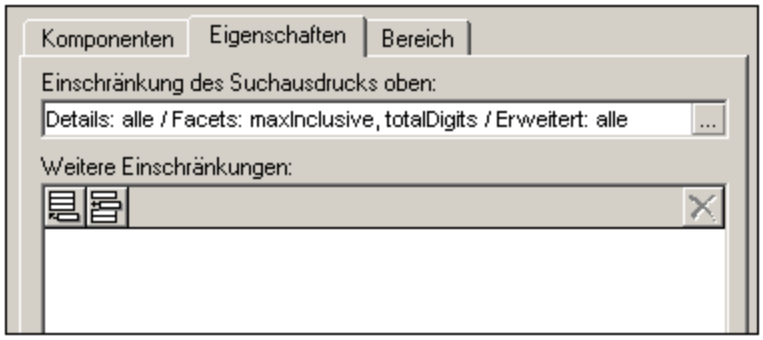
Namespace-Auswahl

Um einen oder mehrere Target Namespaces für die Suche auszuwählen, klicken Sie auf die Schaltfläche **Hinzufügen** oder **Einfügen** und geben Sie den gewünschten Namespace / die gewünschten Namespaces ein.

Wenn kein Target Namespace definiert ist, werden alle Target Namespaces durchsucht. Um einen in diesen Bereich eingegebenen Target Namespace zu löschen, wählen Sie den Target Namespace aus und klicken Sie auf die Schaltfläche **Löschen.**

6.8.3 Eigenschaften

Die Suche kann mit Hilfe von Optionen vom Register "Eigenschaften" auf eine oder mehrere Komponenteneigenschaften (Details und Facets) eingeschränkt werden oder muss dem Inhalt von Eigenschaften entsprechen. Erweitern Sie das Dialogfeld "Suchen" oder "Ersetzen", indem Sie auf die Schaltfläche **Mehr** klicken und wählen Sie anschließend das Register "Eigenschaften" aus (*Abbildung unten*).

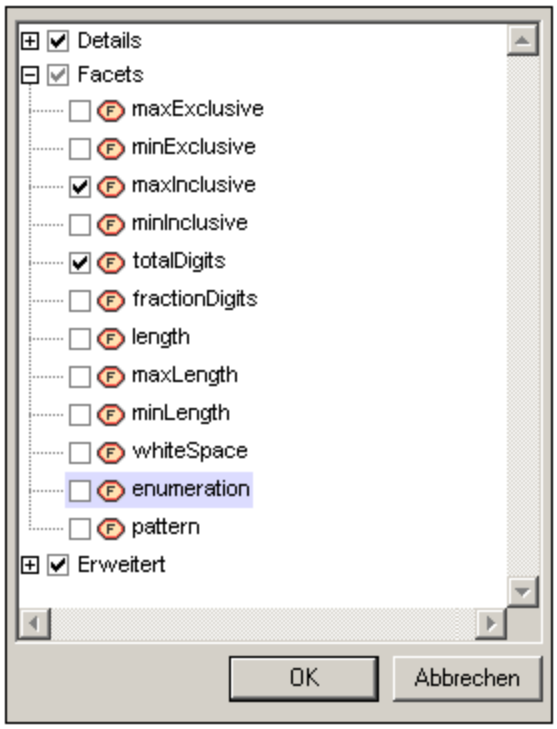


Das Register "Eigenschaften" besteht aus zwei Teilen: (i) einem Bereich zum Einschränken des Hauptsuchbegriffs (der in das Feld "Suchen" eingegeben wird) und (ii) einem Bereich zum Hinzufügen zusätzlicher Inhaltseinschränkungen (mit eigenen Kriterien); *siehe Abschnitt Zusätzliche Einschränkungen 488 unten.*

Eigenschaftsauswahl

Durch Klicken auf die Schaltfläche **Hinzufügen** ... rechts vom Textfeld (*siehe Abbildung oben*) können Sie die zu durchsuchenden Eigenschaftstypen angeben. Daraufhin wird das Dialogfeld "Komponenteneinschränkung" (*Abbildung unten*) aufgerufen, in dem Sie die zu durchsuchenden Eigenschaften durch Anhaken auswählen können. Die Eigenschaften werden in drei Gruppen eingeteilt: (i) Details; (ii) Facets (iii) Erweitert (wie z.B: die Eigenschaft DerivedFrom). Wenn Sie "Details", "Facets" oder "Erweitert" aktivieren, werden alle Eigenschaften in dieser Gruppe ausgewählt. Wenn Sie eine Gruppe deaktivieren, werden alle Eigenschaften in dieser Gruppe - einschließlich aller einzeln ausgewählten Eigenschaften - deaktiviert. Einzelne Eigenschaften können daher nur ausgewählt werden, wenn der Eintrag für die Gruppe deaktiviert ist. Die ausgewählten Eigenschaften werden in das Textfeld eingegeben.

© 2016-2022 Altova GmbH

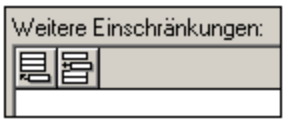


Anmerkung: Jedes Mal, wenn das Register "Eigenschaften" oder das Dialogfeld "Suchen/Ersetzen" geöffnet wird, wird die vorherige Eigenschaftsauswahl beibehalten.

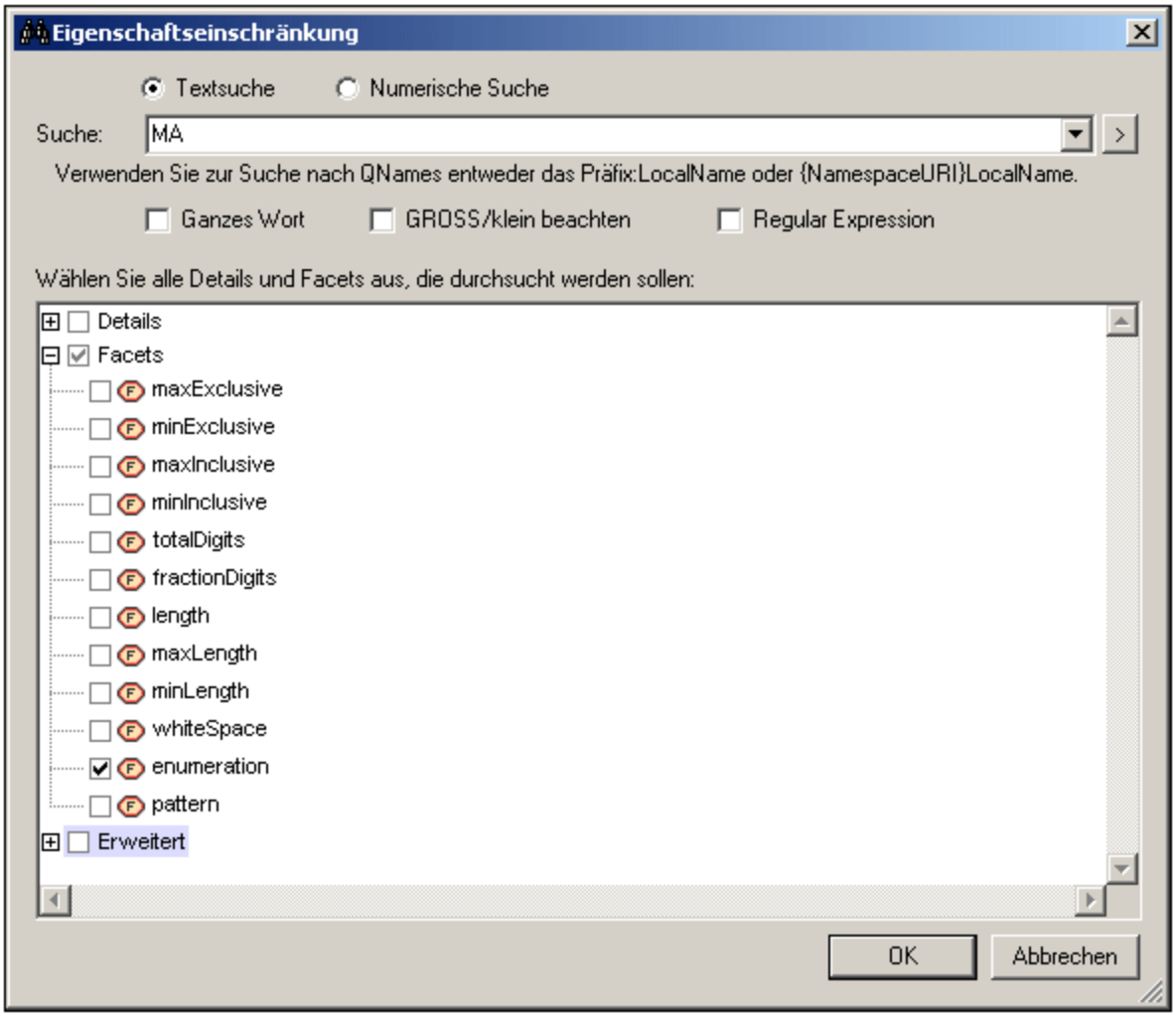
Zusätzliche Einschränkungen

Mit Hilfe einer zusätzlichen Einschränkung können Sie den Wert der zu suchenden Eigenschaft definieren. Wenn Sie z.B. ein Element namens `state` suchen, das eine Enumeration `MA` (für den US-Bundesstaat Massachusetts) hat, könnten Sie den Wert `MA` der Eigenschaft `enumeration` mit der Option "Weitere Einschränkungen" definieren. Gehen Sie dazu folgendermaßen vor:

1. Klicken Sie im Fenster "Weitere Einschränkungen" auf die Schaltfläche **Hinzufügen** oder **Einfügen** (*Abbildung unten*).

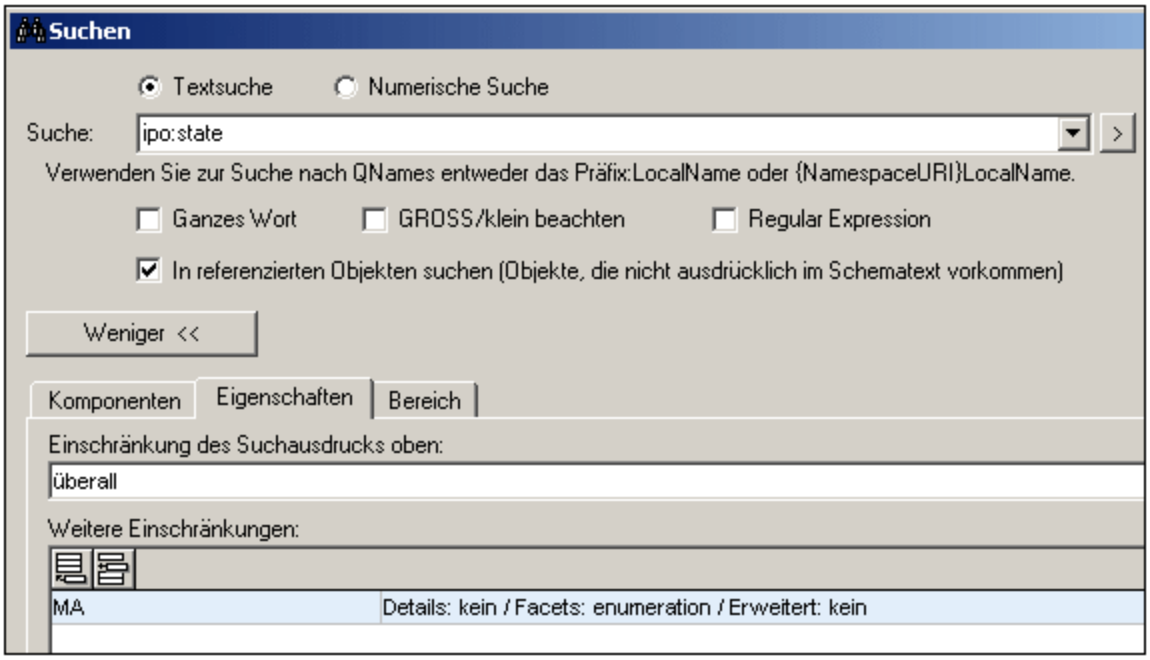


2. Daraufhin wird eine Zeile zum Fenster hinzugefügt und es erscheint das Dialogfeld "Eigenschaftseinschränkung". Deaktivieren Sie alle Eigenschaften und wählen Sie nur die Eigenschaft `enumeration` aus (*Abbildung unten*).



3. Geben Sie in das Textfeld oben im Dialogfeld den zu suchenden Enumerationswert ein, in diesem Fall `MA` (*siehe Abbildung oben*).
4. Klicken Sie auf **OK**. Die zusätzliche Einschränkung wird in die neu erstellte Zeile im Bereich "Weitere Einschränkungen" eingegeben (*Abbildung unten*).

© 2016-2022 Altova GmbH



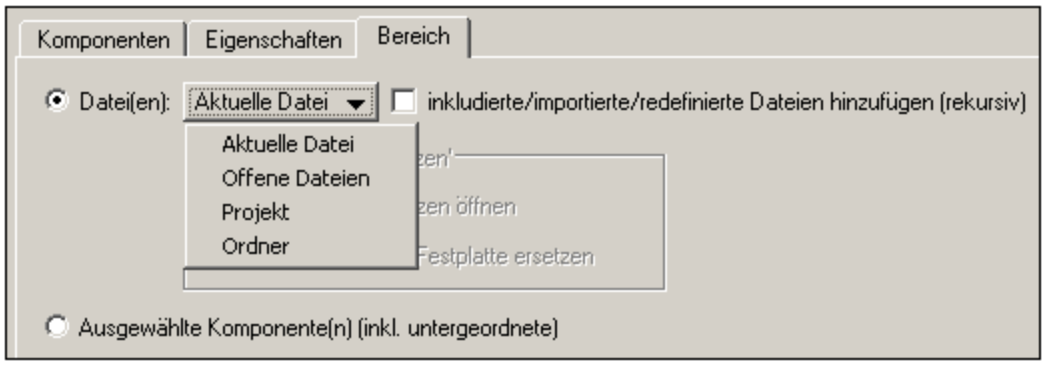
Beachten Sie in der Abbildung oben, dass der Suchbegriff `ipo:state` ist. Auf dem Register "Eigenschaften" gibt `anywhere` an, dass alle Eigenschaften durchsucht werden, doch durch die zusätzliche Einschränkung wird festgelegt, dass die Suche auf Enumerationen mit dem Wert `MA` eingeschränkt werden soll.

Zur weiteren Einschränkung der Suche können mehrere zusätzliche Einschränkungen hinzugefügt werden. Um eine dieser Einschränkungen wieder zu löschen, wählen Sie sie aus und klicken Sie auf die Schaltfläche **Löschen.**

Anmerkung: Jedes Mal, wenn das Register "Eigenschaften" oder das Dialogfeld "Suchen/Ersetzen" geöffnet wird, werden die vorherigen zusätzlichen Einschränkungen beibehalten.

6.8.4 Bereich

Der Bereich für die Suche kann auf dem Register "Bereich" (*Abbildung unten*) definiert werden. Sie können entweder eine Datei/Dateien oder die aktuell ausgewählte Schemakomponente in der Schema-Ansicht auswählen.



Wenn die Option Datei(en) ausgewählt ist, können Sie eine der Optionen aus den folgenden auswählen:

- *Aktuelle Datei:* Es steht eine zusätzliche Option zum Durchsuchen inkludierter, importierter und mit Redefine umdefinierter Dateien zur Verfügung.
- *Offene Dateien:* Alle XML-Schema (XSD)-Dateien, die in XMLSpy geöffnet sind. Nur die Befehle "Alle suchen" und "Alle ersetzen" sind aktiviert; die schrittweise Suche steht nicht zur Verfügung.
- *Projekt:* Das aktuelle aktive Projekt wird mit der Option, externe Ordner zu überspringen, ausgewählt. Nur die Befehle "Alle suchen" und "Alle ersetzen" sind aktiviert; die schrittweise Suche steht nicht zur Verfügung. Wenn die Standardansicht (**Extras | Optionen | Dateiarten | Standardansicht**) für die Dateierweiterung `.xsd` nicht die Schemaansicht ist, werden die. `xsd` Dateien nicht durchsucht.
- *Ordner:* Sie können nach dem gewünschten Ordner suchen. Es steht auch eine Option zum Durchsuchen von Unterordnern zur Verfügung. Nur die Befehle "Alle suchen" und "Alle ersetzen" sind aktiviert; die schrittweise Suche steht nicht zur Verfügung. Wenn die Standardansicht (**Extras | Optionen | Dateiarten | Standardansicht**) für die Dateierweiterung `.xsd` nicht die Schemaansicht ist, werden die. `xsd` Dateien nicht durchsucht.
- *Inkludierte, importierte und redefinierte Dateien* können durch Aktivieren der entsprechenden Optionen in den Bereich eingeschlossen werden.

Im Dialogfeld "Ersetzen" können Sie wählen, ob der Ersetzungsbegriff in die Datei auf der Festplatte kopiert werden soll oder ob die Datei in XMLSpy geöffnet werden soll. Wählen Sie dazu im Dialogfeld die entsprechende Option aus.

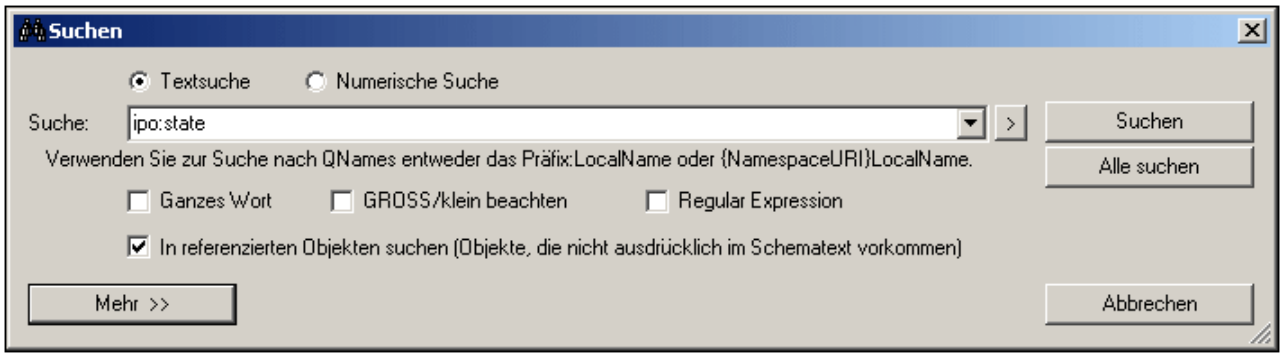
6.8.5 Die Befehle "Suchen" und "Ersetzen"

Der Befehl **Suchen** verhält sich in den Dialogfeldern "Suchen" und "Ersetzen" unterschiedlich. Das Verhalten des Befehls **Suchen** in beiden Dialogfeldern und des Befehls **Ersetzen** wird unten beschrieben.

Dialogfeld "Suchen"

Nach Eingabe des Suchbegriffs und anderer optionaler Kriterien zur Verfeinerung der Suche können Sie entweder auf den Befehl **Suchen (F3)** oder **Alle suchen** klicken *(Abbildung unten*).

© 2016-2022 Altova GmbH



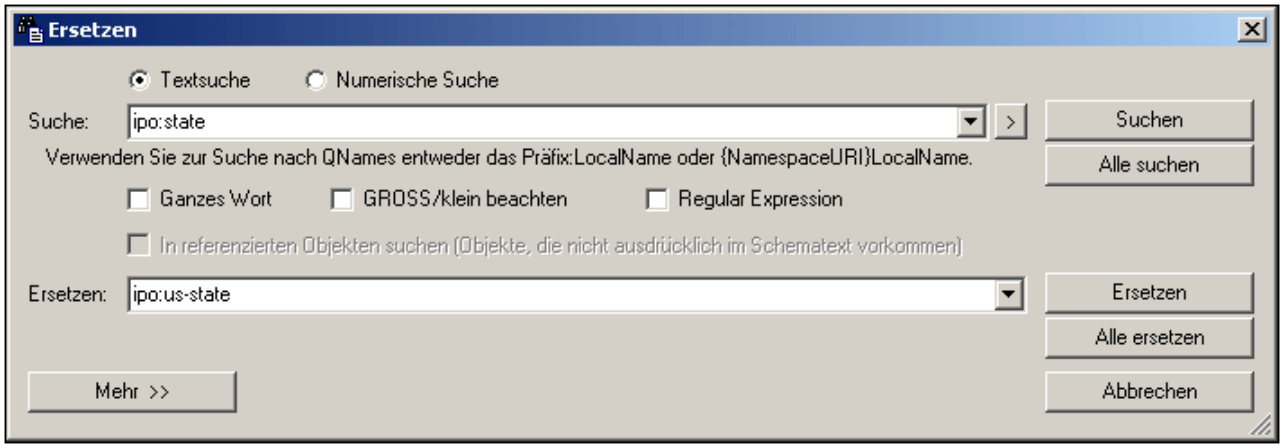
Bei Auswahl des Befehls **Suchen (Strg+F3)** wird das Dialogfeld **Suchen** geschlossen und die nächste Instanz des Suchbegriffs wird im definierten Bereich und unter Berücksichtigung der Suchkriterien gesucht. Die nächste Instanz wird relativ zur aktuell ausgewählten Komponente in der Schema-Ansicht gesucht. Wenn bei der Suche das Ende des Bereichs erreicht wird, beginnt sie nicht automatisch am Anfang des Bereichs. Sie müssen daher sicherstellen, dass sich das gerade in der Schema-Ansicht ausgewählte Dokument vor dem Teil des Dokuments befindet, den Sie durchsuchen möchten.

Das Ergebnis des Befehls **Suchen** wird in der Schema-Ansicht markiert und ebenfalls im Fenster "In Schemas suchen" ausgegeben. Im Fenster "In Schemas suchen" können Sie auf ein Ergebnis klicken, um diese Stelle in der Schema-Ansicht zu markieren.

Wenn Sie den Befehl **Alle suchen** auswählen, wird das Dialogfeld "Suchen" geschlossen und alle Suchergebnisse werden im Fenster "In Schemas suchen" aufgelistet.

Dialogfeld "Ersetzen"

Im Dialogfeld "Ersetzen" (*Abbildung unten*), wird bei Klicken auf den Befehl **Suchen** die nächste Instanz des Suchbegriffs im Verhältnis zur aktuellen Auswahl in der Schema-Ansicht gesucht. Sie können diese Instanz anschließend durch Klicken auf **Ersetzen** ersetzen.



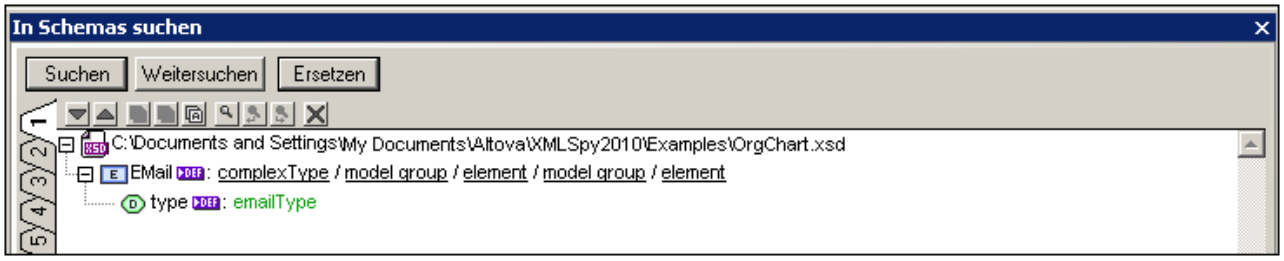
Mit dem Befehl **Alle suchen** wird das Dialogfeld "Ersetzen" geschlossen und alle Suchergebnisse werden im Fenster "In Schemas suchen" aufgelistet.

Mit dem Befehl **Alle ersetzen** werden alle Instanzen des gefundenen Elements ersetzt, das Dialogfeld "Ersetzen" wird geschlossen und die gefundenen Elemente werden im Dialogfeld "In Schemas suchen" aufgelistet.

Anmerkung: Regular Expressions werden im Feld "Ersetzen" nicht unterstützt.

6.8.6 Ergebnisse und Informationen

Jedes Mal, wenn einer der Befehle **Suchen, Alle suchen, Ersetzen** oder **Alle ersetzen** ausgeführt wird, werden die Ergebnisse der Ausführung im Fenster "In Schemas suchen " angezeigt (*Abbildung unten*). Der Begriff, nach dem gesucht wurde, wird grün angezeigt; (In der Abbildung unten sehen Sie, dass nach "email" gesucht wurde, wobei die Groß- und Kleinschreibung nicht berücksichtigt wurde). Beachten Sie, dass auch der Pfad der Schemadatei angegeben wird.



Mit den Befehlen **Alle Suchen** und **Alle ersetzen** werden die im Dokument gefundenen Instanzen aufgelistet.

Anmerkung: Mit den Schaltflächen **Suchen** und **Ersetzen** ganz oben im Fenster wird das Dialogfeld "Suchen" bzw. "Ersetzen" aufgerufen. Mit der Schaltfläche **Weitersuchen** kann die nächste Instanz des Suchbegriffs gesucht werden.

Funktionen des Fensters "In Schemas suchen"

Die Ergebnisse werden auf neun separaten Registern (mit der Nummer 1 bis 9) angezeigt. Sie können dadurch die Ergebnisse einer Suche auf einem Register aufbewahren, eine neue Suche durchführen und die Ergebnisse vergleichen. Wenn Sie im Fenster "In Schemas suchen" auf ein Ergebnis klicken, wird die entsprechende Komponente im Hauptfenster der Schema-Ansicht angezeigt und markiert. Auf diese Art können Sie schnell zur gewünschten Komponente navigieren.

In der Symbolleiste "In Schemas suchen" stehen die folgenden Befehle zur Verfügung:

- Mit den Schaltflächen **Nächstes** und **Vorheriges** wird die im Verhältnis zur aktuell ausgewählten Meldung jeweils nächste bzw. vorherige ausgewählt
- Mit dem Befehlen zum **Kopieren** von Meldungen werden die ausgewählte Meldung, die ausgewählte Meldung und ihre untergeordneten Einträge bzw. alle Meldungen in die Zwischenablage kopiert.
- Mit den **Suchen**-Befehlen wird Text im Fenster "In Schemas suchen" gesucht.
- Mit dem Befehl **Löschen** werden alle Meldungen auf dem gerade aktiven Register gelöscht.

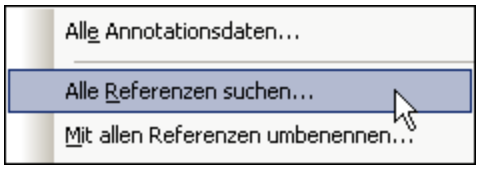
© 2016-2022 Altova GmbH

6.8.7 Suchen und Umbenennen von globalen Komponenten

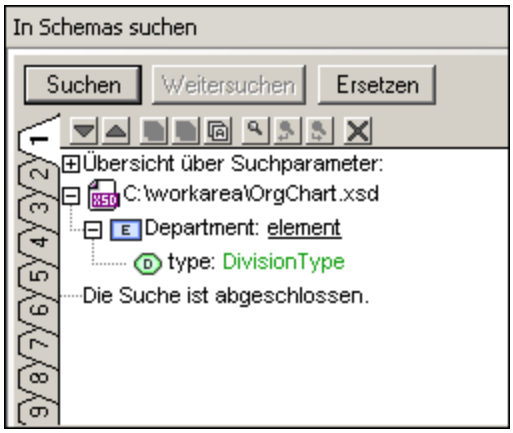
Benannte globale Komponente von XML-Schemas können in einer ausgewählten Datei und in allen Schema-Dateien, die mit der ausgewählten Datei in einer Beziehung stehen, gesucht und umbenannt werden. Benannte globale Komponenten sind alle globalen Komponenten mit Ausnahme von Include, Import, Redefine, Annotation, Comment und PI Comments.

Der Prozess läuft folgendermaßen ab:

1. In der Schema-Übersicht wird die globale Komponente, die gesucht oder umbenannt werden soll, markiert.
2. Wählen Sie im Kontextmenü, das erscheint wenn Sie mit der rechten Maustaste auf die markierte Komponente klicken, den erforderlichen Befehl (**Alle Referenzen suchen** oder **Mit allen Referenzen umbenennen**) aus.



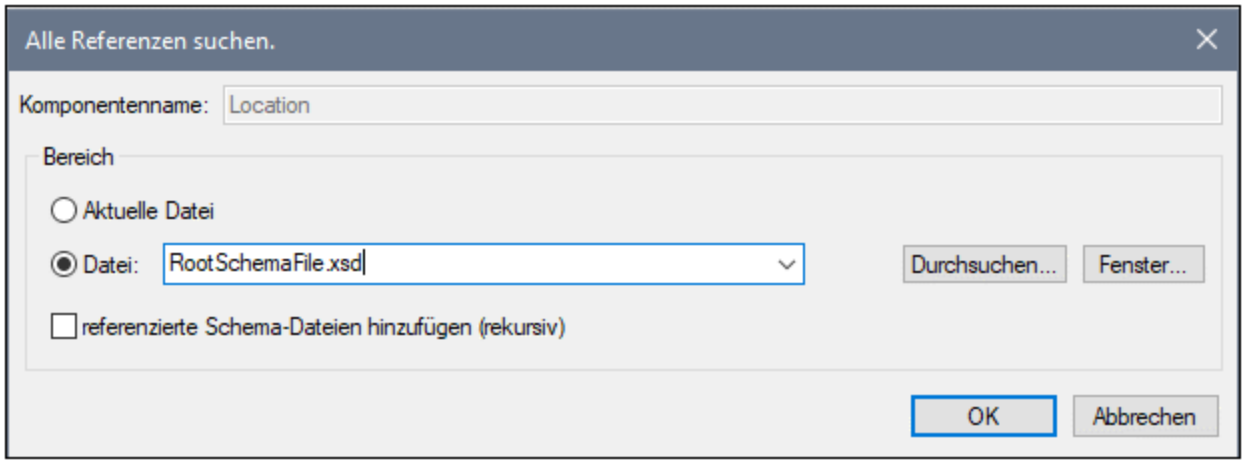
3. Wählen Sie im Dialogfeld, das erscheint, den Bereich der Suche (bzw. der Umbenennung). Geben Sie im Fall einer Umbenennung den neuen Namen der globalen Komponente ein.
4. Nachdem Sie auf **OK** geklickt haben, werden im Fenster "In Schemas suchen" die Suchergebnisse angezeigt (*Abbildung unten*).



Die Speicherorte aller Dateien, in denen Referenzen auf die globale Komponente gefunden werden, werden aufgelistet (*siehe Abbildung oben*). Alle umbenannten Komponente, die gefunden und umbenannt wurden, werden ebenfalls aufgelistet.

Alle Referenzen suchen

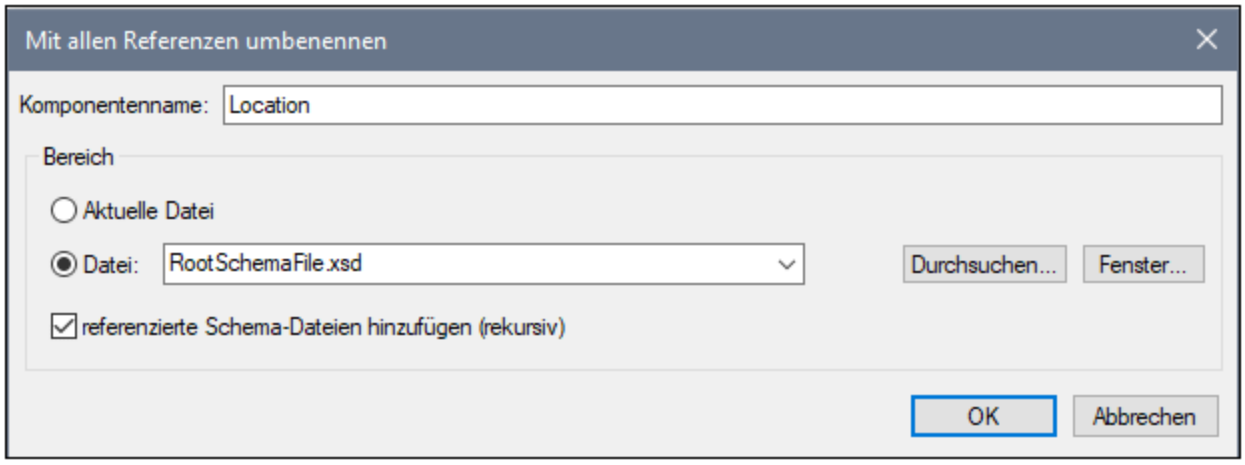
Um das Dialogfeld "Alle Referenzen suchen" (*Abbildung unten*) zu öffnen folgen Sie folgendem Ablauf: (i) Klicken Sie in der Schema-Übersicht mit der rechten Maustaste auf die globale Komponente, (ii) wählen Sie den Befehl **Alle Referenzen suchen** aus dem Kontextmenü, das erscheint.



Der Name der globalen Komponente wird im Feld *Komponentenname* angezeigt; das Feld ist ausgegraut und kann nicht bearbeitet werden. Sie können wählen, ob die aktuelle Datei oder eine andere Datei, nach der Sie Ihr Dateisystem durchsuchen können bzw. die Sie aus einer Liste geöffneter Dateien auswählen können, durchgeführt werden soll. Ferner können Sie dann angeben, ob nach verwandten Dateien (inkludiert, importiert, redefiniert) gesucht werden soll, indem Sie das Kontrollkästchen *referenzierte Schema-Dateien hinzufügen (rekursiv)* im unteren Teil des Dialogfelds aktivieren.

Mit allen Referenzen umbenennen

Um eine globale Komponente umzubenennen klicken Sie mit der rechten Maustaste darauf und wählen Sie **Mit allen Referenzen umbenennen** aus dem Kontextmenü, das erscheint. Dadurch wird das Dialogfeld "Mit allen Referenzen umbenennen" (*Abbildung unten*) aufgerufen.



Der neue Name, den Sie der ausgewählten globalen Komponente geben möchten, muss dann in das Textfeld *Komponentenname* eingegeben werden. Sie können wählen, ob die aktuelle Datei oder eine andere Datei, nach der Sie Ihr Dateisystem durchsuchen können bzw. die Sie aus einer Liste geöffneter Dateien auswählen können, durchgeführt werden soll. Ferner können Sie dann angeben, ob nach verwandten Dateien (inkludiert, importiert, redefiniert) gesucht werden soll, indem Sie das Kontrollkästchen *referenzierte Schema-Dateien hinzufügen (rekursiv)* im unteren Teil des Dialogfelds aktivieren.

© 2016-2022 Altova GmbH

7 XSLT

Altova Website: XSLT Editor

Dieser Abschnitt über XSLT ist in die folgenden Unterabschnitte gegliedert:

- Bearbeiten von XSLT-Dokumenten[499]: Beschreibung der Bearbeitungsfunktionen für XSLT-Dokumente in XMLSpy
- XSLT-Verarbeitung[501]: Beschreibung der verschiedenen Methoden, um über die XMLSpy GUI mit dem Prozessor Ihrer Wahl XSLT-Transformationen durchzuführen. In diesem Abschnitt werden auch wichtige Einstellungen in XMLSpy erläutert.
- XSL-Übersicht[504]: Beschreibung der XSL-Übersicht und des XSL-Info-Fensters, die zum Anzeigen, Navigieren und Verwalten einer Sammlung von XSLT-Dateien dienen.

XPath-Auswertung

Wenn ein XML-Dokument aktiv ist, können Sie XPath-Ausdrücke mit Hilfe des Fensters "XPath/XQuery"[124] auswerten. Dies ist eine äußerst nützliche Funktion, um schnell zu überprüfen, wie ein XPath-Ausdruck ausgewertet werden kann. Geben Sie einen XPath-Ausdruck ein und definieren Sie, ob er relativ zur Dokument-Root oder zu einem im XML-Dokument ausgewählten Kontext-Node ausgewertet werden soll. Das Ergebnis der Auswertung wird sofort im Fenster "XPath/XQuery" angezeigt. Eine Beschreibung zur Verwendung des Fensters "XPath/XQuery" finden Sie im Abschnitt Benutzeroberfläche und Umgebung | Fenster "XPath/XQuery"[124].

XSLT Profiler und Debugger

XMLSpy enthält auch einen XSLT Profiler[567] und einen XSLT Debugger[540] zu korrekten und effizienten Erstellung von XSLT Stylesheets. Diese beiden Features werden im Abschnitt XSLT und XQuery Debugger[540] beschrieben.

Weitere XSLT-Funktionen

Weitere detailliertere Beschreibungen zu den hier beschriebenen Features finden Sie in der Menübefehle in den Beschreibungen zu den entsprechenden Menübefehlen[1409].

Altova XSLT-Prozessoren

Nähere Informationen zur Implementierung des Altova XSLT 1.0, 2.0 und 3.0-Prozessors finden Sie in den Anhängen[2136] unter Informationen zum XSLT- und XQuery-Prozessor[2137].

RaptorXML für die Befehlszeile und Batch-Verarbeitung

Auf der XMLSpy Benutzeroberfläche können mittels der Projektfunktionen Batch-Verarbeitungen durchgeführt werden. Wenn Sie jedoch mehr Flexibilität benötigen, testen Sie das Altova-Produkt RaptorXML, welches Funktionen für die schnelle Validierung von XML-Dateien, die Transformation von XSLT-Dateien und die XQuery-Ausführung enthält. RaptorXML eignet sich ideal für XSLT-Transformationen über die Befehlszeile oder die Batch-Verarbeitung.

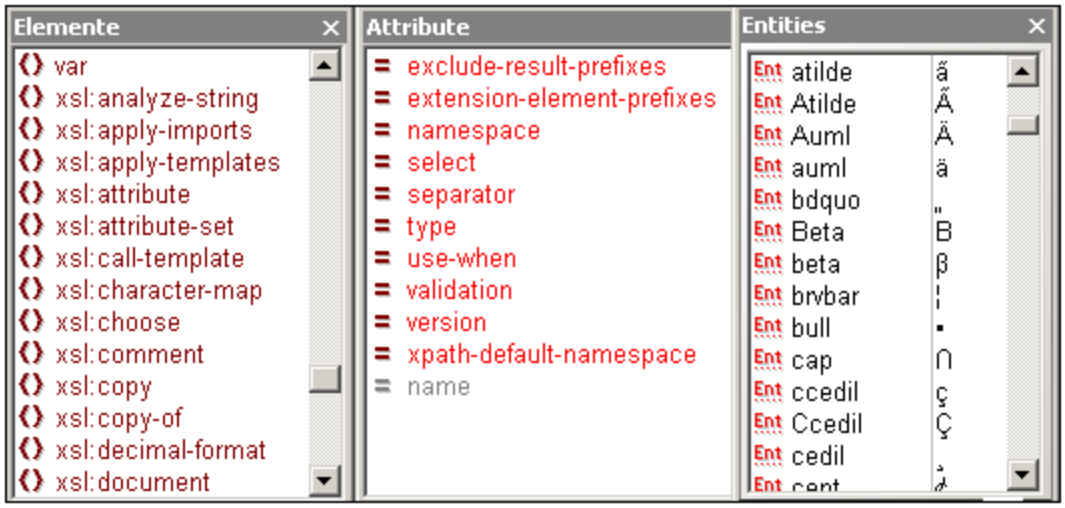
© 2016-2022 Altova GmbH

7.1 XSLT-Dokumente

XSLT 1.0-, 2.0- und 3.0-Dokumente können wie jedes andere XML-Dokument in der Textansicht[144] und der Grid-Ansicht[169] bearbeitet werden. Die Standardansicht, in der ein XSLT-Dokument geöffnet wird, kann im Dialogfeld "Optionen" im Abschnitt "Dateiarten" eingestellt werden.

Eingabehilfen

Eingabehilfen stehen für Elemente, Attribute und Entities zur Verfügung. Die Informationen über die in den Eingabehilfen angezeigten Einträge sind bereits in XMLSpy vordefiniert und sind nicht von im XSLT-Dokument enthaltenen Referenzen abhängig.



Beachten Sie bitte die folgenden Punkte:

1. Wenn Sie über das Dialogfeld "Neues Dokument anlegen" (**Datei | Neu**) ein XSLT-Dokument erstellen, werden, je nachdem welche Dokumentart angelegt wurde - XSLT 1.0, XSLT 2.0 oder XSLT 3.0, die entsprechenden XSLT-Elemente und Attribute in die Eingabehilfen geladen. Zusätzlich dazu werden HTML-Elemente und -Attribute sowie die HTML 4.0 Entity-Gruppen Latin-1, Sonderzeichen und Symbole geladen.
2. Wenn über das Dialogfeld "Neues Dokument anlegen" (**Datei | Neu**) ein XML-Dokument angelegt wird und XSLT-Inhalt erhält, stehen mit Ausnahme der XML-Zeichen-Entities keine Einträge in den Eingabehilfen zur Verfügung.
3. Wenn ein XSLT-Dokument geöffnet wird, das über das Dialogfeld "Neues Dokument anlegen" (**Datei | Neu**) erstellt wurde, stehen die Eingabehilfen wie in Punkt 1 zur Verfügung.
4. Wenn ein XSLT-Dokument geöffnet wird, das **nicht** über das Dialogfeld "Neues Dokument anlegen" (**Datei | Neu**) als XSLT-Dokument erstellt wurde, so stehen Eingabehilfen wie in Punkt 1 oben beschrieben zur Verfügung. Zusätzlich dazu werden in den Eingabehilfen der Textansicht XSL-FO-Elemente und -Attribute aufgelistet.
5. Die Präfixe von Elementen in der Element-Eingabehilfe lauten immer folgendermaßen: `xsl:` Präfix für XSLT-Elemente; kein Präfix für HTML-Elemente; `fo:` Präfix für XSL-FO-Elemente. Um die Eingabehilfen daher verwenden zu können, müssen die Namespace-Deklarationen im XSLT-Dokument Präfixe definieren, die mit den vordefinierten Präfixen in den Eingabehilfen übereinstimmen.

Autokomplettierung

In der Textansicht steht die Autokomplettierungsfunktion während der Eingabe in Form eines Popup-Fensters zur Verfügung, wobei der erste Eintrag in der Popup-Liste, der mit dem eingegeben Text übereinstimmt, markiert erscheint. Wenn Sie ein Element eingeben, erscheint eine Liste von Elementen, wobei die Vorschläge in alphabetischer Reihenfolge angezeigt werden und der passendste Eintrag markiert erscheint. Ähnlich dazu erscheint bei Eingabe eines Attributs eine Liste von Attributvorschlägen. Welche Einträge in der Liste angezeigt werden, ist abhängig von den im vorherigen Abschnitt über Eingabehilfen beschriebenen Regeln.

Intelligente XPath-Editierung

Die intelligente XPath-Editierung steht dort im XSLT-Dokument zur Verfügung, wo XPath-Ausdrücke eingegeben werden können (z.B. im Wert eines `select` Attributs, in Attributwertvorlagen und XSLT 3.0-Wertvorlagen). Die folgenden Funktionen stehen zur Verfügung:

- Syntaxfärbung für die XPath-Konstrukte einschließlich Klammerergänzung bei der Eingabe
- Ein eingeblendeter Tipp mit Informationen über die Funktion, wenn Sie den Cursor über eine XPath-Funktion platzieren.
- XPath-Funktionen und Achsen werden während der Eingabe in Popup-Fenstern vorgeschlagen. Mit den **Pfeil nach oben/Pfeil nach unten**-Tasten können Sie sich durch die Liste der Vorschläge bewegen. Wenn es sich bei der im Popup-Fenster markierten Funktion um eine Funktion handelt, werden in einem weiteren Popup-Fenster Informationen darüber (ihre Signatur) angezeigt
- Wenn im Fenster Info[508] eine XML-Datei zugewiesen wurde, so stehen im Popup-Fenster auch die Elemente und Attribute der XML-Datei zur Verfügung.

Validierung von XSLT-Dokumenten

Das XSLT-Dokument kann anhand des in XMLSpy vordefinierten XSLT-Schemas validiert werden. (klicken Sie auf **XML | Validieren (F8)**). Das richtige Schema wird automatisch ausgewählt, je nachdem, ob es sich um ein XSLT 1.0, XSLT 2.0 oder XSLT 3.0-Dokument handelt (dies ist im `version` Attribut, des `xsl:stylesheet` Elements definiert).

© 2016-2022 Altova GmbH

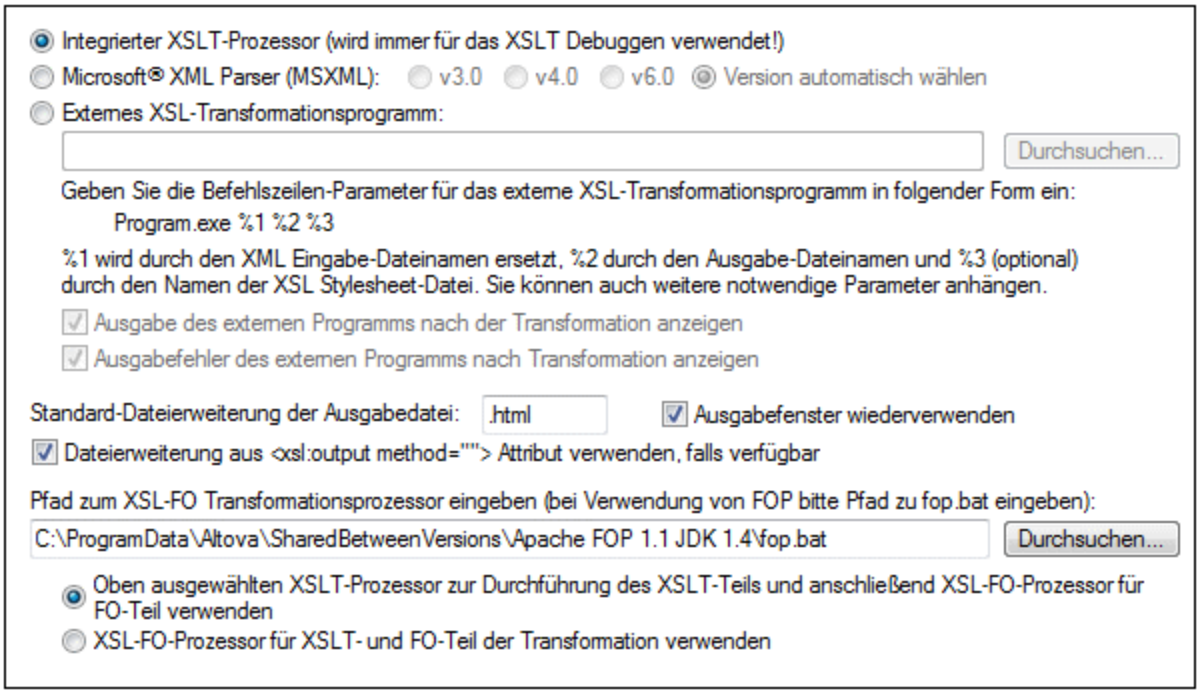
7.2 XSLT-Verarbeitung

Auf der Benutzeroberfläche von XMLSpy stehen zwei Arten von XSLT-Transformation zur Verfügung:

- Der Befehl **XSL/XQuery | XSL-Transformation (F10)** dient dazu, um anhand eines XSLT Stylesheet direkte XML-Transformationen in das in den Stylesheets definierte und beschriebene Ergebnisformat durchzuführen.
- Der Befehl **XSL/XQuery | XSL-FO-Transformation** wird verwendet: (i) für Transformationen von XML in FO in PFD, die in zwei Schritten durchgeführt werden und (ii) für Transformationen von FO in PDF (wird in einem Schritt durchgeführt).

Definieren des XSLT-Prozessors für die Transformation

Welcher XSLT-Prozessor für die Transformation verwendet wird, wird im Dialogfeld "Optionen" im Abschnitt XSL[1633] (*Abbildung unten*) definiert.



Die Optionen, die zur Verfügung stehen, werden im Abschnitt Menübefehle[1633] erläutert. Der Prozessor, der im Abschnitt "XSL" definiert ist, wird für alle XSLT-Transformationen verwenden. Beachten Sie, dass für die XSL-FO-Transformation ein zusätzlicher XSLT-Prozessor zur Verfügung steht: der XSLT-Prozessor, der mit einigen FO-Prozessoren mitgeliefert wird. Um diese Option auszuwählen, aktivieren Sie das entsprechende Optionsfeld um unteren Rand des Registers XSL (*siehe Abbildung oben*).

Definieren des FO-Prozessors

Der FO-Prozessor, der für Transformationen vom FO- ins PDF-Format verwendet werden soll, wird im Textfeld am unteren Rand des Registers XSL im Dialogfeld "Optionen"[1633] (*Abbildung oben*) definiert.

XSLT 1.0, 2.0, 3.0 und die XSLT-Prozessoren von Altova

Die XSLT-Version eines Stylesheet ist im `version` Attribut des `xsl:stylesheet` (oder `xsl:transform`)-Elements definiert. XMLSpy enthält den integrierten XSLT 1.0-, XSLT 2.0- und XSLT 3.0-Prozessor von Altova. Welcher Prozessor ausgewählt wird, hängt vom Wert des Versionsattributs (`1.0`, `2.0` oder `3.0`) ab.

XSLT-Transformation

Der Befehl **XSLT-Transformation (F8)** kann in den folgenden Szenarios verwendet werden:

- Um ein XML-Dokument zu transformieren, das auf der Benutzeroberfläche aktiv ist und dem ein XSLT-Dokument zugewiesen ist[1420]. Wenn kein XSLT-Dokument zugewiesen ist, werden Sie aufgefordert, eine Zuweisung vorzunehmen, wenn Sie den Befehl **XSLT-Transformation (F8)** aufrufen.
- Um ein XSLT-Dokument zu transformieren, das auf der Benutzeroberfläche aktiv ist. Wenn Sie den Befehl **XSLT-Transformation (F8)** aufrufen, werden sie aufgefordert, die XML-Datei anzugeben, die Sie mit dem aktiven XSLT-Stylesheet verarbeiten möchten.
- Um Projektordner und Dateien zu transformieren. Klicken Sie mit der rechten Maustaste auf den Projektordner bzw. die Datei und wählen Sie den Befehl aus.

Rückwärts-Mapping

Wenn die Rückwärts-Mapping[1418]-Funktion aktiviert ist, werden XSLT-Transformationen ausgeführt, sodass das Ergebnisdokument auf die XSLT+XML-Ausgangsdokumente rückgemappt werden kann. Wenn Sie auf einen Node im Ergebnisdokument klicken, werden die **XSLT-Anweisung** *und* die **XML-Quelldaten,** anhand welcher der jeweilige Ergebnis-Node generiert wurde, markiert. Wenn Sie außerdem auf eine XSLT-Anweisung oder einen XML-Daten-Node klicken, werden die entsprechenden Nodes in den anderen beiden Dokumenten markiert. Nähere Informationen finden Sie unter dem Befehl **XSL/XQuery | Rückwärts-Mapping aktivieren**[1418]**.**

XSL:FO-Transformation

Der Befehl **XSL:FO-Transformation** kann in den folgenden Szenarios verwendet werden:

- Zum Transformieren eines XML-Dokuments, das auf der Benutzerfläche aktiv ist und dem ein XSLT-Dokument zugewiesen ist[1420]. Das XML-Dokument wird zuerst mit dem dafür definierten XSLT-Prozessor ins FO-Format transformiert. Anschließend wird das FO-Dokument mit dem dafür definierten FO-Prozessor verarbeitet und es wird ein PDF-Dokument erzeugt. Wenn kein XSLT-Dokument zugewiesen ist, werden Sie aufgefordert, eine Zuweisung vorzunehmen, wenn Sie den Befehl **XSLT-Transformation (F8)** aufrufen.
- Um ein FO-Dokument mit dem festgelegten FO-Prozessor ins PDF-Format zu transformieren
- Um ein XSLT-Dokument zu transformieren, das auf der Benutzeroberfläche aktiv ist. Wenn Sie den Befehl **XSLT-Transformation (F8)** aufrufen, werden sie aufgefordert, die XML-Datei anzugeben, die Sie mit dem aktiven XSLT-Stylesheet verarbeiten möchten.
- Um Projektordner und Dateien zu transformieren. Klicken Sie mit der rechten Maustaste auf den Projektordner bzw. die Datei und wählen Sie den Befehl aus.

Eine Beschreibung zu den Optionen des Dialogfelds "XSL:FO-Ausgabe"[1412] finden Sie im Abschnitt "Menübefehle"[1412].

© 2016-2022 Altova GmbH

Parameter für XSLT

Wenn Sie die XSLT-Prozessoren von Altova verwenden, können XSLT-Parameter in einem praktischen GUI-Dialogfeld gespeichert werden. Alle gespeicherten Parameter werden jedes Mal, wenn eine Transformation durchgeführt wird, an das XSLT-Dokument übergeben. Nähere Informationen dazu finden Sie in der Beschreibung zum Befehl XSLT-Parameter / XQuery-Variable[1414].

Batch-Verarbeitung mit RaptorXML

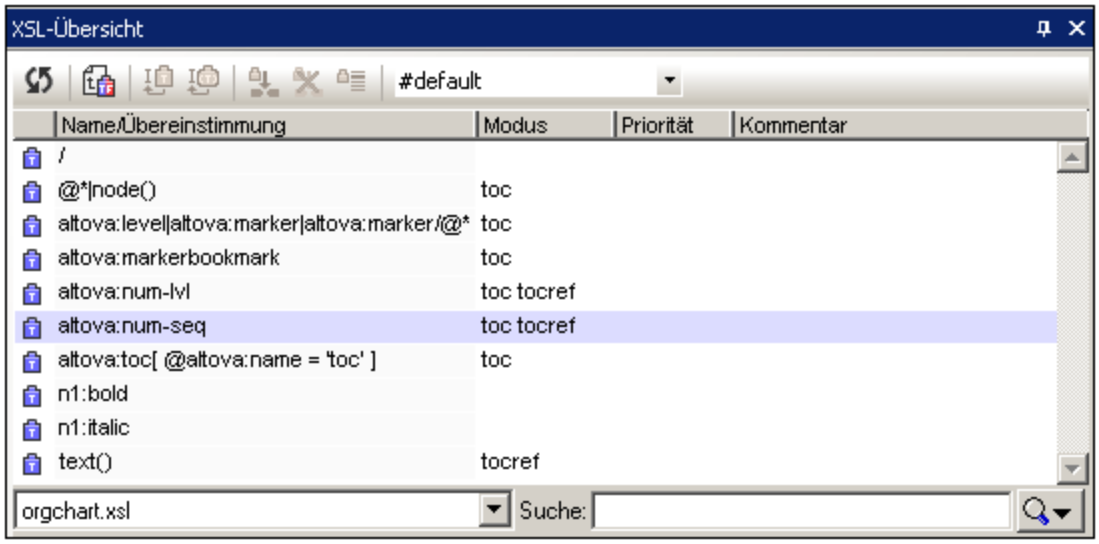
RaptorXML ist eine eigenständige Applikation, die den XML-Validator sowie den XSLT- und XQuery-Prozessor von Altova enthält. Sie kann über die Befehlszeile, eine COM-Schnittstelle, in Java-Programmen und in .NET-Applikationen verwendet werden, um XML-Dokumente zu validieren, XML-Dokumente mit XSLT-Stylesheets zu transformieren und XQuery-Dokumente auszuführen.

Mit Hilfe von RaptorXML können daher XSLT-Transformationen automatisiert werden. So können Sie z.B. eine Batch-Datei erstellen, die RaptorXML aufruft, um eine Gruppe von Dokumenten zu transformieren. Nähere Informationen dazu finden Sie in der Dokumentation zu RaptorXML.

7.3 XSL-Übersicht

Wenn ein XSLT-Dokument in XMLSpy das aktive Dokument ist, werden Informationen über die Struktur des Dokuments im Fenster "XSL-Übersicht" 505 und Informationen über die mit dem aktiven XSLT-Dokument in Zusammenhang stehenden Dateien auf dem Register XSLT des Info-Fensters 508 (welches nur angezeigt wird, wenn es sich beim aktiven Dokument in XMLSpy um ein XSLT-Dokument handelt) angezeigt. Des Weiteren stehen über diese beiden Fenster einige Befehle zur Verfügung, die die Bearbeitung des XSLT-Dokuments und die Verwaltung der damit in Zusammenhang stehenden Dateien erleichtern.

Im **Fenster "XSLT-Übersicht"** 505 (*Abbildung unten*) haben Sie folgende Möglichkeiten:

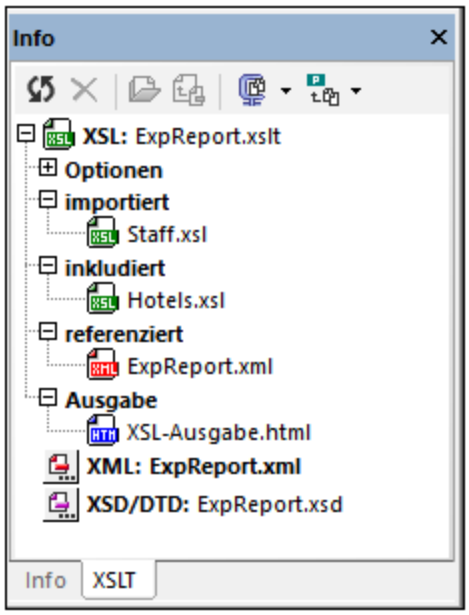


- Anzeigen der Vorlagen und Funktionen im aktiven XSLT-Dokument und in allen importierten und inkludierten XSLT-Dokumenten
- Sortieren der Vorlagen und Funktionen anhand ihrer Namen oder Suchausdrücke, des Modus, der Priorität oder der Kommentare
- Suchen nach bestimmten Vorlagen auf Basis ihres Namens/ihrer Ausdrücke
- Navigation zur entsprechenden Vorlage im XSLT-Dokument über die XSL-Übersicht
- Schnelles Einfügen von Aufrufen von Named Templates
- Definieren einer ausgewählten Named Template als Eintrittspunkt für Transformationen

Nähere Informationen finden Sie im Abschnitt Das Fenster "XSL-Übersicht" 505.

Auf dem **Register "XSLT" des Info-Fensters** 508 (*Abbildung unten*) haben Sie folgende Möglichkeiten:

© 2016-2022 Altova GmbH

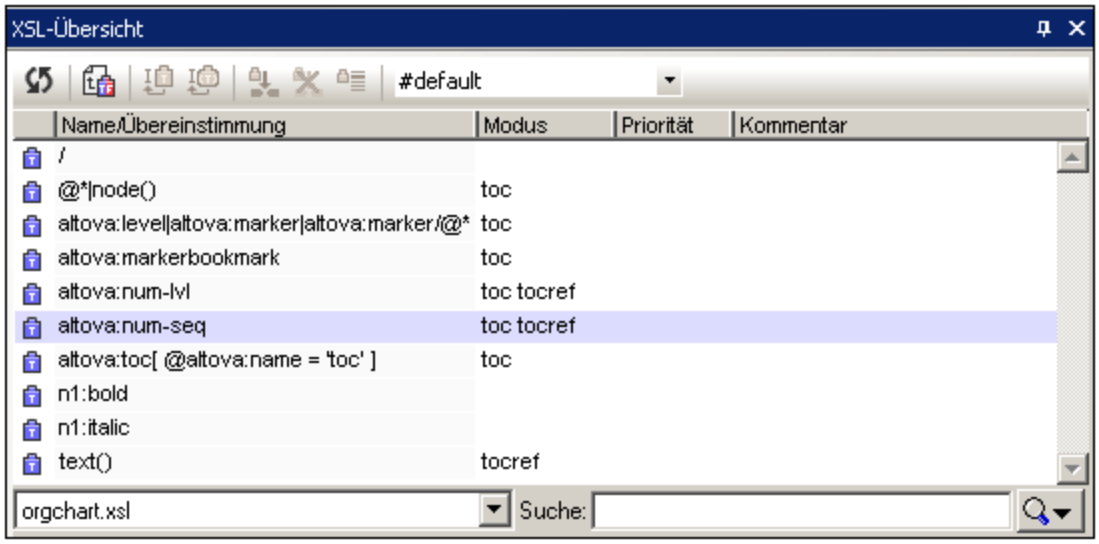


- Anzeigen von Informationen zu allen Dateien, die in Zusammenhang mit dem aktiven XSLT-Dokument stehen, - z.B. der Pfad der importierten und inkludierten Dateien
- Definieren einer XML-Datei für die Transformation mit dem aktiven XSLT-Dokument. Außerdem kann die Schema-Datei (XSD/DTD) zur Validierung der ausgewählten XML-Datei herangezogen werden
- Öffnen einer in Zusammenhang stehenden Datei vom Info-Fenster aus
- Schnelle Gliederung aller in Zusammenhang stehenden Dateien in XMLSpy-Projekte
- Zippen aller in Zusammenhang stehenden Dateien unter einem benutzerdefinierten Pfad

Das Fenster "XSL-Übersicht" 505 und das Register "XSLT" des Info-Fensters 508 werden in den Unterabschnitten dieses Abschnitts näher beschrieben.

7.3.1 Das Fenster "XSL-Übersicht"

Im Fenster "XSL-Übersicht" (*Abbildung unten*) sind alle Vorlagen und Funktionen im aktiven XSLT-Dokument aufgelistet. Vorlagen werden durch ein blaues Symbol gekennzeichnet (Vorlagen ohne einen Parameter; und Vorlagen, die Parameter enthalten). Funktionen werden durch ein rotes Symbol gekennzeichnet. In der Auswahlliste links unten im Fenster können Sie auswählen, ob die aufgelisteten Vorlagen und Funktionen: (i) nur aus dem aktiven XSLT-Dokument (wie in der Abbildung unten) oder (ii) aus dem aktiven XSLT-Dokument und allen inkludierten und importierten Stylesheets stammen sollen.



Es gibt zwei Arten von Vorlagen: (i) Named Templates, und (ii) Vorlagen, die mit einem XPath-Ausdruck übereinstimmen. Zu jeder Vorlage werden folgende Informationen angezeigt:

- der Name (wenn die Vorlagen ein `name` Attribut hat) und/oder der XPath-Ausdruck (wenn die Vorlage ein `match` Attribut hat). Wenn die Vorlage sowohl ein `name` als auch ein `match` Attribut hat, werden beide aufgelistet, wobei der Wert des Attributs `name` zuerst angeführt wird: `namevalue, matchvalue`.
- der Modus, falls definiert. Beachten Sie, dass eine Vorlage mehrere Modi haben kann (*siehe Abbildung oben*).
- die Priorität, falls definiert,
- der Kommentar, der - falls vorhanden - unmittelbar vor der Vorlage oder Funktionen steht.

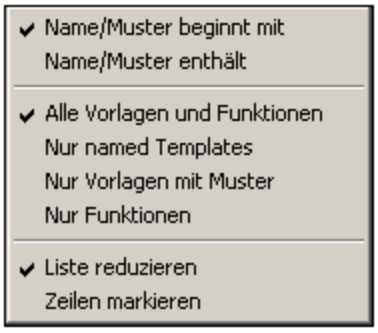
Funktionen im Stylesheet werden nach ihrem Namen aufgelistet. Funktionen haben weder einen Modus noch eine Priorität.

Operationen

Im Fenster "XSL-Übersicht" können die folgenden Operationen ausgeführt werden:

- *Filtern:* Die im Fenster angezeigte Liste kann gefiltert werden, sodass eine der folgenden Arten angezeigt wird: (i) alle Vorlagen und Funktionen (die Standardeinstellung bei jedem Start von XMLSpy); (ii) nur Named Templates; (iii) nur Vorlagen für XPath-Ausdrücke; (iv) nur Funktionen. Klicken Sie zur Auswahl des gewünschten Filters auf den Dropdown-Pfeil rechts von Suchfeld am rechten unteren Rand des Fensters (*Abbildung unten*) und wählen Sie den gewünschten Filter aus (die zweite Befehlsgruppe im Menü). Der ausgewählte Filter wird sofort angewendet und gilt solange, bis er geändert wird oder XMLSpy geschlossen wird.

© 2016-2022 Altova GmbH



- *Sortieren und Suchen:* Jede Spalte kann durch Klicken auf die Spaltenüberschrift in alphabetischer Reihenfolge sortiert werden. Mit jedem Klick auf die Spaltenüberschrift wird die Sortierreihenfolge umgekehrt. Nachdem eine Spalte auf diese Art sortiert wurde, wird der erste Eintrag aus der Liste, der diesen Begriff enthält, markiert, wenn Sie auf einen beliebigen Eintrag in der Liste klicken und anschließend einen Begriff aus der sortierten Spalte eingeben. Auf diese Art können Sie schnell zu Vorlagen mit einem bestimmen Namen/Ausdruck, Modus oder einer bestimmten Priorität springen.
- *Suchen:* Geben Sie in das Suchfeld (rechts unten) den Namen oder den XPath-Ausdruck ein, nach dem Sie suchen möchten. Die Suchergebnisse werden während der Eingabe angezeigt. Die folgenden Suchoptionen stehen in der Dropdown-Liste des Suchfelds (*Abbildung oben*) zur Verfügung: (i) ob der Name oder Ausdruck entweder mit dem Suchbegriff beginnt oder ihn enthält (erste Befehlsgruppe im Menü); die Option "beginnt mit" ist bei jedem Start von XMLSpy die Standardeinstellung; (ii) ob die Suchbegriffe als reduzierte Liste oder markiert angezeigt werden sollen (dritte Befehlsgruppe im Menü); Standardeinstellung bei jedem Start von XMLSpy ist die reduzierte Liste. Diese Auswahloptionen werden sofort angewendet und bleiben solange gültig, bis sie geändert oder XMLSpy geschlossen wird.
- *Neuladen:* Nachdem das Stylesheet geändert wurde, klicken Sie in der Symbolleiste des Fensters auf die Schaltfläche **Synchronisieren** , um die XSL-Übersicht zu aktualisieren.
- *Gehe zu Element:* Klicken Sie, wenn eine Vorlage oder Funktion im Fenster "XSL-Übersicht" ausgewählt ist, in der Symbolleiste des Fensters auf die Schaltfläche **Gehe zu Definition** , um die Vorlage oder Funktion im Dokument in der Design-Ansicht zu markieren. Alternativ dazu können Sie auf einen Eintrag auch doppelklicken um dorthin zu springen.
- *Aktionen für Named Templates:* Im Zusammenhang mit Named Templates können zwei Gruppen von Aktionen ausgeführt werden: (i) Aufrufe des Named Template (mit `xsl:call-template`) können am Cursor in das Stylesheet eingefügt werden; und (ii) ein Named Template kann als Ausgangspunkt für eine Transformationen definiert werden. Die Befehle für diese Aktionen werden über Symbolleistenschaltflächen ausgeführt und sind im Folgenden beschrieben.

Vorlagenmodus für die Transformation

Die Auswahlliste in der Symbolleiste, die den Titel *Modus für Transformationen festlegen*, hat, listet (i) alle Modi im Stylesheet sowie (ii) einen leeren Eintrag (mit dem der Standardmodus ausgewählt wird) aus. Bei XSLT 3.0 Stylesheets wird außerdem der (iii) Modus `#unnamed` aufgelistet. Durch Auswahl eines Modus aus der Dropdown-Liste wird dieser Modus für die Transformation verwendet. Der Modus `#unnamed` wird (für alle XSLT-Versionen) auf alle Vorlagen angewendet, die kein `mode` Attribut haben.

Bei XSLT 1.0- und XSLT2.0 Stylesheets ist der Standardmodus der Modus `#unnamed`. Wenn Sie daher den leeren Eintrag auswählen, wird der Standardmodus (also der Modus `#unnamed`, der somit auf alle Vorlagen ohne `mode` Attribut angewendet wird) ausgewählt.

In XSLT 3.0 Stylesheets kann das `xslt` Element der obersten Ebene ein `default-mode` Attribut haben, das den Standardmodus für die Transformation enthält. Wenn in der Auswahlliste *Modus für Transformationen*

festlegen der leere Eintrag (Standardmodus) ausgewählt ist, so wird der im `default-mode` Attribut definierte Modus als Transformationsmodus verwendet. Wenn in der Auswahlliste der Modus `#unnamed` ausgewählt ist, wird die Transformation auf alle Vorlagen mit einem unbenannten (unnamed) Modus, also auf Vorlagen ohne `mode`-Attribut angewendet.

Anmerkung: Eine Vorlage kann einen Moduswert `#all` erhalten, damit sie auf alle Modi angewendet werden kann.

Named Templates

Wenn ein Named Template ausgewählt ist, werden ein oder mehrere Befehle zu Named Templates in der Symbolleiste des Fensters aktiv (*Abbildung unten*).



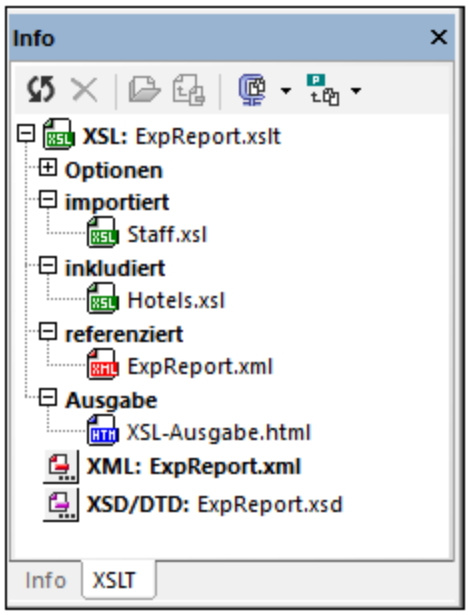
Die Befehle in der Symbolleiste (*Abbildung oben*) von links nach rechts:

- *xsl:call-template einfügen:* Dieser Befehl wird aktiv, wenn im Fenster "XSL-Übersicht" ein Named Template ausgewählt wird. Der Befehl fügt ein `xsl:call-template` Element am Cursoreinfügepunkt in das Stylesheet ein. Das `name` Attribut des `xsl:call-template` Elements, das ins Stylesheet eingefügt wird, erhält einen Wert, der dem des `name` Attributs der ausgewählten Vorlage entspricht. Auf diese Art wird mit `xsl:call-template` das ausgewählte Named Template ausgewählt.
- *xsl:call-template mit Param einfügen:* Dieser Befehl wird aktiv, wenn im Fenster "XSL-Übersicht" ein Named Template mit einem oder mehreren `xsl:param` Child-Elementen ausgewählt wird. Wie beim Befehl **xsl:call-template einfügen** fügt der Befehl ein `xsl:call-template` Element ein, in diesem Fall jedoch mit einem entsprechenden `xsl:with-param` Child-Element für jedes `xsl:param` Child-Element des ausgewählten Named Template. Die Namen des eingefügten `xsl:call-template` und seiner `xsl:with-param` Child-Elemente entspricht den Namen der ausgewählten Vorlage und ihrer `xsl:param` Child-Elemente.
- *Definiert die derzeit ausgewählte benannte Vorlage als Eintrittspunkt für die Transformation:* Wenn ein Named Template als Eintrittspunkt für eine Transformation definiert wird, beginnen in XMLSpy ausgeführte Transformationen an diesem Named Template. Im Fenster "XSL-Übersicht" wird ein solches Named Template fett geschrieben (siehe Abbildung zu Beginn dieses Abschnitts).
- *Löscht die benannte Vorlage als Eintrittspunkt für die Transformation:* Wird aktiv, wenn ein Named Template als Eintrittspunkt für Transformationen definiert wurde.
- *Springt zur derzeit ausgewählten benannten Vorlage als Eintrittspunkt für die Transformation:* Wird aktiv, wenn ein Named Template als Eintrittspunkt für Transformationen definiert wurde. Wenn sich der Fokus im Fenster "XSL-Übersicht" auf einem anderen Punkt als dem für Transformationen als Eintrittspunkt definierten Named Template befindet, wird das Named Template bei Klick auf diese Schaltfläche im Fenster "XSL-Übersicht" markiert, sodass Sie darauf schneller Zugriff haben.

7.3.2 Das Fenster "Info"

Auf dem Register "XSLT" des Info-Fensters, welches nur angezeigt wird, wenn es sich beim aktiven Dokument in XMLSpy um ein XSLT-Dokument handelt, werden alle importierten und inkludierten XSLT-Dateien, die mit dem aktiven XSLT-Dokument in Zusammenhang stehen, angezeigt. Sie können über dieses Register auch auswählen, welche XML-Datei anhand der XSLT-Datei transformiert werden soll, wenn eine Transformation gestartet wird, wenn das XSLT-Dokument das aktive Dokument ist.

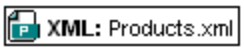
© 2016-2022 Altova GmbH

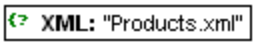


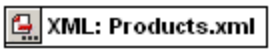
Auf dem Register "XSLT" des Info-Fensters werden die folgenden Dateien angezeigt:

- *XSLT-Dateien:* Alle importierten und inkludierten XSLT-Dateien werden aufgelistet (*siehe Abbildung oben*). Der Pfad der einzelnen Dateien wird in einem Popup-Fenster angezeigt, wenn der Maus-Cursor über die Datei platziert wird. Wenn Sie auf eine importierte oder inkludierte Datei doppelklicken oder sie auswählen und anschließend in der Symbolleiste des Info-Fensters auf die Schaltfläche **Öffnen** klicken, wird die Datei in einem neuen Fenster geöffnet. Mit der Symbolleisten-Schaltfläche **Gehe zur Include/Import-Stelle** wird die include/import-Deklaration im aktiven XSLT-Dokument markiert.

- *XML-Datei:* Dem aktiven XSLT-Stylesheet kann für Transformationen eine XML-Datei zugewiesen werden. Der Pfad der zugewiesenen XML-Datei wird in einem Popup-Fenster angezeigt, wenn der Maus-Cursor über die Datei platziert wird. Wenn eine XML-Datei definiert ist und der Menübefehl **XSL/XQuery | XSL-Transformation (F10)** ausgewählt wird, wird an der definierten XML-Datei mit Hilfe des aktiven XSLT-Dokuments als Stylesheet eine Transformation ausgeführt. Die XML-Datei kann durch Klicken auf das **XML**-Symbol und Suchen nach der Datei ausgewählt werden; die ausgewählte Datei wird fett angezeigt. Alternativ dazu kann die XML-Datei über das Dialogfeld "Projekteigenschaften" (*XML-Input-Datei bei XSL/XQuery/Update-Transformationen*) oder über eine Verarbeitungsanweisung im XSLT-Dokument in der Form: `<?altova_samplexml "Products.xml"?>` zugewiesen werden. In beiden Fällen wird die XML-Datei im Info-Fenster mit dem entsprechenden Symbol angezeigt:

   zugewiesen über das Dialogfeld "Projekteigenschaften"

   zugewiesen über eine Verarbeitungsanweisung im XSLT-Dokument

   zugewiesen durch Klicken auf das XML-Symbol und Suchen der gewünschten Datei; der Eintrag erscheint fett

  Falls mehr als eine der obigen Zuweisungen vorhanden sind, ist die Auswahlpriorität die folgende: (i) Projekt; (ii) Verarbeitungsanweisung; (iii) Suche durch den Benutzer. Die XML-Datei kann durch

Doppelklick oder durch Auswählen und Klicken auf die Symbolleisten-Schaltfläche **Öffnen** geöffnet werden.

- *XSD/DTD-Datei:* Wenn die ausgewählte XML-Datei eine Referenz auf ein Schema hat (XML-Schema oder DTD), so wird diese Schema-Datei im XSD/DTD-Eintrag angezeigt. Alternativ dazu kann die Schema-Datei wie bei der XML-Datei über das Dialogfeld "Projekteigenschaften" (*Validierung*) oder durch Klicken auf die Schaltfläche **XSD/DTD** und Suchen nach der gewünschten Schema-Datei ausgewählt werden. Wenn die Schema-Datei über das Dialogfeld "Projekteigenschaften" ausgewählt wird, wird neben dem Eintrag ein Projekt-Symbol angezeigt, andernfalls wird das anklickbare Symbol **XSD/DTD** mit dem Dateieintrag angezeigt - entweder in normaler Schrift (wenn das Schema von der XML-Datei referenziert wird) oder fett (das Schema wurde vom Benutzer über das Symbol **XSD/DTD** ausgewählt). Sollte die Schema-Datei über mehrere Methoden ausgewählt worden sein, so ist die Reihenfolge der Priorität die folgende: (i) Projekt; (ii) vom Benutzer ausgewählt; (iii) im XML-Dokument referenziert. Der Pfad der zugewiesenen XSD-Datei wird in einem Popup-Fenster angezeigt, wenn der Maus-Cursor über die Datei platziert wird. Die Schema-Datei kann durch Doppelklick oder durch Anklicken der Datei und Auswahl der Symbolleistenschaltfläche **Öffnen** geöffnet werden.

Anmerkung: Wenn eine XML- oder XSD/DTD-Datei über das Dialogfeld "Projekteigenschaften" ausgewählt wurde, müssen Sie zum Löschen dieser Auswahl wieder dieses Dialogfeld aufrufen und die Einstellung hier löschen. Wenn die Auswahl über die Schaltflächen **XML** oder **XSD/DTD** getroffen wurde, müssen Sie zum Löschen dieser Einstellung die Datei auswählen und in der Symbolleiste des Info-Fensters auf die Schaltfläche **Löschen** klicken.

Optionen

Intelligente XPath-Editierung: Wenn eine XML-Datei zugewiesen wurde, ist die Struktur des XML-Dokuments bekannt und die intelligente XPath-Editierung[499] schließt auch Elemente und Attribute ein. An den Stellen im XSLT-Dokument, an denen ein XPath-Ausdruck eingegeben werden kann, werden verfügbare Elemente und Attribute in einem Popup-Fenster angezeigt. Diese Option ist standardmäßig aktiv. Um die intelligente XPath-Editierung zu deaktivieren, deaktivieren Sie das Kontrollkästchen. Die Einstellung wird für jede XSLT-Datei separat gespeichert, wenn die Datei geschlossen wird und wieder verwendet, wenn die Datei erneut geöffnet wird.

Symbolleisten-Schaltflächen

Die Schaltflächen der Symbolleiste des Info-Fensters (*Abbildung unten*) sind von links nach rechts die folgenden:



- *Info neu laden:* Aktualisiert das Info-Fenster, damit die im XSLT-Dokument vorgenommenen Änderungen berücksichtigt werden.
- *XML/XSD-Zuweisung löschen:* Löscht die vom Benutzer über die Schaltflächen XML bzw. XSD/DTD vorgenommene Zuweisung. Wählen Sie die zu löschende Datei aus und klicken Sie auf diese Schaltfläche.
- *Dokument öffnen:* Öffnet das ausgewählte Dokument.
- *Gehe zur include/import-Stelle:* Wenn eine importierte oder inkludierte Datei ausgewählt ist, wird bei Klicken auf diese Schaltfläche die entsprechende import- oder include-Deklaration im XSLT-Dokument markiert.
- *Alle lokalen Dokumente zippen:* Zippt alle im Info-Fenster aufgelisteten Dokumente unter einem benutzerdefinierten Pfad. Alternativ dazu können auch nur die ausgewählten Dokumente gezippt

© 2016-2022 Altova GmbH

werden; wählen Sie dazu im Dropdown-Menü dieser Schaltfläche den Befehl **Ausgewählte lokale Dokumente zippen** aus.

- *Alle Dateien zu Projekt hinzufügen:* Fügt alle Dateien zum aktuellen Projekt hinzu. Alternativ dazu können auch nur die ausgewählten Dokumente hinzugefügt werden; wählen Sie dazu im Dropdown-Menü dieser Schaltfläche den Befehl **Ausgewählte Dateien zu Projekt hinzufügen** aus.

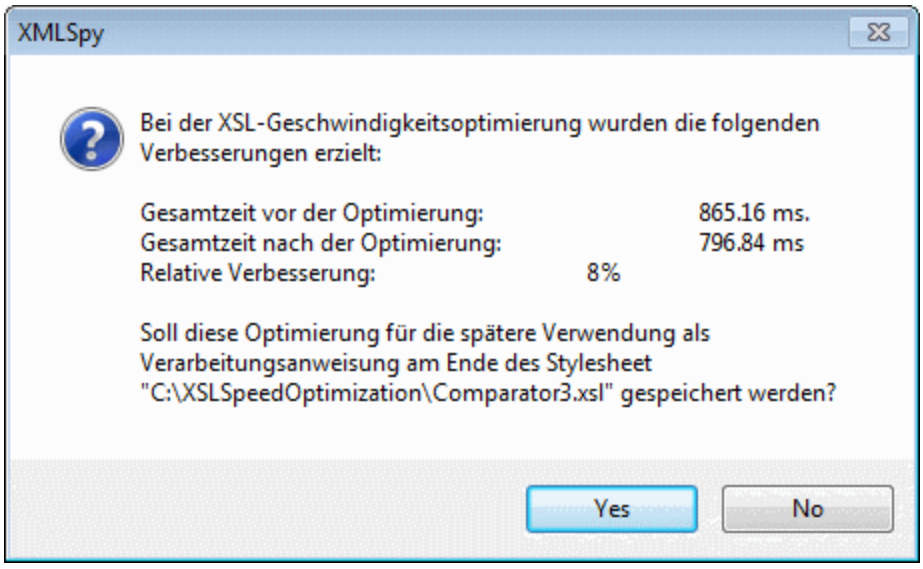
7.4 XSL-Geschwindigkeitsoptimierung

Bei der XSL-Geschwindigkeitsoptimierung (in diesem Abschnitt auch als Optimierung bezeichnet) können XSLT Stylesheets optimiert werden, damit Transformationen schneller durchgeführt werden können. Bei der Optimierung wird das XSLT Stylesheet an einem XML-Dokument ausgeführt. Dabei wird die Leistung analysiert. Anhand dieser Analyse wird eine Optimierungstrategie ermittelt, die im XSLT Stylesheet (in Form einer Verarbeitungsanweisung am Ende des Stylesheet) gespeichert werden kann. Mit Hilfe des optimierten Stylesheet können Transformationen in Zukunft schneller durchgeführt werden.

Optimieren eines XSLT Stylesheet

Um ein XSLT Stylesheet zu optimieren, benötigen Sie neben einem XSLT Stylesheet ein XML-Dokument, aus dem die Beispieldaten genommen werden. Das XML-Dokument muss groß genug sein, dass alle Teile des XSLT Stylesheet zur Anwendung kommen, damit das XSLT Stylesheet gründlich analysiert wird. Gehen Sie bei der Optimierung folgendermaßen vor:

1. Klicken Sie, während entweder das XSLT Stylesheet oder das XML-Dokument aktiv ist, auf den Menübefehl **XSL/XQuery | XSL-Geschwindigkeitsoptimierung** 1412 oder klicken Sie auf das Optimierungssymbol in der Hauptsymbolleiste.
2. Je nachdem, ob das XSLT oder das XML-Dokument aktiv ist, werden Sie aufgefordert, ein XML-Dokument bzw. ein XSLT Stylesheet auszuwählen. Nachdem Sie dies getan haben und auf **OK** geklickt haben, wird die Analyse gestartet. (Wenn das XSLT- bzw. XML-Dokument bereits eine Zuweisung hat, wird dieser Schritt übersprungen. Die Analyse wird sofort nach Aufruf des Befehls gestartet.)
3. Wenn die Optimierungsanalyse fehlgeschlagen ist, wird eine entsprechende Meldung angezeigt. (Die möglichen Ursachen für eine fehlgeschlagene Optimierungsanalyse 513 sind unten beschrieben.) Wenn die Analyse erfolgreich war, wird ein Dialogfeld mit den Ergebnissen der Analyse angezeigt (*Abbildung unten*).



Sie haben in diesem Dialogfeld die Möglichkeit, die Optimierungsanweisungen (in Form einer Verarbeitungsanweisung am Ende des Stylesheet) im XSLT Stylesheet zu speichern. Klicken Sie auf **Ja**, um die Optimierung zu speichern und auf **Nein**, um sie zu verwerfen. Sobald eine Optimierung gespeichert wird, werden alle zuvor gespeicherten Optimierungen überschrieben.

© 2016-2022 Altova GmbH

Mit Hilfe des optimierten Stylesheet können Sie Transformationen nun schneller durchführen.

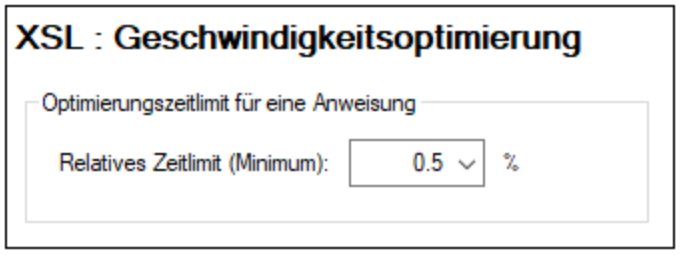
Ursachen für fehlgeschlagene Optimierungsanalysen

Wenn bei der XSL-Geschwindigkeitsoptimierung keine Lösung gefunden wird, kann dies einen oder mehrere der folgenden Gründe haben:

- Das XSLT Stylesheet ist bereits effizient und muss nicht weiter optimiert werden.
- Die bereitgestellte XML-Dokument war nicht groß genug, um eine Optimierung zu ermöglichen. Versuchen Sie es mit einem größeren Dokument.
- Das Limit/die Limits für die Optimierung ist/sind zu hoch. Ändern Sie die Limits im Dialogfeld "Optionen" im Abschnitt "XSL-Geschwindigkeitsoptimierung"[1636]. *Siehe unten*.
- Für diese spezifische XSLT-Struktur stehen keine Optimierungsmöglichkeiten zur Verfügung. Wenden Sie sich bitte an den Altova Support.

Einstellungen für die XSL-Geschwindigkeitsoptimierung

Die Einstellungen für die Optimierung werden im Dialogfeld "Optionen" im Abschnitt XSL-Geschwindigkeitsoptimierung[1636] (**Extras | Optionen,** *Abbildung unten*) vorgenommen.

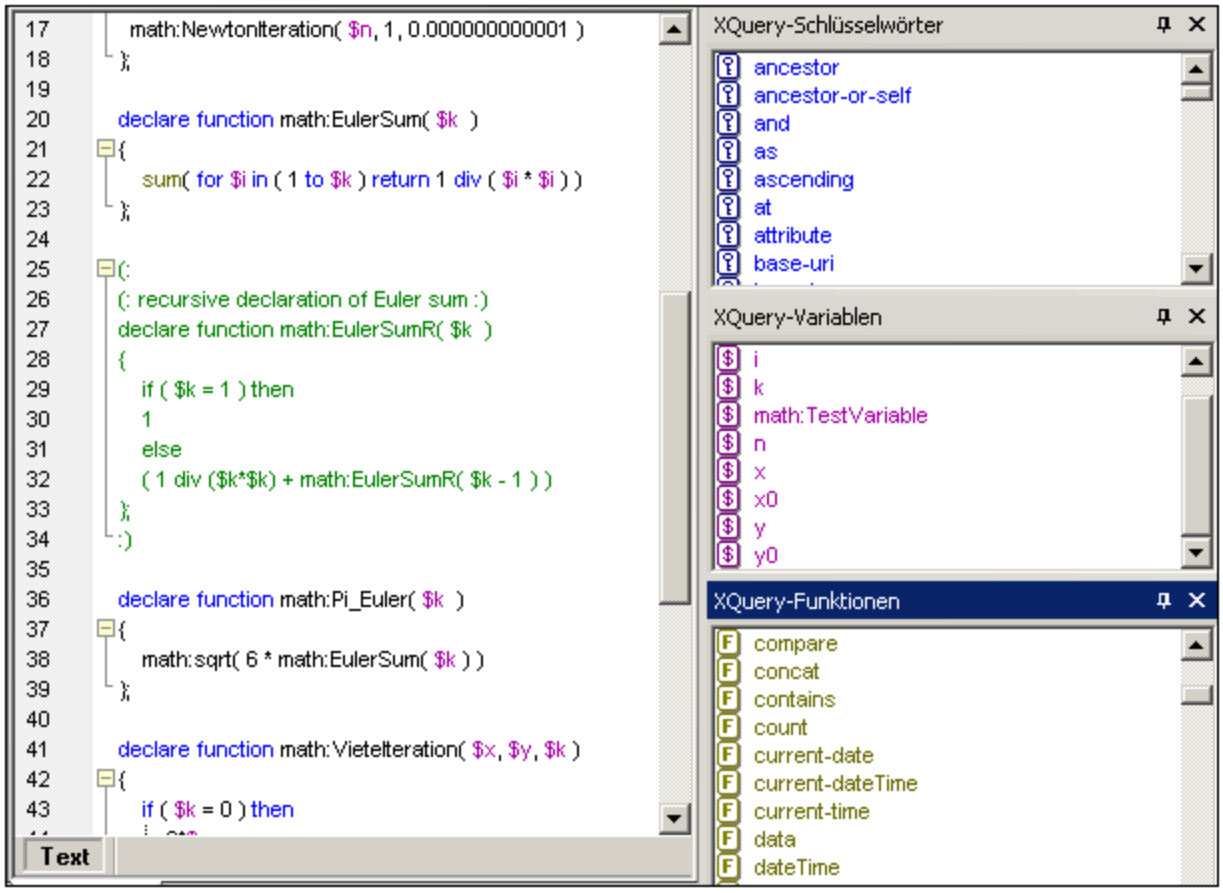


Sie können für die Optimierung ein Zeitlimit für einzelne XSLT-Anweisungen in einem XSLT Stylesheet definieren. Der Wertebereich reicht von 0,1% der Gesamttransformationsdauer bis 99%. Wenn die Ausführung einer Anweisung länger dauert, als im Zeitlimit definiert, so wird die Optimierungsanalyse aufgerufen. Andernfalls wird keine Analyse durchgeführt. Wenn die Optimierungsanalyse fehlschlägt, könnte der Grund dafür sein, dass das Zeitlimit in den Optimierungseinstellungen zu hoch ist. Setzen Sie es in diesem Fall herab.

8 XQuery

Altova Website: XQuery Editor

XQuery- und XQuery Update[528]-Dokumente können in der Textansicht bearbeitet werden. Zur einfachen Bearbietung bietet diese Ansicht (*siehe Abbildung*) Eingabehilfen, Syntaxfärbung und intelligente Bearbeitungshilfen. Zusätzlich dazu können Sie Ihr XQuery-Dokument validieren und (mit einer optionalen XML-Datei, wenn nötig) mit Hilfe des Altova XQuery-Prozessors ausführen .



Hinweis: XQuery- und XQuery Update-Dateien können nur in der Textansicht bearbeitet werden. Für XQuery-Dateien stehen keine anderen Ansichten zur Verfügung.

XQuery- und XQuery Update-Dateizuordnungen

XQuery- und XQuery Update-Dokumente werden in XMLSpy als zwei unterschiedliche Dokumenttypen erkannt. Normalerweise haben XQuery-Dokumente die Erweiterung `.xq`, während XQuery Update-Dokumente die Dateierweiterung `.xqu` haben. Im Dialogfeld "Optionen" im Abschnitt *Dateiarten* (**Extras | Optionen | Dateiart**[1606]) können Sie weitere Dateierweiterungen mit diesen Dateitypen verknüpfen und Dateitypzuordnungen auch ändern.

Die Dokumenttypzuordnung einer Dateierweiterung ist wichtig, da je nach Verknüpfung bei Aufruf des Befehls **XQuery/ Update-Ausführung** entweder eine XQuery-Ausführung oder ein XQuery Update[528] ausgeführt wird.

© 2016-2022 Altova GmbH

In diesem Abschnitt

Dieser Abschnitt ist in die folgenden Kapitel gegliedert:

- Bearbeiten von XQuery-Dokumenten 516
- XQuery-Auswertung 523
- XQuery-Validierung 524
- XQuery-Auswertung/Update 525
- XQuery Update Facility 528
- XQuery und XML-Datenbanken 535

Weitere damit in Zusammenhang stehende Features und Informationen:

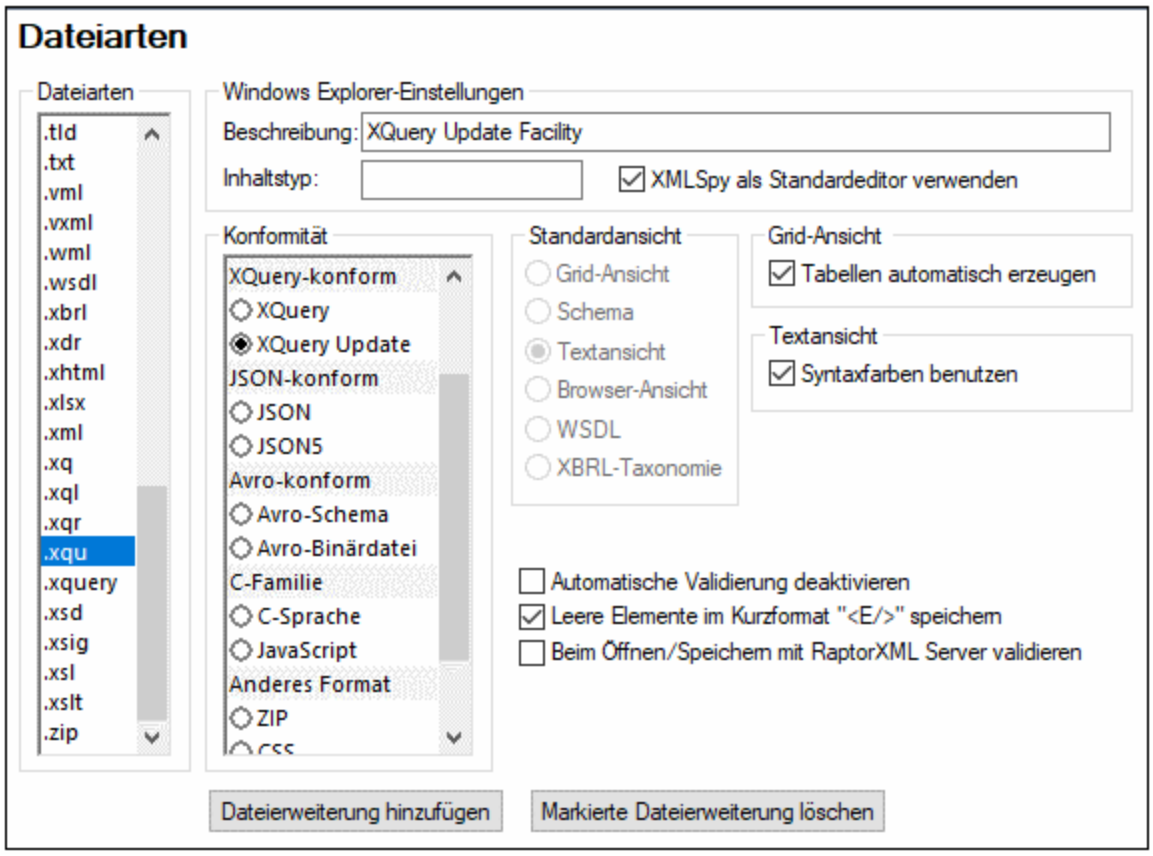
- XSLT/XQuery Debugger und Profiler 539
- Informationen zum XSLT- und XQuery-Prozessor 2137
- Ausgabefenster "XPath/XQuery" 124
- Extras | Optionen | Dateiarten 1606
- Extras | Optionen | XQuery 1636

RaptorXML für die Verarbeitung über die Befehlszeile und die Batch-Verarbeitung

Die Benutzeroberfläche von XMLSpy ermöglicht die Batch-Verarbeitung mittels der Projektfunktionalität. Wenn Sie jedoch mehr Flexibilität benötigen, sollten Sie das Altova-Produkt RaptorXML ausprobieren, das den neuesten XQuery-Prozessor von Altova enthält. RaptorXML ist ideal für die Ausführung von XQuery oder die Batch-Verarbeitung über die Befehlszeile geeignet.

8.1 Bearbeiten von XQuery-Dokumenten

XQuery- und XQuery Update-Dokumente werden in XMLSpy als zwei unterschiedliche Dokumenttypen erkannt. Der Dokumenttyp (XQuery oder XQuery Update) ist im Dialogfeld "Optionen" im Abschnitt *Dateiarten* (**Extras | Optionen | Dateiarten**[1606], *Abbildung unten*) einer Dateierweiterung zugewiesen. Wenn eine Datei vom Typ XQuery oder XQuery Update in XMLSpy geöffnet wird, stehen die XQuery-Bearbeitungsfunktionen der Textansicht für diese Datei zur Verfügung.



Aktuell in XMLSpy als XQuery und XQuery Update definierte Dateierweiterungen

XQuery	`.xq` `.xql` `.xqr` `.xquery`
XQuery Update	`.xqu`

Anmerkung: Die in diesem Abschnitt beschriebenen Bearbeitungsfunktionen sind für XQuery- und XQuery Update-Dokumente identisch.

XQuery-Ausführung/Update

Der Befehl **XSL/XQuery | XQuery/ Update-Ausführung** führt je nach dem Dateityp der ausgewählten XQuery-Datei entweder eine XQuery-Ausführung oder ein XQuery Update aus. Nähere Informationen dazu finden Sie im Abschnitt XQuery-Ausführung/Update[525].

© 2016-2022 Altova GmbH

8.1.1 XQuery-Dokumente

Ein XQuery- oder XQuery Update-Dokument wird automatisch im XQuery-Bearbeitungsmodus der Textansicht geöffnet, wenn es XQuery- oder XQuery Update-konform ist. Im Dialogfeld "Optionen" im Abschnitt *Dateiarten* (**Extras | Optionen | Dateiarten** 1606, *Abbildung unten*) wird eine Datei als bestimmter Dokumenttyp definiert.

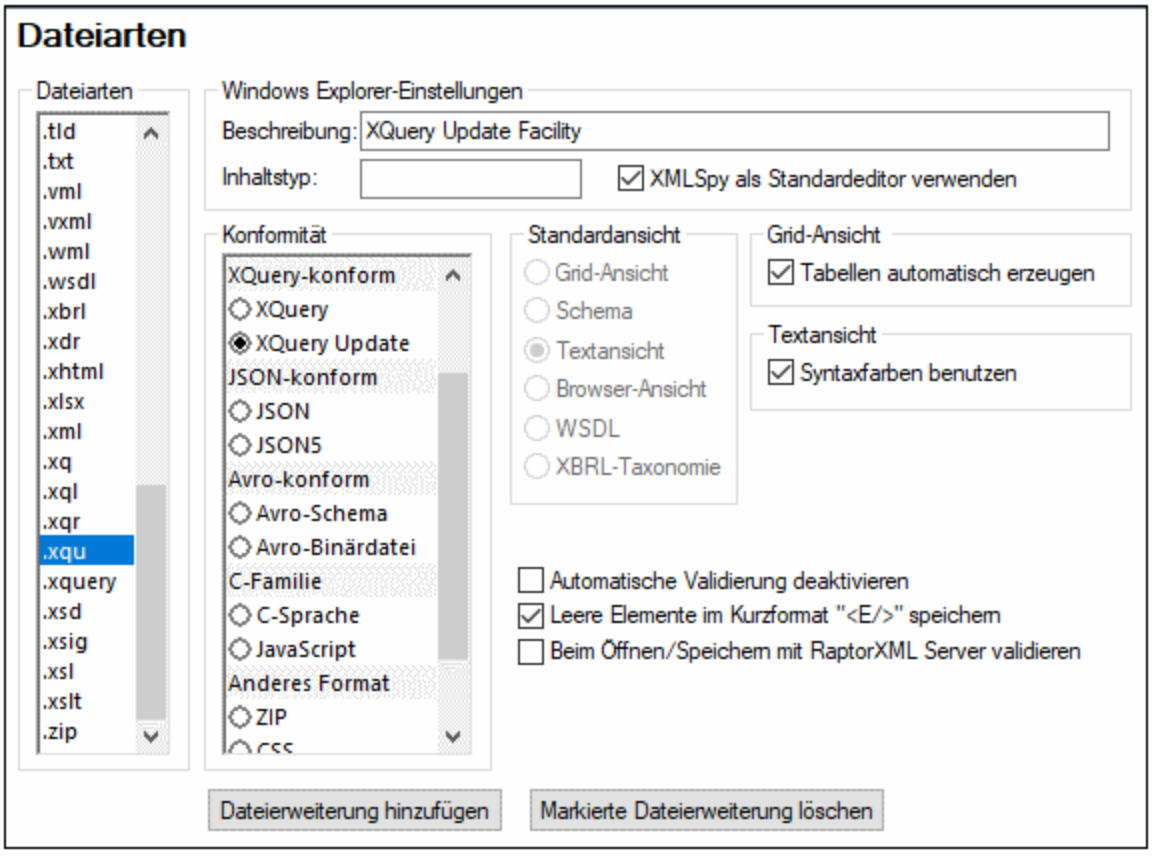
Aktuell in XMLSpy als XQuery und XQuery Update definierte Dateierweiterungen

XQuery	`.xq .xql .xqr .xquery`
XQuery Update	`.xqu`

Definieren weiterer Dateierweiterungen als XQuery-konform

So definieren Sie weitere Dateierweiterungen als XQuery-konform:

1. Wählen Sie **Extras | Optionen**. Daraufhin wird das Dialogfeld "Optionen" angezeigt (*siehe Abbildung unten*).



2. Wählen Sie das Register **Dateiarten**.
3. Klicken Sie auf "Neue Dateierweiterung hinzufügen", um der Liste der Dateiarten eine neue Dateierweiterung hinzuzufügen.

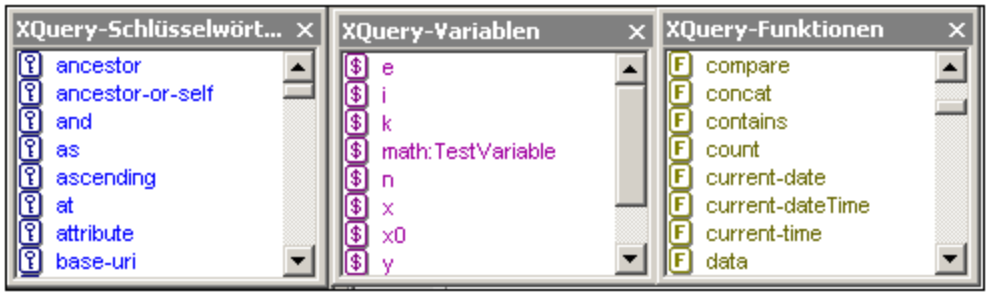
4. Aktivieren Sie unter *Konformität*, den Eintrag *XQuery-konform* und anschließend *XQuery* oder *XQuery Update*.

Außerdem sollten Sie in diesem Dialogfeld die folgenden Einstellungen vornehmen:

- *Beschreibung*: `XML Query Language` oder `XQuery Update Facility`
- *Inhaltstyp*: `text/xml`
- Wenn Sie XMLSpy als Standardeditor für XQuery-Dateien verwenden wollen, aktivieren Sie das Kontrollkästchen **XMLSpy als Standardeditor verwenden**.

8.1.2 XQuery Eingabehilfen

In der Textansicht stehen Ihnen Im XQuery-Modus drei Eingabehilfen zur Verfügung: XQuery-Schlüsselwörter (blau), XQuery-Variablen (violett) und XQuery-Funktionen (olivgrün).



Beachten Sie bitte die folgenden Punkte:

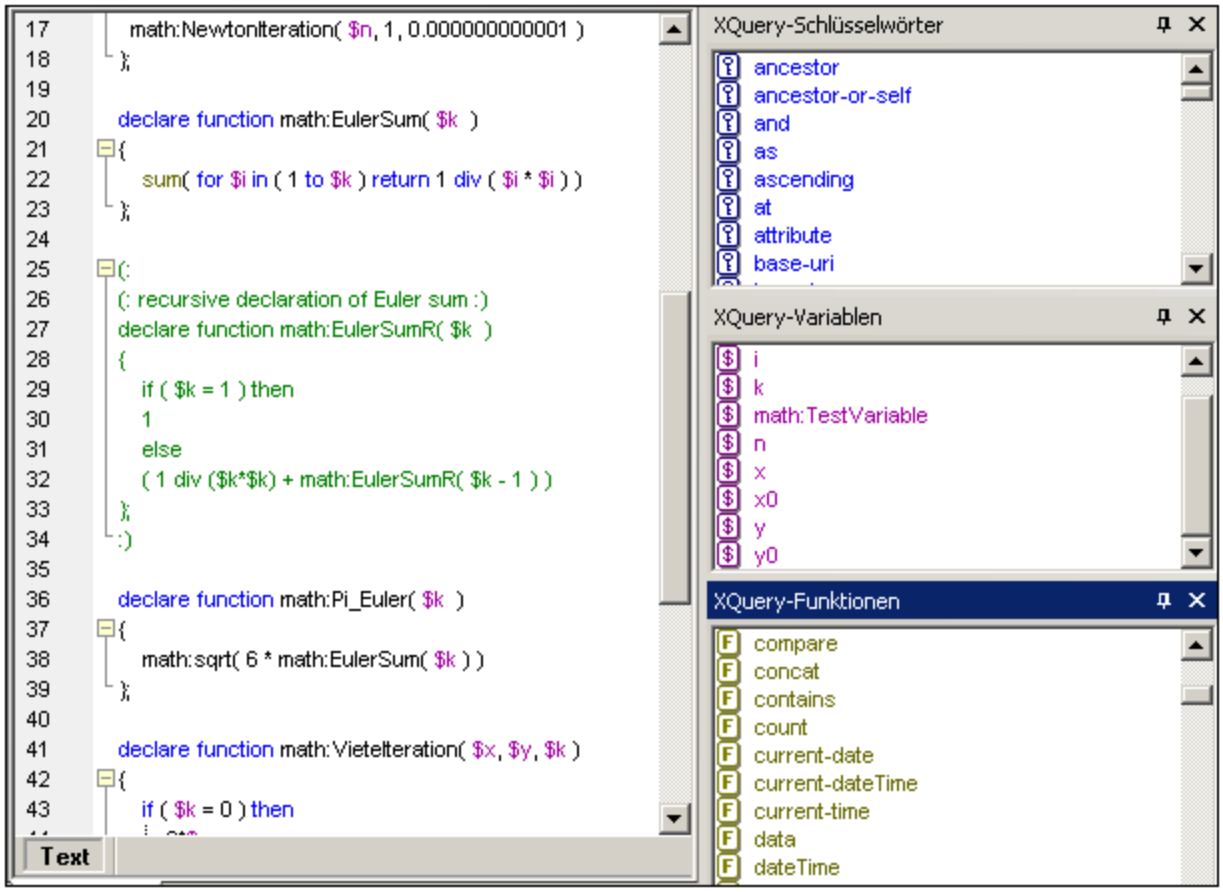
- Die Farbe der Einträge in den drei Eingabehilfen ist unterschiedlich und entspricht der im Text verwendeten Syntaxfärbung. Diese Farben können nicht geändert werden.
- Die aufgelisteten Schlüsselwörter und Funktionen sind diejenigen, die von den Altova XQuery-Prozessoren unterstützt werden.
- Die Variablen werden im XQuery-Dokument selbst definiert. Wenn Sie in der Textansicht ein `$`-Zeichen und ein weiteres Zeichen eingeben, erscheint das Zeichen in der Variablen-Eingabehilfe (es sei denn, es existiert eine Variable, die aus genau diesem Zeichen besteht). Sobald ein eingegebener Variablenname mit einem bereits existierenden Variablennamen übereinstimmt, verschwindet der neu eingegebene Variablenname aus der Eingabehilfe.
- Zum Navigieren in einer Eingabehilfe klicken Sie auf einen Eintrag in der Eingabehilfe und bewegen Sie sich mittels Bildlaufleiste, Mausrad oder der Bild auf- bzw. Bild ab-Taste durch die Liste.

Um einen der in den Eingabehilfen aufgelisteten Einträge in das Dokument einzufügen, setzen Sie den Cursor an die gewünschte Stelle und doppelklicken Sie auf den Eintrag.Beachten Sie, dass einige Zeichenfolgen sowohl ein Schlüsselwort als auch eine Funktion darstellen (`empty`, `unordered` und `except`). Der jeweilige Eintrag wird eingefügt, je nachdem, auf welchen Sie doppelklicken.

© 2016-2022 Altova GmbH

8.1.3 XQuery Syntaxfärbung

Ein XQuery-Dokument kann sowohl aus XQuery-Code als auch XML-Code bestehen. In diesem Abschnitt wird die Standard-Syntaxfärbung für den XQuery-Code beschrieben. Die Syntaxfärbung für XML-Code in einem XQuery-Dokument ist dieselbe wie die in normalen XML-Dokumenten verwendete. Alle Syntaxfärbungseinstellungen (sowohl für XQuery-Code als auch für XML-Code) werden im Dialogfeld "Optionen" (**Extras | Optionen**) im Abschnitt "Schriftarten und Farben" 1622 vorgenommen. Beachten Sie, dass XML-Elemente XQuery-Code enthalten können. Der XQuery-Code wird dabei in geschwungene Klammern **{}** gesetzt (*Beispiel siehe Abbildung*).



Im XQuery-Modus der Textansicht wird für XQuery-Code die folgende Standard-Syntaxfärbung verwendet:

- *(: Comments einschließlich 'Smiley'-Trennzeichen sind grau :)*
- XQuery-Schlüsselwörter sind blau: **Schlüsselwort**
- XQuery-Variablen, einschließlich des Dollar-Zeichens sind violett: **$start**
- XQuery-Funktionen, nicht aber die einschließenden Klammern sind grün: **function()**
- Strings sind orange: **"Prozedur"**
- Der gesamte andere Text, wie z.B. Path-Ausdrücke ist schwarz (wie der unterstrichene Text unterhalb). Dies sieht so aus:
  for $s in doc("report1.xml")//section[section.title = **"Procedure"**]
  return ($s//incision)[2]/instrument)

Sie können diese Standardfarben sowie andere Schriftarteinstellungen im Dialogfeld "Optionen" (**Extras | Optionen**) im Abschnitt "Schriftarten und Farben"[1622] ändern.

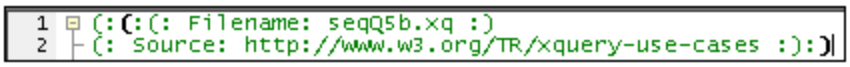
8.1.4 Intelligentes Editieren von XQuery

Im XQuery-Modus der Textansicht stehen Ihnen die folgenden intelligenten Editierfeatures zur Verfügung.

- Anzeigen von Klammerpaaren[520]
- Schlüsselwörter[520]
- Variablen[521]
- Funktionen[521]
- Visuelle Orientierungshilfen[522]

Anzeigen von Klammerpaaren

Die Funktion zum Anzeigen von Klammerpaaren markiert die öffnende und schließende Klammer eines Klammerpaars, sodass der Inhalt eines Klammerpaars klar ersichtlich ist. Dies ist v.a. bei verschachtelten Klammern, wie dies bei XQuery-Comments (*siehe Abbildung unten*) der Fall ist, hilfreich.



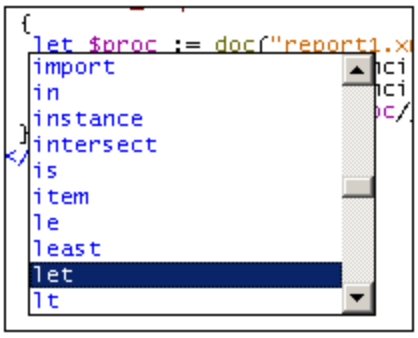
- Die Klammerpaaranzeige wird aktiviert, indem Sie den Cursor entweder unmittelbar vor oder unmittelbar hinter eine (öffnende oder schließende) Klammer setzen. Diese Klammer wird zusammen mit ihrem Gegenstück (schwarz und fett) hervorgehoben. Beachten Sie die Position des Cursors im obigen Screenshot.
- Die Klammerpaaranzeige ist für runde `()`, eckige `[]` und geschwungene Klammern `{}` aktiv. Eine Ausnahme bilden spitze Klammern **<>**, die für XML-Tags verwendet werden.

Anmerkung: Wenn Sie den Cursor innerhalb eines Start- oder End-Tags eines XML-Elements setzen, wird das Klammerpaars markiert. Wenn Sie **Strg+E** drücken, springt der Cursor zum anderen Teil der Klammer. Bei wiederholtem Drücken von **Strg+E** können Sie zwischen dem Start- und End-Tag hin- und herwechseln. Dies ist ein weiteres Hilfsmittel schnell durch Ihr Dokument navigieren zu können.

Schlüsselwörter

XQuery-Schlüsselwörter sind in Query-Ausdrücken verwendete Anweisungen und werden blau angezeigt. Wenn Sie den Cursor in oder unmittelbar vor oder hinter ein Schlüsselwort platzieren, wird das Schlüsselwort ausgewählt. Wenn Sie nun **Strg+Leertaste** drücken, wird in einem Popup-Menü eine komplette Liste aller Schlüsselwörter angezeigt. Sie können durch die Liste scrollen und auf ein Schlüsselwort doppelklicken, um das ausgewählte Schlüsselwort zu ersetzen.

© 2016-2022 Altova GmbH

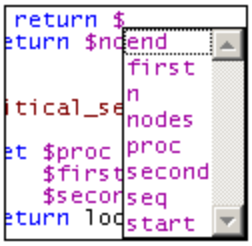


In der oben gezeigten Abbildung wurde der Cursor in das Schlüsselwort `let` gesetzt. Wenn Sie auf ein Schlüsselwort aus der Liste doppelklicken, wird das Schlüsselwort `let` durch dieses ersetzt.

Variablen

Variablennamen ist das **$**-Zeichen vorangestellt. Variablen werden violett angezeigt. Die Vorgangsweise bei dieser intelligenten Editierfunktion ist ähnlich wie bei der für Schlüsselwörter. Es gibt zwei Arten, um die Popup-Liste aller Variablen in einem Dokument aufzurufen:

- Geben Sie zuerst ein **$**-Zeichen ein und drücken Sie **Strg+Leertaste**
- Wählen Sie eine Variable aus und drücken Sie **Strg+Leertaste**. (Zum Auswählen einer Variable setzen Sie den Cursor unmittelbar hinter das **$** -Zeichen, in den Namen einer Variable oder unmittelbar hinter den Namen der Variable.)

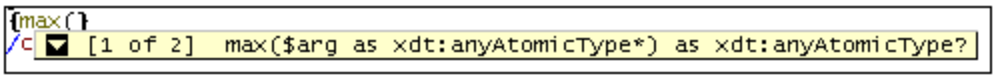


Um eine Variable (während der Eingabe) hinter dem **$**-Zeichen einzufügen oder eine ausgewählte Variable zu ersetzen, doppelklicken Sie im Popup-Menü auf die gewünschte Variable.

Funktionen

Genau wie bei Schlüsselwörtern und Variablen wird hier ein Popup-Menü mit vordefinierten Funktionen angezeigt, wenn Sie eine Funktion (angezeigt in Grün) auswählen und **Strg+Leerzeichen** drücken. (Zum Auswählen einer Funktion setzen Sie den Cursor in einen Funktionsnamen oder unmittelbar vor oder hinter den Namen. Der Cursor darf nicht zwischen die Klammern gesetzt werden, die auf den Funktionsnamen folgen). Wenn Sie im Popup-Menü auf einen Funktionsnamen doppelklicken, wird der ausgewählte Funktionsname durch die Funktion aus dem Popup-Menü ersetzt.

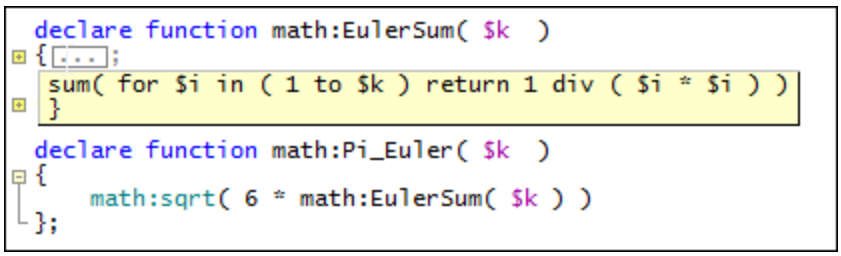
Um einen Tipp anzuzeigen, der die Bezeichnung einer Funktion enthält (*siehe Abbildung unten)*, setzen Sie den Cursor unmittelbar hinter die öffnende Klammer und drücken Sie **Strg+Leertaste.** Beachten Sie, dass die Signatur nur für XQuery-Standardfunktionen angezeigt werden kann.



Der nach unten weisende Pfeil bedeutet, dass es mehrere Funktionen mit demselben Namen gibt, dass diese jedoch unterschiedliche Argumente oder Rückgagetypen haben. Klicken Sie auf die Pfeilspitze um die Bezeichnung der nächsten Funktion anzuzeigen (falls vorhanden); wenn Sie mehrmals hintereinander darauf klicken, werden der Reihe nach alle Funktionen dieses Namens angezeigt. Alternativ dazu können Sie auch durch Drücken von **Strg+Umschalt+Nach unten** oder **Strg+Umschalt+nach oben** durch die Funktionen wechseln.

Visuelle Orientierungshilfen

Die Textein- und -ausklappfunktion ist bei geschwungenen XQuery-Klammern, bei XQuery Comments, XML-Elementen und XML-Comments aktiv. Mit Hilfe dieser Funktion können Sie diese Nodes ein- und ausklappen. Solche Nodes sind in der Klappleiste durch ein +/--Zeichen markiert (*siehe Abbildung unten*). Die Klappleiste kann über das Dialogfeld "Einstellungen für Textansicht" 1506 ein- und ausgeblendet werden. Ein eingeklappter Node wird durch Auslassungszeichen gekennzeichnet (*siehe Abbildung unten*). Wenn Sie den Mauszeiger über die Auslassungspunkte platzieren, wird der Inhalt des eingeklappten Node in einem Popup-Fenster angezeigt (*in der Abbildung unten blau markiert*). Falls der Inhalt für das Popup-Fenster zu groß ist, so wird dies durch Auslassungspunkte am unteren Rand des Popup-Fenster angezeigt.



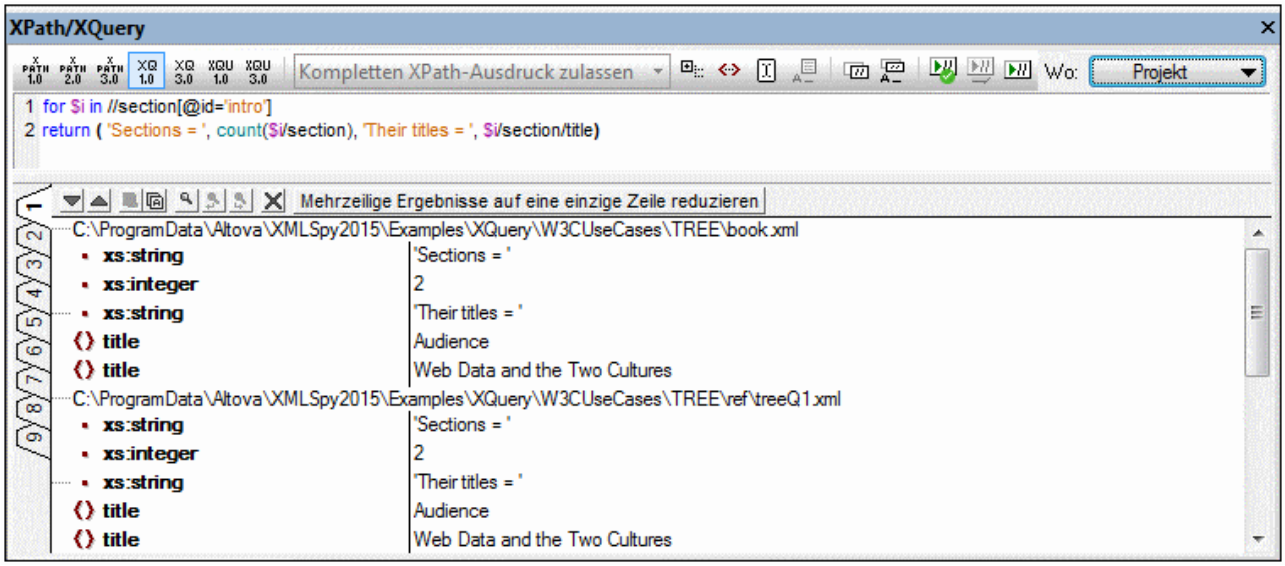
Mit der Schaltfläche **Alle ein-/ausklappen** in der Symbolleiste "Text" werden **alle** Nodes in ihrer erweiterten Form angezeigt bzw. auf das Dokumentelement der obersten Ebene reduziert.
Wenn Sie auf das +/--Symbol eines Node klicken, stehen die folgenden Optionen zur Verfügung:

Klick [-]	Klappt den Node ein.
Klick [+]	Zeigt den Node ausgeklappt an, sodass seine untergeordneten Nodes, je nachdem, in welchem Zustand sie sich vor dem Einklappen befunden haben, im erweiterten bzw. reduzierten Zustand zu sehen sind.
Umschalt+Klick [-]	Reduziert alle untergeordneten Nodes, belässt jedoch den angeklickten Node in seiner erweiterten Form.
Strg+Klick [+]	Erweitert den angeklickten Node sowie alle seine untergeordneten Nodes.

© 2016-2022 Altova GmbH

8.2 XQuery-Auswertung

XQuery-Ausdrücke können im XPath/XQuery-Ausgabefenster (*Abbildung unten*) anhand eines oder mehrerer Dokumente ausgewertet werden.



Gehen Sie dazu folgendermaßen vor:

1. Geben Sie im oberen Bereich des Fensters den XQuery-Ausdruck ein.
2. Wählen Sie in der Auswahlliste *Wo* (*siehe Abbildung oben*) aus, wo sich das gewünschte Dokument befindet. Die Optionen sind: (i) Aktuelle Datei; (ii) Offene Dateien; (iii) Projekt, (iv) Ordner.
3. Klicken Sie auf **XPath/XQuery-Ausdruck auswerten (F5)**. Daraufhin wird der Ausdruck anhand der XML-Datei(en) ausgewertet. Wenn der angegebene Pfad (*Wo*) mehr als eine XML-Datei enthält, werden alle XML-Dateien nach Datenstrukturen oder Inhalt, der mit dem Ausdruck übereinstimmt, durchsucht. Die Ergebnisse aller verfügbaren Übereinstimmungen werden im unteren Bereich angezeigt.

In der Abbildung oben wird eine Abfrage nach einem Abschnittselement mit dem Attribut `@id='intro'` durchgeführt. Das Ergebnis der Abfrage ist die Anzahl der Unterabschnitte dieses `intro` Abschnitts und ihrer Überschriften.

Nähere Informationen dazu finden Sie auch unter Ausgabefenster "XPath/XQuery" 124 und Vorschau auf und Anwendung von XQuery Updates 529.

8.3 XQuery-Validierung

Um ein XQuery- oder XQuery Update-Dokument zu validieren, gehen Sie vor wie folgt:

1. Machen Sie das XQuery-Dokument zum aktiven Dokument.
2. Wählen Sie **XML | Validieren** oder drücken Sie **F8** oder klicken Sie in der Symbolleiste auf das Symbol **Validieren**. .

 *Symbolleisten-Schaltfläche "Validieren"*

Das Dokument wird daraufhin auf die korrekte XQuery-Syntax überprüft.

© 2016-2022 Altova GmbH

8.4 XQuery/Update-Ausführung

Ein XQuery- oder XQuery Update-Dokument kann auf die folgenden Arten ausgeführt werden:

- wenn das XQuery- oder Update-Dokument aktiv ist.
- wenn ein XML-Dokument aktiv ist.

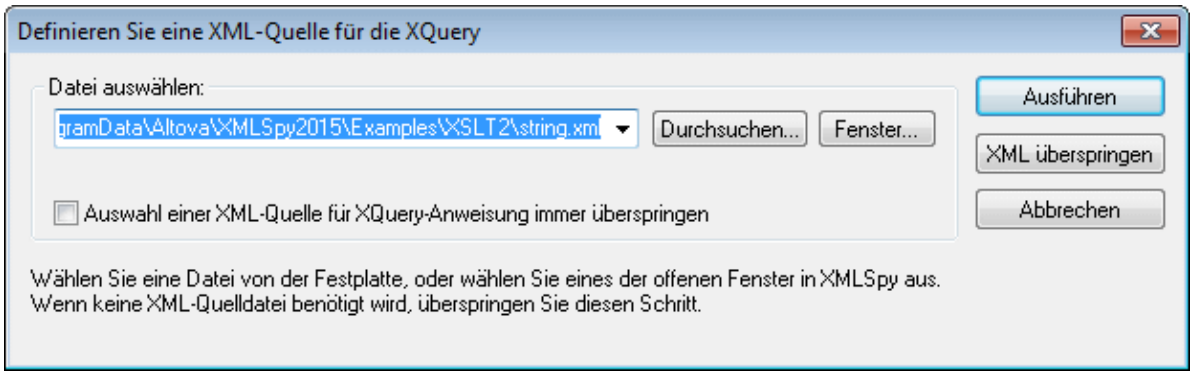
Anmerkung: Ob es sich bei einem Dokument um ein XQuery-Dokument oder ein XQuery Update-Dokument handelt, hängt von der Dateierweiterung des Dokuments ab. XMLSpy erkennt die Dateitypzuordnungen anhand der Definitionen im Abschnitt "Dateiarten" im Dialogfeld "Optionen". (**Extras | Optionen | Dateiarten**[1606]).

Anmerkung: Sie können für XQuery Update auch Update-Ausdrücke im XPath/XQuery-Ausgabefenster eingeben und eine Vorschau der Updates anzeigen. Wenn die Updates zulässig sind, können Sie die Updates anwenden und die aktualisierte Datei anschließend speichern. Nähere Informationen dazu finden Sie unter XQuery Update Facility[528] und Vorschau auf und Anwendung von Updates[529].

Ausführung mit aktivem XQuery- oder XQuery Update-Dokument

Um ein geöffnetes, aktives XQuery- oder XQuery Update-Dokument auszuführen, gehen Sie vor wie folgt:

1. Machen Sie das XQuery- oder XQuery Update-Dokument zum aktiven Dokument.
2. Wählen Sie **XSL/XQuery | XQuery/Update-Ausführung** oder klicken Sie in der Symbolleiste auf das Symbol für den Befehl. Daraufhin erscheint das Dialogfeld "Definieren Sie eine XML-Quelle für die XQuery" (*Abbildung unten*).



3. Navigieren Sie entweder zu einer XML-Datei und führen Sie diese aus oder überspringen Sie die Auswahl einer XML-Quelldatei.

 *Symbolleisten-Schaltfläche "XQuery/ Update-Ausführung"*

Normalerweise ist ein XQuery-Dokument **nicht** mit einem einzigen XML-Dokument verknüpft. Dies liegt daran, dass XQuery-Ausdrücke mit der Funktion `doc()` beliebig viele XML-Dokumente auswählen können. Bevor Sie einzelne XQuery-Dokumente in XMLSpy ausführen, können Sie allerdings ein XML-Quelldokument für die Ausführung auswählen. In solchen Fällen ist der Dokument-Node der ausgewählten XML-Quelldatei der Ausgangskontext auf der Root-Ebene des XQuery-Dokuments. Pfade, die mit einem Schrägstrich beginnen, werden mit diesem Dokument-Node als Kontext-Datenelement aufgelöst.

Anmerkung: Der Befehl **XQuery/ Update-Ausführung** steht auch im Kontextmenü von Einträgen im Projektfenster[118] zur Verfügung.

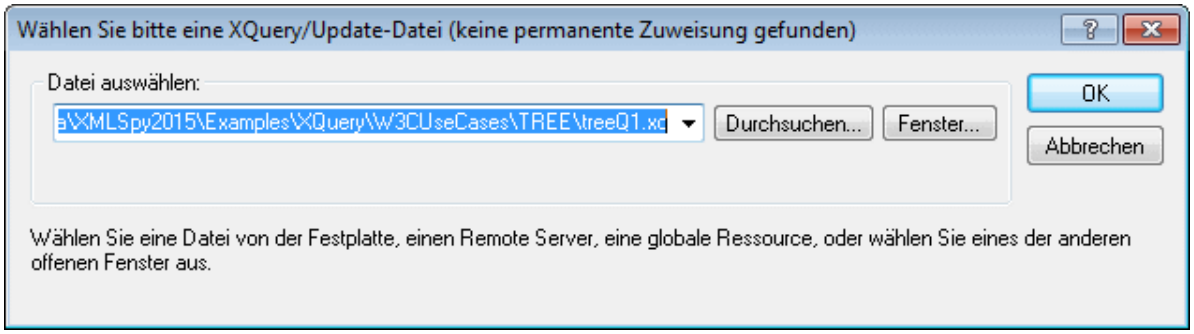
Ergebnis der Ausführung / des Update

- *XQuery-Ausführung:* Das Ergebnisdokument wird als temporäre Datei generiert, die im gewünschten Dateiformat und mit der gewünschten Erweiterung unter jedem Pfad gespeichert werden kann.
- *XQuery Update:* Das Update wird in einer Datei gespeichert oder die aktualisierte Datei wird geöffnet, um eine Vorschau davon anzuzeigen. Anschließend können Sie das Dokument entweder speichern oder schließen, ohne es zu speichern. Sie können festlegen, welche der beiden Aktionen durchgeführt werden soll. Diese Einstellung erfolgt im Dialogfeld "Optionen" im Abschnitt *XQuery* (**Extras | Optionen |XQuery**[1606]).

Ausführung bei aktiven XML-Dokument

Um ein XQuery- oder XQuery Update-Dokument an einem *aktiven XML-Dokument* auszuführen, gehen Sie folgendermaßen vor:

1. Machen Sie das XML-Dokument zum aktiven Dokument.
2. Wählen Sie den Befehl **XSL/XQuery | XQuery/ Update-Ausführung** oder klicken Sie in der Symbolleiste auf das Symbol für den Befehl. Daraufhin erscheint das Dialogfeld "Wählen Sie eine XQuery/Update-Datei" (*Abbildung unten*).



3. Navigieren Sie zur XQuery- oder XQuery Update-Datei und klicken Sie auf **OK**.

Symbolleisten-Schaltfläche "XQuery/ Update-Ausführung"

Ergebnis der Ausführung / des Update

- *XQuery-Ausführung:* Das Ergebnisdokument wird als temporäre Datei generiert, die im gewünschten Dateiformat und mit der gewünschten Erweiterung unter jedem Pfad gespeichert werden kann.
- *XQuery Update:* Das Update wird in einer Datei gespeichert oder die aktualisierte Datei wird geöffnet, um eine Vorschau davon anzuzeigen. Anschließend können Sie das Dokument entweder speichern oder schließen, ohne es zu speichern. Sie können festlegen, welche der beiden Aktionen durchgeführt werden soll. Diese Einstellung erfolgt im Dialogfeld "Optionen" im Abschnitt *XQuery* (**Extras | Optionen |XQuery**[1606]).

Rückwärts-Mapping

Wenn die Rückwärts-Mapping[1418]-Funktion aktiviert ist, werden XQuery-Ausführungen durchgeführt, sodass das Ergebnisdokument auf die XQuery+XML-Ausgangsdokumente rückgemappt werden kann. Wenn Sie auf einen Node im Ergebnisdokument klicken, werden die **XQuery-Anweisung** *und* die **XML-Quelldaten,** anhand welcher der jeweilige Ergebnis-Node generiert wurde, markiert. Wenn Sie außerdem auf eine XQuery-

© 2016-2022 Altova GmbH

Anweisung oder einen XML-Daten-Node klicken, werden die entsprechenden Nodes in den anderen beiden Dokumenten markiert. Nähere Informationen finden Sie unter dem Befehl **XSL/XQuery | Rückwärts-Mapping aktivieren**[1418].

XQuery-Variablen

Wenn Sie die Altova XQuery-Prozessoren verwenden, können Variablen über ein praktisches Dialogfeld gespeichert werden. Alle gespeicherten Variablen werden, jedes Mal, wenn Sie mit XMLSpy ein XQuery-Dokument ausführen, an das XQuery-Dokument übergeben. Nähere Informationen dazu finden Sie unter der Beschreibung zum Befehl XSLT-Parameter / XQuery-Variable[1414].

Altova XQuery-Prozessoren

Nähere Informationen darüber, wie die Altova XQuery-Prozessoren implementiert sind und XQuery-Dateien verarbeiten, finden Sie unter Implementierung des XQuery-Prozessors[2137].

8.5 XQuery Update Facility

XQuery Update Facility ist eine Erweiterung der XQuery-Sprache, mit Hilfe derer Teile des XML-Dokuments geändert werden können. Bei einer normalen XQuery-Ausführung wird das gesamte Dokument neu generiert und muss wieder unter demselben Pfad gespeichert werden. Dies ist nicht immer effizient, wenn nur kleine Teile des Dokuments geändert werden müssen. Mit Hilfe von Update Facility werden nur die Teile des Dokuments, die geändert werden müssen, aktualisiert.

XQuery Update Facility ist in den folgenden Spezifikationen als Erweiterung zu XQuery 1.0 und XQuery 3.1 beschrieben:

- XQuery Update Facility 1.0 (W3C Recommendation vom 17. März 2011)
- XQuery Update Facility 3.0 (W3C Working Draft vom 19. Februar 2015)

XQuery Update Facility in XMLSpy

Im Folgenden wird erklärt, wie XQuery Update in XMLSpy funktioniert:

- Eine Aktualisierung wird durch einen Update-Ausdruck durchgeführt. So kann ein Update-Ausdruck z.B. definieren, dass ein Node in einem XML-Dokument umbenannt wird:
```
rename node /documents/doc-01 as "document-01"
```
- In der Praxis werden in ein einziges Dokument, das XQuery Update-Dokument, mehrere Update-Ausdrücke eingegeben.
- Bei der Ausführung der einzelnen Update-Ausdrücke am zu aktualisierenden Dokument werden die Ergebnisse nicht sofort angewendet, sondern zu einer *Pending Updates List (PUL) (Liste der bevorstehenden Aktualisierungen)* hinzugefügt. Als Ergebnis enthält die PUL die Ergebnisse aller Update-Ausdrücke. Anschließend werden alle Aktualisierungen aus der PUL auf einmal angewendet.
- In XMLSpy können die PUL-Aktualisierungen auf zwei verschiedene Arten angewendet werden:
  (i) Nach Überprüfung der Vorschau durch den Benutzer über die Benutzeroberfläche. Der Vorteil ist, dass die Aktualisierung abgebrochen werden kann, wenn in der Vorschau unerwünschte Ergebnisse angezeigt werden. Die Vorschau steht bei Ausführung des Befehls XQuery/Update-Ausführung[525] oder bei Auswertung von XQuery Update-Ausdrücken im XPath/XQuery-Ausgabefenster[529] zur Verfügung. Eine Anleitung zum Definieren der Vorschauoption finden Sie in den entsprechenden Abschnitten der Dokumentation.
  (ii) Direkt und ohne Interaktion durch den Benutzer. Der Vorteil ist, dass die Aktualisierung ausgeführt wird, ohne dass der Benutzer etwas tun muss. Die direkte Anwendung von Aktualisierungen (ohne Anzeige einer Vorschau) steht bei Ausführung des Befehls XQuery/Update-Ausführung[525] oder bei Auswertung von XQuery Update-Ausdrücken im XPath/XQuery-Ausgabefenster[529] zur Verfügung. Eine Anleitung zum Definieren der direkten Aktualisierung finden Sie in den entsprechenden Abschnitten der Dokumentation.

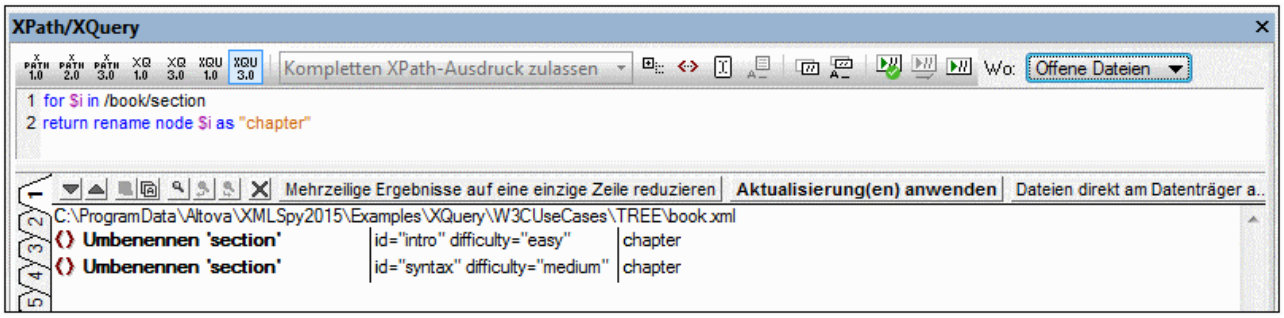
XMLSpy bietet eine leistungsstarke XQuery Update-Vorschaufunktion, mit der Sie eine Vorschau der Auswirkungen von Update-Ausdrücken am aktiven XML-Dokument anzeigen können, bevor Sie die Updates anwenden. Diese Funktion ist im Abschnitt Vorschau auf und Anwendung von Updates[529] beschrieben.

© 2016-2022 Altova GmbH

8.5.1 Vorschau auf und Anwendung von Updates

Wenn Sie ein XML-Dokument mit XQuery Update ändern möchten, können Sie eine Vorschau der Aktualisierungen anzeigen, bevor Sie diese auf das XML-Dokument anwenden und das geänderte Dokument speichern.

Im XPath/XQuery-Ausgabefenster (*Abbildung unten*) können Sie einen oder mehrere Update-Ausdrücke eingeben und anschließend in der **Pending Update List (PUL)** (Liste der bevorstehenden Aktualisierungen), die im unteren Bereich angezeigt wird (*siehe Abbildung unten*), eine Vorschau der Aktualisierungen anzeigen. Wenn die PUL Ihren Erwartungen entspricht, können Sie die Aktualisierungen auf das Dokument anwenden und das geänderte Dokument anschließend speichern. Wenn Sie die Änderungen in der PUL nicht übernehmen möchten, können Sie die Änderungen entweder nicht anwenden oder die Datei nicht speichern.



Um eine PUL für eine aktive XML-Datei zu erstellen, gehen Sie folgendermaßen vor:

1. Wählen Sie in der Symbolleiste des XPath/XQuery-Ausgabefensters (*Abbildung oben*) eines der Symbole **XQU 1.0** oder **XQU 3.0** aus (diese Symbole stehen für XQuery Update 1.0 bzw. XQuery Update 3.0).
2. Geben Sie im oberen Bereich des Fensters einen oder mehrere Update-Ausdrücke ein. Eine Beschreibung von Update-Ausdrücken und ihrer Syntax finden Sie unter Update-Operationen und -Syntax 531.
3. Wählen Sie in der Symbolleiste in der Auswahlliste "Wo" aus, welche Dateien nach Aktualisierungen durchsucht werden sollen:
   Offene Dateien: Alle aktuell in XMLSpy geöffnete Dateien werden durchsucht.
   Aktuelle Datei: Nur die derzeit aktive Datei wird durchsucht. Wenn die Auswahl *Aktuelle Datei* aktiv ist, so ist die Symbolleisten-Schaltfläche **XPath/XQuery-Ausdruck bei der Eingabe auswerten** aktiv.
   Projekt: Das derzeit aktive Projekt wird durchsucht.
   Ordner: Sie können einen Ordner auswählen, der durchsucht werden soll.
4. Um den/die Update-Ausdruck/Ausdrücke auszuführen und die PUL anzuzeigen, klicken Sie auf die Symbolleisten-Schaltfläche **XPath/XQuery-Ausdruck auswerten**.

Symbolleiste des XPath/XQuery-Ausgabefensters

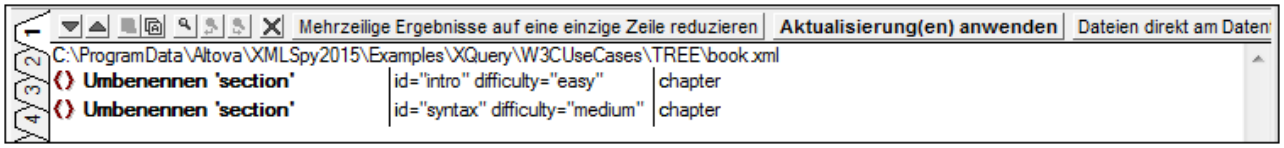
In der Symbolleiste des XPath/XQuery-Ausgabefensters stehen von links nach rechts die folgenden Schaltflächen zur Verfügung (*in der Abbildung unten rot umrahmt*):



- Mit der Ein/Aus-Schaltfläche *Überschrift in der Ausgabe anzeigen* definieren Sie, ob der Dateiname der aktualisierten Dateien angezeigt werden soll oder nicht.
- Mit der Ein/Aus-Schaltfläche *Gesamtes Ergebnis anzeigen* geben Sie an, ob der gesamte NAttribute des Node angezeigt werden sollen.
- Wenn die Schaltfläche *Aktuelle Auswahl in Dokument als Ausgangspunkt für XPath/XQuery festlegen* aktiviert ist, wird die Cursorposition in der aktiven Datei als Ausgangspunkt für relative XPath-Ausdrücke verwendet. Der Ausdruck wird relativ zu diesem Node ausgewertet.
- Anstatt den Locator-Pfad-Ausdruck eines Node manuell einzugeben, können Sie dies von XMLSpy erledigen lassen. Gehen Sie dazu folgendermaßen vor: (i) Platzieren Sie den Cursor an die Stelle im XPath-Ausdruck, an der der Locator-Pfad eingegeben werden soll; (ii) Setzen Sie den Cursor in den Start-Tag des Node, der adressiert werden soll; (iii) Klicken Sie auf das Symbol *Kopiert den XPath von der aktuellen Auswahl in das Bearbeiten-Feld* , um den Locator-Pfad in den Ausdruck einzugeben. Der Locator-Pfad wird als absoluter Pfad, der am Root-Node des Dokuments beginnt, eingegeben.
- Mit der Ein/Aus-Schaltfläche *XML-Datei validieren* aktivieren/deaktivieren Sie die XML-Datei-Validierung.
- Der Befehl *XPath/XQuery-Ausdruck bei der Eingabe auswerten* ist aktiv, wenn die Parse-Option *Aktuelle Datei* ist. Wenn der Befehl aktiv ist, wird die PUL während der Eingabe des Update-Ausdrucks generiert.
- Der Befehl *XPath/XQuery-Ausdruck auswerten* wertet den/die Update-Ausdruck/Ausdrücke aus und generiert die PUL.
- Eine Beschreibung der Auswahlliste *Wo* finden Sie weiter oben.

Der PUL (Pending Update List)-Bereich (Liste der bevorstehenden Aktualisierungen)

Im PUL-Bereich sehen Sie alle Aktualisierungen, die durchgeführt werden sollen. Wenn in der Symbolleiste des Fensters die Option "Überschrift in der Ausgabe anzeigen" aktiv ist, so wird der Pfad der Zieldateien angezeigt. Der PUL-Bereich ist in drei vertikale Abschnitte gegliedert (*siehe Abbildung unten*): (i) die auszuführende Update-Aktion; (ii) den Inhalt des zu aktualisierenden Ziel-Node; (iii) das Ergebnis der Aktualisierungsaktion.



Im PUL-Bereich stehen die folgenden Befehle zur Verfügung:

- Mit den Schaltflächen *Weiter* und *Zurück* wird die nächste bzw. vorhergehende Meldung ausgewählt.
- Die *Zeile(n) kopieren*-Befehle kopieren die ausgewählte Zeile bzw. alle Zeilen in die Zwischenablage.
- Der Befehl *Suchen* dient zum Suchen von Text im PUL-Bereich.
- Der Befehl *Löschen* löscht alle Zeilen im PUL-Bereich.
- Der Befehl *Mehrzeilige Ergebnisse auf eine Zeile reduzieren* reduziert mehrzeilige Einträge in der Aktualisierungsliste auf eine einzige Zeile.
- Der Befehl *Aktualisierung(en) anwenden* wendet die bevorstehenden Aktualisierungen auf die Zieldateien an. Die Aktualisierungen können bei der Aktualisierung in einer Datei gespeichert werden

 © 2016-2022 Altova GmbH

oder die aktualisierte Datei kann angezeigt werden (und anschließend manuell gespeichert oder nicht gespeichert werden). Siehe nächste Option.

- Über die Auswahlliste *Dateien direkt aktualisieren* können Sie auswählen, (i) ob Dateien direkt am Datenträger aktualisiert werden sollen oder (ii) ob die aktualisierten Dateien geöffnet und aktiv gemacht werden sollen. Bei Auswahl der letzteren Option werden nicht aktive Zieldateien zu aktiven gemacht. Sie können anschließend entscheiden, ob Sie das geänderte Dokument speichern möchten oder nicht.

Anmerkung: Wenn eine oder mehr Dateien direkt am Datenträger aktualisiert wurden, wird eine Liste der geänderten Dateien angezeigt. Zu jedem Eintrag in der Liste wird der Pfad der Datei und ein anklickbarer Link zur Datei angezeigt.

8.5.2 Update-Operationen und Syntax

MIt Hilfe von XQuery Update Facility lassen sich die folgenden Operationen durchführen:

- Löschen 531 von einem oder mehreren Nodes
- Einfügen 532 von einem oder mehreren Nodes vor, nach oder innerhalb eines bestimmten Node
- Umbenennen 532 eines Node
- Ersetzen 533 eines Node durch eine Sequenz von Datenelementen
- Ersetzen des Werts 533 eines Node durch den Stringwert einer Sequenz von Datenelementen

Die Schlüsselwörter und die Syntax dieser Operationen sind in den Unterabschnitten dieses Abschnitts beschrieben.

8.5.2.1 Löschen von Nodes

Beschreibung und Syntax
Löscht einen oder mehrere Nodes.

```
delete node nodeSequence
delete nodes nodeSequence
```

Details

- Der Ausdruck *nodeSequence* gibt eine Sequenz des/der zu löschenden Nodes zurück. Alle ausgewählten Nodes werden für die Löschung markiert.
- Es spielt keine Rolle, ob der Singular **node** oder der Plural **nodes** verwendet wird. Die Zahl muss nicht der Anzahl der Datenelemente in *nodeSequence* entsprechen.

Beispiele

```
for $i in /book/section return
delete nodes $i/@id
```

8.5.2.2 Einfügen von Nodes

Beschreibung und Syntax
Fügt vor, nach oder innerhalb des angegebenen Ziel-Node einen oder mehrere Nodes ein.

```
insert (node|nodes) items into targetNode
insert (node|nodes) items as first into targetNode
insert (node|nodes) items as last into targetNode
insert (node|nodes) items before targetNode
insert (node|nodes) items after targetNode
```

Details

- Der Ausdruck *items* muss eine Sequenz von Datenelemente zurückgeben. Obwohl das Schlüsselwort **node|nodes** verwendet wird, kann es sich bei *items* um eine Sequenz von Datenelementen handeln, die keine Nodes sind.
- Der Ausdruck *targetNode* muss auf einen einzigen Ziel-Node verweisen.
- Wenn das Schlüsselwort **into** verwendet wird, muss es sich bei *targetNode* um einen Element- oder Dokument-Node handeln.
- Wenn der Schlüsselsatz **as first** oder **as last** verwendet wird, erfolgt die Einfügung als erste bzw. letzte Child-Elemente.
- Wenn das Schlüsselwort **into** alleine verwendet wird, werden Attribute an vorhandene Attribute angehängt und Elemente als erste Child-Elemente eingefügt.
- Wenn das Schlüsselwort **before** oder **after** verwendet wird, kann *targetNode* jeden beliebigen Typ haben.
- Wenn ein Attribut eingefügt wird, darf sein Name nicht mit dem eines bereits vorhandenen Attributs identisch sein.

Beispiele

```
for $i in /book/section return
insert nodes (attribute id { 'somevalue' }, <newelement>some content including the numbers
"{ 1 to 3}"</newelement>)
into $i
```

8.5.2.3 Umbenennen eines Node

Beschreibung und Syntax
Benennt ein Element, Attribut oder einen Processing Instruction Node um.

```
rename node targetNode as name
```

Details

- Der Ausdruck *targetNode* muss auf einen einzigen Ziel-Node verweisen, bei dem es sich um ein Element, Attribut oder eine Processing Instruction handeln kann.

© 2016-2022 Altova GmbH

- Das Ergebnis des Ausdrucks *name* muss ein QName oder String sein.
- Wenn ein QName konstruiert wird, wird der obligatorische Namespace lokal deklariert.

Beispiele

```
rename node /book/title as 'header-1'

rename node /book/title as QName("http://www.altova.com/xquf", "header-1")
```

8.5.2.4 Ersetzen eines Node

Beschreibung und Syntax
Ersetzt einen Node durch eine Sequenz von Datenelementen beliebiger Art.

```
replace node targetNode with items
```

Details

- Der Ausdruck *targetNode* muss auf einen einzigen Ziel-Node verweisen.
- Der Ausdruck *items* muss eine Sequenz von Datenelementen zurückgeben. Diese Sequenz ersetzt den Ziel-Node.
- Ein Ziel-Node kann mit Ausnahme von Attribut-Nodes durch jede Art von Sequenz ersetzt werden.
- Ein Attribut-Node kann nur durch einen Attribut-Node ersetzt werden. *Siehe Beispiel unten*.

Examples

```
replace node //hr with '<line/>'

for $i in //@height return
replace node $i with (attribute line-height{'12pt'})
```

8.5.2.5 Ersetzen des Werts eines Node

Beschreibung und Syntax
Ersetzt den Wert eines Node durch den String-Wert einer Sequenz von Datenelementen.

```
replace value of node targetNode with items
```

Details

- Der Ausdruck *targetNode* muss auf einen einzigen Ziel-Node verweisen.
- Der Ausdruck *items* muss eine Sequenz von Datenelementen zurückgeben.
- Der Inhalt des Ziel-Node wird durch den String-Wert der vom Ausdruck *items* zurückgegebenen Sequenz ersetzt. Dies bedeutet, dass der Ziel-Node nur einen Text-Node enthält.

Examples

```
for $i in //title return
replace value of node $i with ('Draft Title')
```

8.5.2.6 Die fn:put-Funktion

In XQuery Update Facility 1.0 ist die **fn.put**-Funktion als Erweiterung zur vordefinierten XQuery-Funktionsbibliothek definiert. (In diesem Abschnitt wird das **fn:** Namespace-Präfix als an den Namespace `http://www.w3.org/2005/xpath-functions` gebunden betrachtet)

```
fn:put($node as node(), $uri as xs:string) as empty-sequence()
```

Die Funktion speichert ein Dokument oder Element unter dem durch `$uri` definierten Pfad. Sie wird normalerweise aufgerufen, um eine Ressource in einem externen Speichersystem wie z.B. einem Dateisystem oder einer Datenbank zu erstellen. Die externen Auswirkungen von `fn:put` werden durch ihre Implementierung definiert, da sie sich außerhalb der Domain von XQuery auswirken. Dies geschieht, damit, wenn `fn:put` in einem Dokument-Node aufgerufen und kein Fehler gemeldet wird, eine darauf folgenden Abfrage das gespeicherte Dokument durch Aufruf von `fn:doc` mit derselben URI aufrufen kann.

Nähere Informationen dazu finden Sie in der Spezifikation.

© 2016-2022 Altova GmbH

8.6 XQuery und XML-Datenbanken

Mit Hilfe eines XQuery-Dokuments können Sie eine Abfrage an einer XML-Datenbank (XML DB) durchführen. Derzeit wird diese XQuery-Funktionalität nur für IBM DB2-Datenbanken unterstützt. Der Vorgang beim Abfragen einer XML-DB mit Hilfe von XQuery läuft folgendermaßen ab: (i) der XQuery-Prozessor wird so eingestellt, dass XML in einer Datenbank - und nicht XML in einem XML-Dokument - abgefragt wird; und (ii) die XML-Daten in der DB werden aufgerufen.

Dabei müssen die folgenden Schritte durchgeführt werden:

1. Definieren Sie das XQuery-Dokument[535] für die Abfrage der XML-DB, indem Sie am Beginn des Dokuments das Schlüsselwort `XQUERY` einsetzen.
2. Aktivieren Sie (über das Fenster "Info") für das aktive XQuery-Dokument die DB-Unterstützung[535] und stellen Sie (im Dialogfeld "Verbindungsassistent") eine Verbindung zur DB her[535].
3. Fügen Sie im XQuery-Dokument DB-spezifische XQuery-Erweiterungen ein[536], um die DB-Daten aufrufen zu können und für XQuery-Operationen verfügbar zu machen.
4. Führen Sie das XQuery-Dokument[538] in XMLSpy aus.

Einrichten des XQuery-Dokuments für die Abfrage der XML DB

Um das XQuery-Dokument für eine Abfrage einer XML DB zu konfigurieren, öffnen Sie das XQuery-Dokument (oder erstellen Sie ein neues XQuery-Dokument) und geben Sie das Schlüsselwort `XQUERY` (Groß- und Kleinschreibung irrelevant) am Beginn des Dokuments (vor dem Prolog) ein; *siehe Beispiel unten*.

```
XQUERY (: Retrieve details of all customers :)
declare default element namespace "http://www.altova.com/xquery/databases/db2";
<a> {db2-fn:xmlcolumn("CUSTOMER.INFO")} </a>
```

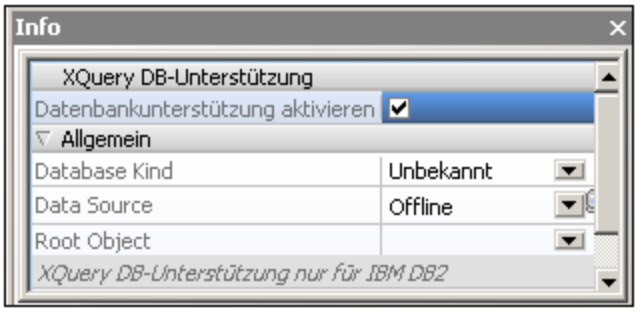
Wenn im Dokument der optionale Ausdruck `xquery version` verwendet wird, wird dennoch das Schlüsselwort `XQUERY` benötigt.

```
XQUERY xquery version "1.0"; (: Retrieve details of all customers :)
declare default element namespace
"http://http://www.altova.com/xquery/databases/db2";
<a> {db2-fn:xmlcolumn("CUSTOMER.INFO")} </a>
```

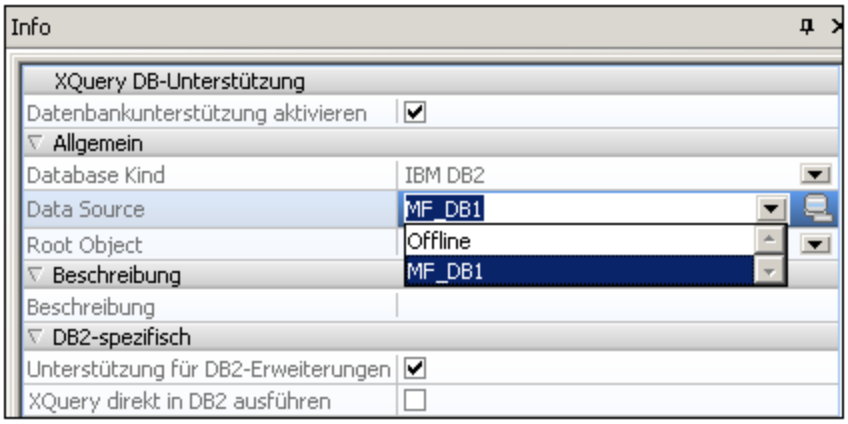
Anmerkung: Die integrierten XMLSpy XQuery-Prozessoren lesen das Schlüsselwort `XQUERY` als Indikator dafür, dass eine XML DB aufgerufen werden soll. Folglich führt der Versuch, ein XQuery-Dokument, das das Schlüsselwort `XQUERY` enthält, an einem XML-Dokument auszuführen, das nicht in einer XML DB enthalten ist, zu einem Fehler.

Aktivieren der DB-Unterstützung für XQuery und Herstellen einer Verbindung zur DB

Die DB-Unterstützung für ein XQuery-Dokument wird durch Aktivierung des Kontrollkästchens "Datenbankunterstützung aktivieren" im Fenster "Info" (*Abbildung unten)* aktiviert. Beachten Sie: Die Unterstützung von DB muss für jedes XQuery-Dokument separat aktiviert werden. Jedes Mal muss ein XQuery-Dokument neu geöffnet werden.



Wenn Sie die DB-Unterstützung im Fenster "Info" aktivieren, erscheint ein Verbindungsassistent, mit Hilfe dessen Sie eine Verbindung zu einer Datenbank herstellen können. Derzeit werden nur IBM DB2-Datenbanken unterstützt. Die Herstellung einer Verbindung zu einer DB ist im Abschnitt Herstellen einer Verbindung zu einer Datenbank[969] beschrieben. Wenn bereits Verbindungen zu Datenquellen bestehen, werden diese im Fenster "Info" in der Auswahlliste "Datenquellen" aufgelistet (*Abbildung unten*) und Sie können eine dieser Datenquellen als Datenquelle für das aktive XQuery-Dokument auswählen. Im Fenster "Info" können Sie auch das Root-Objekt aus den verfügbaren Root-Objekten in der Auswahlliste auswählen.



Der Verbindungsassistent (mit Hilfe dessen Sie eine Verbindung zu einer DB herstellen können), kann jederzeit durch Klicken auf die Schaltfläche  im Fenster "Info" aufgerufen werden.

Anmerkung: Beim Schließen eines XQuery-Dokuments geht auch die Verbindung zur Datenbank verloren. Wenn Sie das XQuery-Dokument später wieder öffnen, müssen Sie auch die DB-Verbindung wieder herstellen.

IBM DB2-spezifische Erweiterungen für XQuery

Zum Abrufen von Daten aus einer IBM DB2-Datenbank können in XQuery-Dokumenten zwei IBM DB2-spezifische Funktionen verwendet werden:

- `db2-fn:xmlcolumn` ruft eine gesamte XML-Spalte ab, ohne Sie zu durchsuchen oder zu filtern.
- `db2-fn:sqlquery` ruft Werte auf Basis einer SQL `SELECT`-Anweisung ab

Sie können anschließend mit den XML-Daten, die mit Hilfe dieser Funktionen abgerufenen wurden, mit XQuery-Standardkonstrukten arbeiten. *Siehe Beispiele unten*.

© 2016-2022 Altova GmbH

db2-fn:xmlcolumn: Das Argument der Funktion ist ein String-Literal, bei dem die Groß- und Kleinschreibung beachtet werden muss, das eine XML-Spalte in einer Tabelle ermittelt. Das String-Literal-Argument muss ein qualifizierter Spaltenname von Typ XML sein. Die Funktion gibt alle XML-Daten in der Spalte als Sequenz zurück, ohne daran eine Suchbedingung anzuwenden. Im folgenden Beispiel werden alle Daten der (XML)-Spalte `INFO` der Tabelle `CUSTOMER` innerhalb eines `<newdocelement>` Elements der obersten Ebene zurückgegeben:

```
XQUERY (: Retrieve details of all customers :)
declare default element namespace "http://www.altova.com/xquery/databases/db2";
<newdocelement> {db2-fn:xmlcolumn("CUSTOMER.INFO")} </newdocelement>
```

Die abgerufenen Daten können anschließend mit XQuery-Konstrukten abgefragt werden. Im Beispiel unten werden die XML-Daten, die aus der XML-Spalte `INFO` der Tabelle `CUSTOMER` abgerufen wurden, mit Hilfe eines XQuery-Konstrukts gefiltert, sodass nur die Profile der Kunden aus Toronto abgerufen werden.

```
XQUERY (: Retrieve details of Toronto customers :)
declare default element namespace "http://www.altova.com/xquery/databases/db2";
<newdocelement> {db2-fn:xmlcolumn("CUSTOMER.INFO")/customerinfo[addr/city='Toronto']}
</newdocelement>
```

Anmerkung: Im Beispiel oben ist das Dokument-Element der XML-Dateien in den einzelnen Zellen das Element `customerinfo`, und als Root-Node der XML-Sequenz, die von `db2-fn:xmlcolumn` zurückgegeben wird, wird ein abstrakter Node oberhalb der `customerinfo` Nodes angenommen.

db2-fn:sqlquery: Die Funktion verwendet eine SQL Select-Anweisung als Argument und gibt eine Sequenz von XML-Werten zurück. Die abgerufene Sequenz wird anschließend mit XQuery-Konstrukten abgefragt. Im folgenden Beispiel wird die Spalte `INFO` in der Tabelle `CUSTOMER` nach Datensätzen durchsucht, deren `CID`-Feld einen Wert zwischen `1000` und `1004` hat.

```
XQUERY (: Retrieve details of customers by Cid:)
declare default element namespace "http://www.altova.com/xquery/databases/db2";

<persons>
   {db2-fn:sqlquery("SELECT info FROM customer WHERE CID>1000 AND CID<1004")/
   <person>
      <id>{data(@Cid)}</id>
      <name>{data(name)}</name>
   </person>}
</persons>
```

Die oben angeführte XQuery gibt die folgende Ausgabe zurück:

```
<persons xmlns="http://www.altova.com/xquery/databases/db2">
   <person>
       <id>1001</id>
       <name>Kathy Smith</name>
   </person>
   <person>
       <id>1002</id>
       <name>Jim Jones</name>
   </person>
   <person>
       <id>1003</id>
       <name>Robert Shoemaker</name>
```

```
   </person>
</persons>
```

Beachten Sie die folgenden Punkte:

- Die Standard-Element-Namespace-Deklaration im Prolog gilt für das gesamte XQuery-Dokument und dient sowohl zur Navigation im XML-Dokument als auch zur Erstellung neuer Elemente. D.h. der XQuery-Selektor `name` wird zu `<default-element-namespace>:name` erweitert und Elemente, die erstellt werden, wie z.B. `persons`, befinden sich im Standard-Element-Namespace.
- Bei der SQL Select-Anweisung wird die Groß- und Kleinschreibung nicht beachtet.
- Die `WHERE`-Klausel der Select-Anweisung sollte ein weiteres Datenbankelement referenzieren - keinen Node innerhalb der aufgerufenen XML-Datei.
- Der Schrägstrich "`/`" hinter der Funktion `db2-fn:sqlquery` steht für das erste Datenelement der zurückgegebenen Sequenz und dieses Datenelement ist der Kontext-Node für die weitere Navigation.

Ausführen des XQuery-Dokuments

Um das XQuery-Dokument auszuführen, wählen Sie den Befehl **XQuery-Ausführung** (Menü **XSL/XQuery**). Alternativ dazu können Sie auch **Alt+F10** drücken oder auf die Schaltfläche "XQuery-Ausführung"  klicken. Das Ergebnis der Ausführung wird in einem neuen Dokument angezeigt.

© 2016-2022 Altova GmbH

9 XSLT/XQuery Debugger und Profiler

XMLSpy enthält einen XSLT/XQuery Debugger[540] und einen XSLT/XQuery Profiler[567], damit Sie schneller korrekte XSLT- und XQuery-Dokumente erstellen können. Diese beiden Funktionalitäten sind in den Unterabschnitten dieses Abschnitts beschrieben.

9.1 XSLT und XQuery Debugger

Der XSLT and XQuery Debugger dient zum Testen und Debuggen von XSLT Stylesheets und XQuery-Dokumenten. Über die Oberfläche des XSLT and XQuery Debugger können Sie gleichzeitig eine Ansicht des XSLT/XQuery-Dokuments, des Ergebnisdokuments und des XML-Quelldokuments anzeigen. Sie können sich anschließend Schritt für Schritt durch das XML-XSLT/XQuery-Dokument bewegen. Bei jedem Schritt wird die entsprechende Ausgabe generiert. Wenn eine XML-Quelldatei angezeigt wird, wird die entsprechende Position in dieser Datei bei jedem Schritt markiert. Gleichzeitig zeigen Fenster der Benutzeroberfläche Informationen zur Behebung von Fehlern an.

Der XSLT and XQuery Debugger wird immer während einer **Debugger-Sitzung** geöffnet. Es gibt die folgenden Arten von Debugger-Sitzungen:

- XSLT 1.0, wobei der integrierte Altova XSLT 1.0-Prozessor verwendet wird
- XSLT 2.0, wobei der integrierte Altova XSLT 2.0-Prozessor verwendet wird
- XSLT 3.0, wobei der integrierte Altova XSLT 3.0-Prozessor verwendet wird
- XQuery 1.0, wobei der integrierte Altova XQuery 1.0-Prozessor verwendet wird
- XQuery 3.1, wobei der integrierte Altova XQuery 3.1-Prozessor verwendet wird

Anmerkung: XQuery Update Facility 1.0 und XQuery Update Facility 3.0 werden in XSLT and XQuery Debugger derzeit nicht unterstützt.

Die Art der Debugger-Sitzung wird automatisch gewählt und hängt von der Art des Dokuments ab, von dem aus die Debugger-Sitzung gestartet wird (Dieses Dokument bezeichnen wir hier als das *aktive Dokument* oder die *aktive Datei.*). XSLT-Debugger-Sitzungen werden von XSLT-Dateien aus gestartet (Die Version hängt vom Wert des Attributs `version` des Elements `xsl:stylesheet` (oder `xsl:transform`) im XSLT Stylesheet ab (`"1.0"` für XSLT 1.0, `"2.0"` für XSLT 2.0 und `3.0` für XSLT 3.0)). XQuery-Debugger-Sitzungen werden von XQuery-Dateien aus auf ähnliche Art gestartet. Wenn die aktive Datei eine XML-Datei ist, hängt die Auswahl davon ab, ob Sie eine XSLT- oder eine XQuery-Datei auf die XML-Datei anwenden. In ersterem Fall hängt die Auswahl der Debugger-Art davon ab, ob es sich beim Stylesheet um ein XSLT 1.0-, ein XSLT 2.0- oder ein XSLT 3.0-Stylesheet handelt.

In der nachfolgenden Tabelle sehen Sie eine Zusammenfassung dieser Informationen.

Aktive Datei	Verknüpfte Datei	Debugger-Sitzung
XSLT 1.0	XML; (erforderlich)	XSLT 1.0 (mit dem integrierten Altova XSLT 1.0-Prozessor)
XSLT 2.0	XML; (erforderlich)	XSLT 2.0 (mit dem integrierten Altova XSLT 2.0-Prozessor)
XSLT 3.0	XML oder JSON; (erforderlich)	XSLT 3.0 (mit dem integrierten Altova XSLT 3.0-Prozessor)
XQuery 1.0	XML; (optional)	XQuery 1.0 (mit dem integrierten Altova XQuery 1.0-Prozessor)
XQuery 3.1	XML oder JSON; (optional)	XQuery 3.1 (mit dem integrierten Altova XQuery 3.0-Prozessor)
XML	XSLT 1.0, 2.0 oder 3.0 oder XQuery 1.0 oder 3.1; (erforderlich)	XSLT 1.0, 2.0 oder 3.0 oder XQuery 1.0 oder 3.1. Die XSLT-Version hängt vom Wert des Attributs `version` des Elements `xsl:stylesheet` (oder `xsl:transform`) im XSLT Stylesheet ab.

© 2016-2022 Altova GmbH

Nähere Informationen zu den drei Altova-Prozessoren finden Sie in Anhängen[2136] unter Informationen zu den XSLT- und XQuery-Prozessoren[2137].

Automatisieren von XSLT- und XQuery-Aufgaben mit RaptorXML Server 2023

Altova RaptorXML Server ist eine kostenlose Applikation, die den RaptorXML-Validator sowie einen XLST 1.0, 2.0 und 3.0-Prozessor sowie einen XQuery 1.0- und 3.1-Prozessor enthält. Das Tool kann über die Befehlszeile, über eine COM-Schnittstelle, in Java-Programmen und in .NET-Applikationen zur Validierung von XML-Dokumenten, zum Transformieren von XML-Dokumenten unter Verwendung von XSLT Stylesheets sowie zum Ausführen von XQuery-Dokumenten verwendet werden.

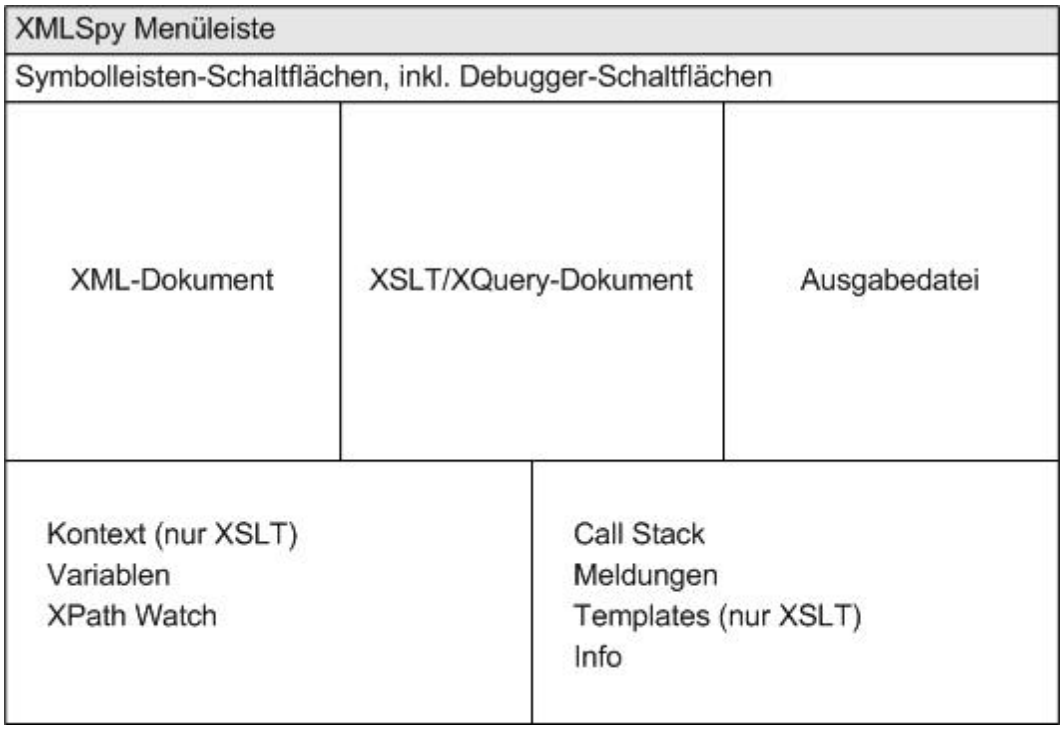
XSLT- und XQuery-Aufgaben können daher mittels RaptorXML Server automatisiert werden. So können Sie z.B. eine Batch-Datei erstellen, die RaptorXML aufruft, um damit eine Reihe von XML-Dokumenten zu transformieren oder eine Reihe von XQuery-Dokumenten auszuführen. Nähere Informationen dazu finden Sie in der RaptorXML Server-Dokumentation.

Altova Website: XSLT Debugger, XQuery Debugger

9.1.1 Funktionsweise und Benutzeroberfläche

Die grundlegende Vorgangsweise beim Debuggen von XSLT- und XQuery-Dateien mit XSLT and XQuery Debugger ist die folgende. Beachten Sie, dass ein Unterschied besteht zwischen einer Debugging-Sitzung und dem Debugger, auch wenn beide mit dem Befehl **XSL/XQuery | Debugger starten** gestartet werden. Sie müssen zuerst die Debugging-Sitzung starten und das XSLT- oder XQuery-Dokument anschließend Schritt für Schritt mit dem Debugger durchgehen.

- Öffnen Sie eine Debugger-Sitzung (mit dem Befehl **XSL/XQuery | Debugger starten**. Die Art der Sitzung (XSLT 1.0, XSLT 2.0, XSLT 3.0 oder XQuery 1.0 oder XQuery 3.1) wird anhand der aktiven Datei ausgewählt (*siehe* XSLT und XQuery Debugger[540]). Der XSLT and XQuery Debugger kann nur in der Textansicht und der Grid-Ansicht verwendet werden. Wenn die Ansicht des aktiven Dokuments beim Start der Debugger-Sitzung nicht die Textansicht oder die Grid-Ansicht ist, werden Sie gefragt, ob in die Textansicht, also die Standardansicht des XSLT and XQuery Debugger gewechselt werden darf. Sie können diese Option im Dialogfeld "Debugger-Einstellungen"[546] auch permanent aktivieren.
- Gehen Sie das XSLT- bzw. XQuery-Dokument schrittweise mit Hilfe der Befehle **Einsteigen, Aussteigen** und **Überspringen** (Menü **XSL/XQuery**) durch. (Wenn Sie an diesem Punkt den Befehl **Debugger starten** wählen, überprüft der Debugger die gesamte Transformation oder Ausführung und hält nur an Breakpoints an. Wenn kein Breakpoint gesetzt wurde, überprüft der Debugger die gesamte Transformation in einem einzigen Schritt, ohne die Debugging-Ergebnisse anzuzeigen.) Wenn eine XML-Datei mit der Sitzung verknüpft ist, wird die jeweilige Stelle in der XML-Datei markiert. Gleichzeitig wird für jeden Schritt in der Ergebnisdatei eine Ausgabe generiert, d.h. das Ergebnisdokument wird Schritt für Schritt erzeugt. Auf diese Art können Sie analysieren, welche Auswirkung jede einzelne Anweisung in der XSLT- bzw. XQuery-Datei hat.



Alternativ zur Anzeige aller drei Dokumente (XML, XSLT/XQuery, Ergebnisdokument) (siehe Abb. oben), können Sie auch eine Ansicht mit zwei Dokumenten (XSLT/XQuery und Ergebnisdokument) oder eine Ansicht mit nur einem einzigen Dokument auswählen.

- Während eine Debugger-Sitzung geöffnet ist, werden auf der Benutzeroberfläche Informationsfenster mit Informationen über die verschiedenen Aspekte der Transformation/Ausführung angezeigt (Variablen, XPath-Überwachung, Call Stack, Meldungen, Info, usw.).
- Sie können den Debugger während einer Debugger-Sitzung anhalten (dies ist nicht dasselbe wie die Beendung der Debugger-Sitzung), um Änderungen am Dokument vorzunehmen. Alle Editierfunktionen, die in Ihrer XMLSpy-Umgebung zur Verfügung stehen, können auch beim Editieren einer Datei während einer Debugger-Sitzung verwendet werden. Wenn der Debugger angehalten wird, bleibt die XSLT and XQuery Debugger-Oberfläche geöffnet und Sie haben Zugriff auf alle Informationen in den Debugger-Fenstern. Sie können nach dem Anhalten den Debugger in derselben Debugger-Sitzung erneut starten und das XSLT/XQuery-Dokument von Beginn an erneut debuggen.
- Sie können in der XSLT-Datei Breakpoints setzen, um die Verarbeitung an bestimmten Punkten zu unterbrechen. Damit wird der Vorgang beschleunigt, da Sie nicht jede einzelne Anweisung im XSLT- bzw. XQuery-Dokument manuell durchgehen müssen.
- Sie können in der XSLT-Datei Tracepoints setzen. Bei Anweisungen in denen ein Tracepoint gesetzt wurde, wird der Wert dieser Anweisung ausgegeben, sobald diese Anweisung erreicht wird.
- Beenden einer Debugger-Sitzung. Dadurch wird die XSLT and XQuery Debugger-Oberfläche geschlossen und Sie kehren zur vorher angezeigten XMLSpy Umgebung zurück. Die Informationen in den Informationsfenstern stehen nun nicht mehr zur Verfügung. Die Informationen zu den Breakpoints und Tracepoints bleiben jedoch in den jeweiligen Dateien erhalten, bis die Datei geschlossen wird. (Wenn Sie also eine weitere Debugger-Sitzung starten, bei der eine Datei mit Breakpoints verwendet wird, werden die Breakpoints in der neuen Debugger-Sitzung berücksichtigt.)

Anmerkung: Die Debugger-Symbolleiste und -Symbole werden automatisch angezeigt, wenn eine Debugger-Sitzung gestartet wird.

 © 2016-2022 Altova GmbH

9.1.2 Befehle und Symbolleistenschaltflächen

Die Debugger-Befehle stehen im Menü **XSL/XQuery** und als Symbolleistenschaltflächen zur Verfügung. Die Debugger-Symbole erscheinen automatisch in der Symbolleiste, sobald eine Debugger-Sitzung geöffnet wird.



Debugger starten (Alt+F11)
Startet die Verarbeitung des XSLT/XQuery-Dokuments bzw. fährt damit fort, bis das Ende des Dokuments erreicht ist. Wurden Breakpoints gesetzt, so wird die Verarbeitung an diesen Stelle angehalten. Wenn der Debugger noch nicht gestartet wurde, starten Sie mit diesem Symbol die Sitzung, bis sie am ersten Node angehalten wird. Bei laufender Debugger-Sitzung wird das XSLT/XQuery-Dokument bis zu Ende verarbeitet oder bis der nächste Breakpoint erreicht wird. Wenn Tracepoints gesetzt wurden, wird der Wert der Anweisung für die der Tracepoint gesetzt wurde, im Fenster "Trace" angezeigt, wenn diese Anweisung erreicht wird.



Nur aktives Dokument ansehen
Maximiert das Fenster des gerade aktiven Dokuments auf der Debugger-Oberfläche.



XSLT/XQuery und Ausgabe ansehen

Zeigt das XSLT- und das Ausgabedokument an und blendet das XML-Dokument aus.



XML, XSLT/XQuery und Ausgabe anzeigen
Zeigt das XML-, XSLT/XQuery- und das Ausgabedokument an. Dies ist die Standardansicht, wenn ein XML-Dokument für die Debugger-Sitzung verknüpft wurde.



Debugger anhalten

Hält den Debugger an. Dies ist nicht dasselbe wie das Beenden der Debugger-Sitzung, in der der Debugger ausgeführt wird. Diese Funktion eignet sich dazu, um ein Dokument während der Debugger-Sitzung zu bearbeiten oder innerhalb derselben Debugger-Sitzung andere Dateien zu verwenden. Nachdem Sie den Debugger angehalten haben, müssen Sie ihn erneut starten, um am Anfang des XSLT/XQuery-Dokuments mit dem Debuggen zu beginnen.



Einsteigen (F11)
Geht die einzelnen Nodes und XPath-Ausdrücke im Stylesheet schrittweise durch. Dieser Befehl dient auch zum erneuten Starten des Debuggers, nachdem er angehalten wurde.



Überspringen (Strg+F11)
Überspringt den aktuellen Node und geht zum nächsten Node auf derselben Ebene bzw. zum nächsten Node auf der nächsthöheren Ebene. Dieser Befehl dient auch zum erneuten Starten des Debuggers, nachdem er angehalten wurde.



Aussteigen (Umschalt+F11)
Steigt aus dem aktuellen Node aus und geht zum nächsten Geschwisternode des Parent-Node bzw. zum nächsthöheren Node oberhalb des Parent-Node.



Aktuellen Execution Node einblenden
Zeigt den aktuellen Execution Node im XSLT/XQuery-Dokument und den entsprechenden Kontext-Node im XML-Dokument an bzw. wählt diese aus. Dies ist nützlich, wenn Sie auf andere Register geklickt haben, die bestimmten Code im XSLT Stylesheet bzw. der XML-Datei anzeigen oder markieren, und zum Ausgangsort zurückkehren wollen.



Debugger neu starten
Löscht das Ausgabefenster und startet die Debugger-Sitzung mit den aktuell ausgewählten Dateien neu.



Breakpoint einfügen/löschen (F9)
Fügt einen Breakpoint an der aktuellen Cursorposition ein bzw. löscht diesen. Breakpoints innerhalb einer Zeile können für Nodes sowohl im XSLT/XQuery-Dokument als auch im XML-Dokument definiert werden. Sie legen fest, an welcher Stelle die Verarbeitung angehalten werden soll. Oberhalb des Nodes, an dem Sie einen Breakpoint gesetzt haben, erscheint eine strichlierte rote Linie. Für Abschlussnodes können keine Breakpoints gesetzt werden. Breakpoints in Attributen werden in XSLT-Dokumenten ignoriert. Der Befehl kann auch durch Rechtsklick auf die gewünschte Stelle aufgerufen werden.



Tracepoint einfügen/löschen (Umschalt+F9)
Fügt einen Tracepoint an der aktuellen Cursorposition ein. In XSLT-Dokumenten können Tracepoints innerhalb einer Zeile gesetzt werden. Wenn beim Debuggen eine Anweisung mit einem Tracepoint erreicht wird, wird das Ergebnis dieser Anweisung im Fenster "Trace" ausgegeben. Oberhalb von Nodes, in denen ein Tracepoint gesetzt wurde, erscheint eine strichlierte blaue Linie. Tracepoints können nicht in schließenden Nodes gesetzt werden. Dieser Befehl steht auch zur Verfügung, wenn Sie mit der rechten Maustaste an die Stelle für den Tracepoint klicken.



Breakpoint aktivieren/deaktivieren (Strg+F9)
Dieser Befehl (in der Symbolleiste nicht verfügbar) schaltet bereits definierte Breakpoints ein oder aus. Ein Breakpoint, der in aktiviertem Zustand rot angezeigt wird, erscheint grau, wenn er deaktiviert ist. Der Debugger

© 2016-2022 Altova GmbH

hält an deaktivierten Breakpoints nicht an. Um einen Breakpoint ein- bzw. auszuschalten, setzen Sie den Cursor in den jeweiligen Node-Namen und klicken Sie auf den Befehl "Breakpoint aktivieren/deaktivieren". Der Befehl kann auch durch Rechtsklick auf die gewünschte Stelle aufgerufen werden.



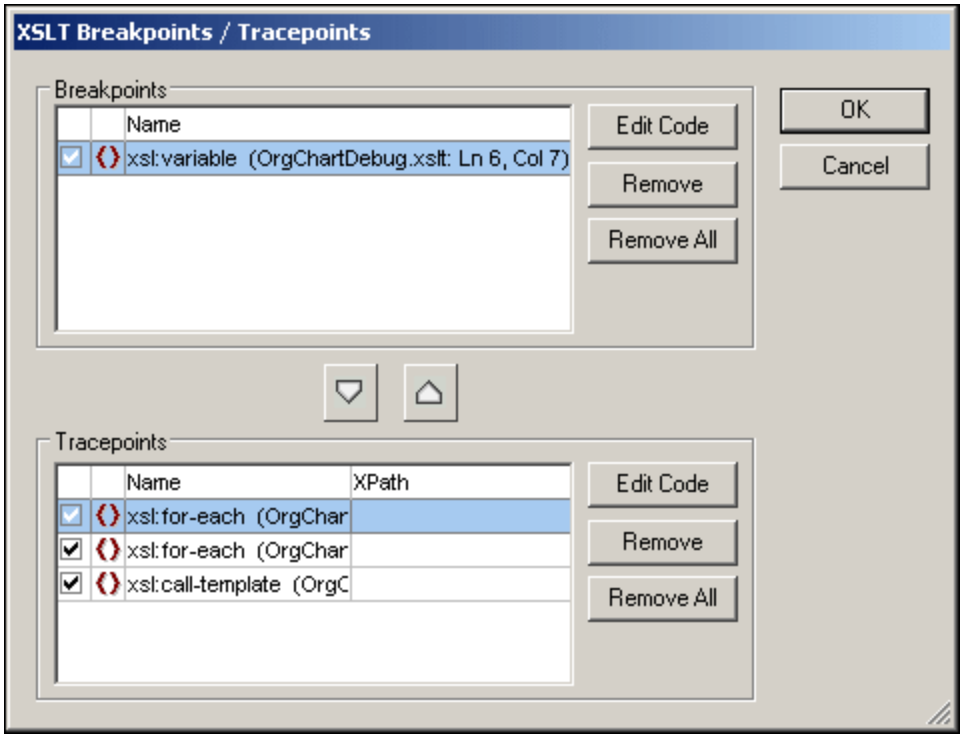
Tracepoint aktivieren/deaktivieren (Umschalt+Strg+F9)
Dieser Befehl (in der Symbolleiste nicht verfügbar) aktiviert oder deaktiviert bereits definierte Tracepoints. Ein Tracepoint, der in aktiviertem Zustand blau angezeigt wird, erscheint grau, wenn er deaktiviert ist. Bei deaktivierten Tracepoints wird im Fenster "Trace" keine Ausgabe angezeigt. Um einen Tracepoint zu deaktivieren bzw. zu aktivieren, setzen Sie den Cursor in den jeweiligen Node-Namen und klicken Sie auf den Befehl "Tracepoint aktivieren/deaktivieren". Der Befehl kann auch durch Rechtsklick auf die Stelle für den Tracepoint aufgerufen werden.



Debugger-Sitzung beenden
Beendet die Debugger-Sitzung und kehrt zur normalen XMLSpy-Ansicht zurück, die vor dem Start der Debugger-Sitzung aktiv war. Ob die Ausgabedokumente, die für die Debugger-Sitzung geöffnet wurden, weiterhin geöffnet bleiben, hängt von den Einstellungen ab, die Sie im Dialogfeld XSLT/XQuery Debugger-Einstellungen 546 vorgenommen haben.



Dialogfeld "XSLT Breakpoints/Tracepoints"
Dieser Befehl ruft das Dialogfeld "XSLT/XQuery Breakpoints/Tracepoints" auf, in dem eine Liste aller in den Dateien der aktiven Debugger-Sitzung definierten Breakpoints/Tracepoints (einschließlich aller deaktivierten) angezeigt wird.

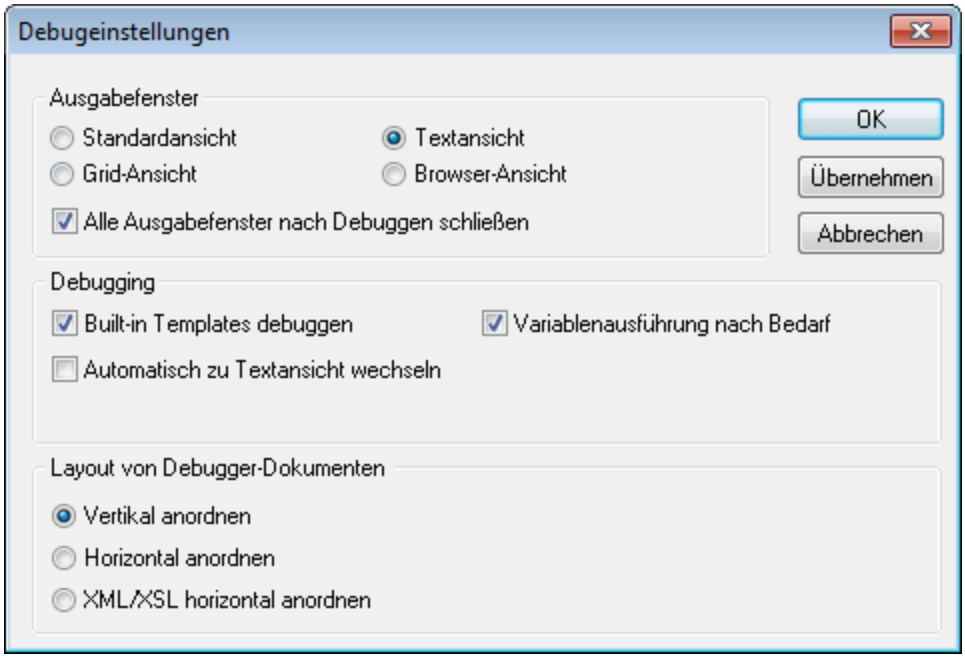


Die Kontrollkästchen zeigen an, ob ein Breakpoint/Tracepoint aktiviert (mit Häkchen versehen) oder deaktiviert ist. Sie können den markierten Breakpoint/Tracepoint durch Klicken auf die Schaltfläche **Löschen** entfernen. Bei Klicken auf die Schaltfläche **Alle löschen** werden alle Breakpoints bzw. Tracepoints gelöscht. Bei Auswahl der Schaltfläche **Code bearbeiten** wird der entsprechende Breakpoint/Tracepoint in der Datei direkt angezeigt.

9.1.3 Debugeinstellungen

Im Dialogfeld XSLT and XQuery Debugger-Einstellungen können Sie Debugger- und Ausgabeoptionen einstellen, die für alle Debugger-Sitzungen gelten. Um das Einstellungsdialogfeld aufzurufen, wählen Sie den Befehl **XSL/XQuery | Debugeinstellungen** oder klicken Sie auf das Symbol  in der Symbolleiste. Die verschiedenen Einstellungen werden weiter unten beschrieben.

© 2016-2022 Altova GmbH



Ausgabefenster

Konfiguriert die Ansicht des Ausgabefensters (Standardansicht, Textansicht, Grid-Ansicht oder Browser-Ansicht). Die Standardansicht ist die Ansicht, die im Dialogfeld "Optionen" (**Extras | Optionen** 1602) im Abschnitt "Dateiarten" für einen Dateityp (erkennbar an der Dateierweiterung, z.B. `.xslt` oder `.xq`) ausgewählt wurde. Für XSLT-Transformationen ist der Ausgabedateityp in der XSLT-Datei definiert. Für XQuery-Ausführungen ist der Ausgabedateityp durch die im Dialogfeld "XQuery-Einstellungen" ausgewählte "Serialization Method" festgelegt. (*siehe unten*).

Die Option "Alle Ausgabefenster nach Debuggen schließen" gibt Ihnen die Möglichkeit, die Ausgabefenster, die während der Debugger-Sitzung geöffnet waren, auch nach dem Schließen der Debugger-Sitzung geöffnet zu lassen.

Debugging

Bei Auswahl der Einstellung "Built-in Templates debuggen" **steigt** der Debugger, in built-in Templates **ein**, wann immer dies notwendig ist. Dies hat nichts mit der **Anzeige** von built-in Templates zu tun, wenn Sie im Abschnitt "Templates" auf diese Art von Template-Eintrag klicken oder wenn der Callstack einen Node aus der Built-in Template-Datei anzeigt.

Der XSLT Debugger kann nur in der Text- oder der Grid-Ansicht verwendet werden. Bei Aktivierung der Option "Automatisch zu Textansicht wechseln" wird zum Debuggen automatisch in die Textansicht eines Dokuments gewechselt, wenn ein Dokument nicht in der Text- oder Grid-Ansicht angezeigt wird. (Der XQuery Debugger kann nur in der Textansicht verwendet werden.) Wenn das Kontrollkästchen *Variablenausführung nach Bedarf* aktiviert ist, wird bei Aufruf der Variablen in die Definition einer Variablen eingestiegen. Andernfalls steigt der Debugger nicht in die Variablendefinition ein, wenn er auf einen Variablenaufruf trifft, sondern fährt mit dem nächsten Schritt fort.

Layout von Debugger-Dokumenten

Die Debugger-Dokumente sind die Dokumente, die im Debugger geöffnet sind. Sie können auswählen, ob diese Dokumente vertikal oder horizontal bzw. mit der Option "XML/XSL horizontal anordnen" angeordnet werden sollen, wobei das Ergebnisdokument im letzteren Fall vertikal neben dem XML- und dem XSLT-Dokument angeordnet wird.

9.1.4 Starten einer Debugger-Sitzung

Am einfachsten lässt sich eine Debugger-Sitzung von einer XSLT-, XQuery- oder XML-Datei aus starten. Wenn der aktiven Datei bereits die erforderliche Datei (siehe Tabelle der verknüpften Dateien 540) zugewiesen wurde, wird die Debugger-Sitzung sofort gestartet. Andernfalls werden Sie aufgefordert, die erforderliche Datei zuzuweisen. Da XQuery-Dateien eine Verknüpfung mit einer XML-Datei weder benötigen noch enthalten, können Sie einstellen, ob Sie bei jedem Start einer XQuery Debugger-Sitzung (von einer XQuery-Datei aus) aufgefordert werden sollen, eine optionale XML-Dateiverknüpfung anzugeben, oder nicht.

Vordefinierte Verknüpfungen

Vordefinierte Verknüpfungen sind nur für XSLT Debugger-Sitzungen relevant und beziehen sich auf Fälle, in denen die Dateiverknüpfung bereits in der aktiven Datei vorhanden ist. Um in einer XML- oder XSLT-Datei eine Zuweisung vorzunehmen, gehen Sie folgendermaßen vor:

- In XML/JSON-Dateien: Öffnen Sie die Datei, wählen Sie **XSL/XQuery | XSL zuweisen** und wählen Sie die XSLT-Datei aus.
- In XSLT-Dateien: Öffnen Sie die Datei, wählen Sie den Befehl **XSL/XQuery | XML-Beispieldatei zuweisen...** und wählen Sie die XML-Datei aus.

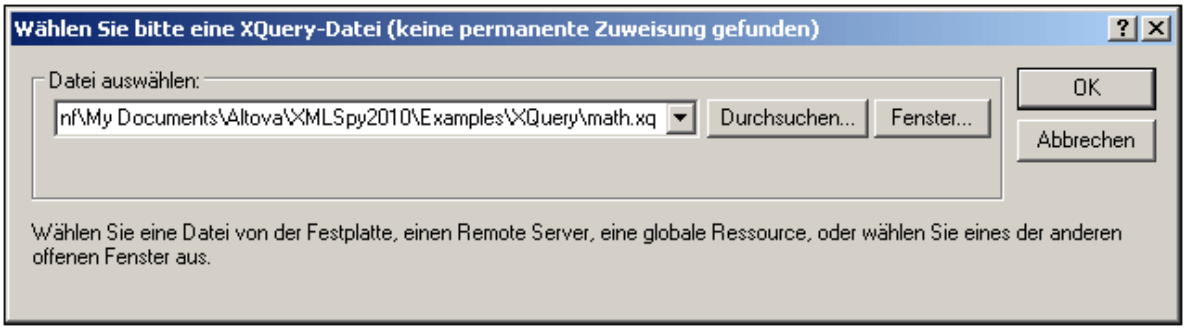
Wenn Sie den Befehl **XSL/XQuery | Debugger starten** wählen, wird die Debugger-Sitzung direkt gestartet, d.h. ohne dass Sie aufgefordert werden, die aktive Datei mit einer Datei zu verknüpfen.

Direkte Zuweisung

Falls die aktive Datei keine vordefinierte Verknüpfung enthält, werden Sie aufgefordert, eine anzugeben. Wenn Sie den Befehl **XSL/XQuery | Debugger starten** wählen, geschieht Folgendes:

- bei XML-Dateien: Sie werden aufgefordert, eine XSLT- oder XQuery-Datei auszuwählen.
- bei JSON-Dateien: Sie werden aufgefordert, eine XSLT-, XQuery- oder XQuery Update-Datei auszuwählen.
- bei XSLT-Dateien: Sie werden aufgefordert, eine XML-Datei auszuwählen.
- bei XQuery-Dateien: Sie haben die Option, eine XML-Datei auszuwählen. Diese Option kann übersprungen werden.

© 2016-2022 Altova GmbH



(Das Dialogfeld in der Abbildung wird angezeigt, wenn Sie eine Debugger-Sitzung von einer XQuery-Datei aus starten.)

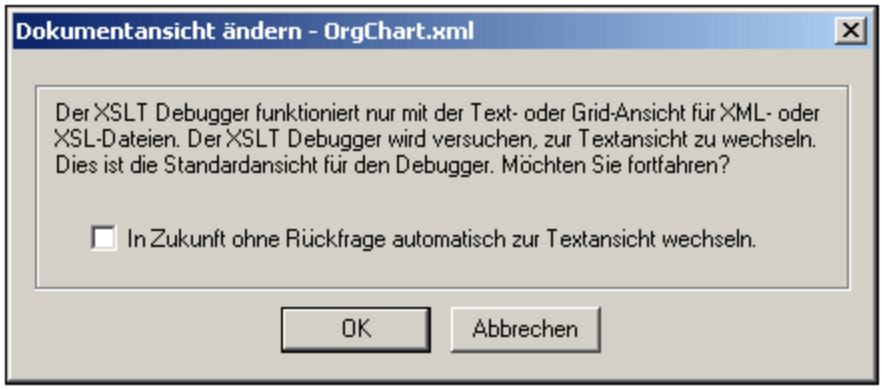
Nach Auswahl der erforderlichen Verknüpfungsdatei bzw. Auslassen der optionalen Verknüpfung wird die Debugger-Sitzung gestartet.

Alternative Methode zur Dateizuordnung

Im Dialogfeld "Projekteigenschaften" können Sie Verknüpfungen vordefinieren. Klicken Sie auf **Projekt | Projekteigenschaften** und nehmen Sie die gewünschten Dateizuweisungen vor, indem Sie das Kontrollkästchen **XSL-Vorlage** / **XML-Vorlage** aktivieren.

Debugger-Ansicht

Der XSLT and XQuery Debugger funktioniert nur in der Text- und der Grid-Ansicht. Wenn die jeweilige XML- oder XSLT-Datei in einer anderen Ansicht geöffnet ist, bzw. wenn eine SPS-Datei mit einer XML-Datei verknüpft ist, erscheint das folgende Dialogfeld, wenn Sie eine Debugger-Sitzung unter Verwendung einer dieser Dateien starten.

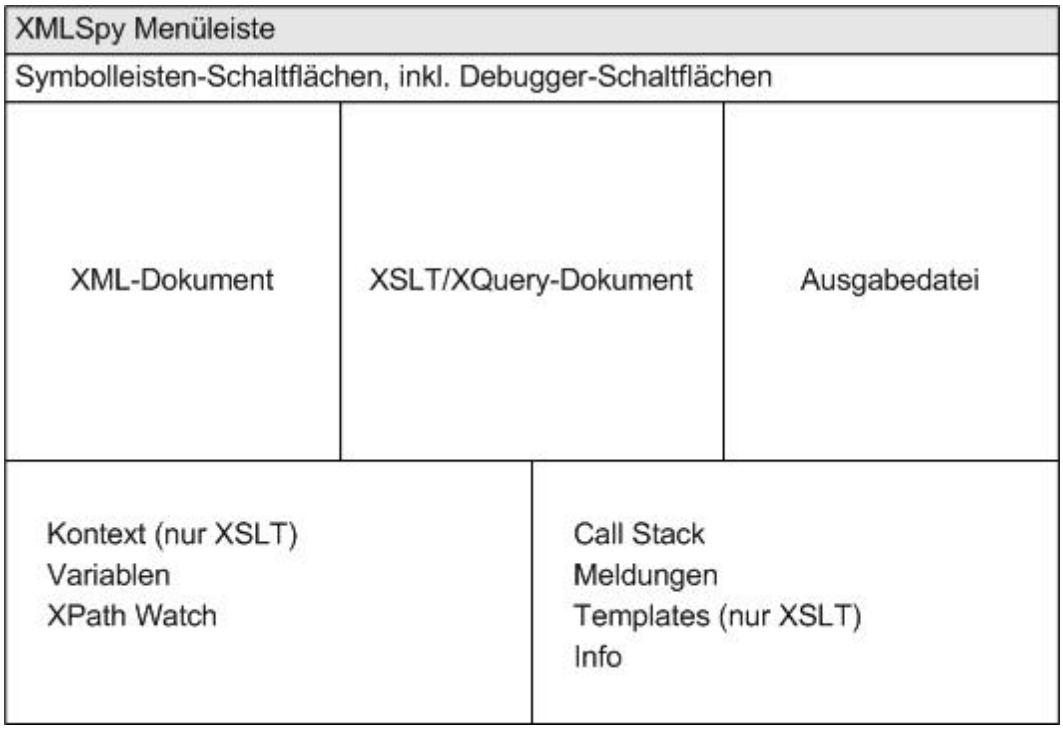


Wenn Sie auf OK klicken, wird das Dokument in der Textansicht geöffnet. Beachten Sie, dass XQuery-Dateien immer in der Textansicht angezeigt werden.

9.1.5 Informationsfenster

Informationsfenster, die während der Debugger-Sitzung im XSLT and XQuery Debugger-Anwendungsfenster geöffnet werden, enthalten Informationen über verschiedene Debugging-spezifische Aspekte der XSLT-Transformation bzw. der XQuery-Ausführung.

In XSLT-Debugger-Sitzungen stehen acht Informationsfenster zur Verfügung, in XQuery-Debugger-Sitzungen fünf. Diese Fenster sind standardmäßig in zwei Gruppen unterteilt und sind am unteren Rand der XSLT and XQuery Debugger-Oberfläche (*siehe Abbildung unten*) angeordnet. Die Fenster und die darin angezeigten Informationen werden in diesem Abschnitt näher beschrieben.



Standard-Layout der XSLT and XQuery Debugger-Oberfläche.

In der ersten Informationsfenstergruppe werden die folgenden Fenster in Form von Registerkarten in einem einzigen Fenster angezeigt:

- Kontext 551 (nur bei XSLT Debugger-Sitzungen)
- Variablen 552
- XPath-Überwachung 552

In der zweiten Informationsfenstergruppe werden die folgenden Fenster in Form von Registerkarten in einem einzigen Fenster angezeigt:

- Call Stack 553
- Meldungen 553
- Templates 554 (nur bei XSLT Debugger-Sitzungen)
- Info 552

© 2016-2022 Altova GmbH

- Ablaufverfolgung 555

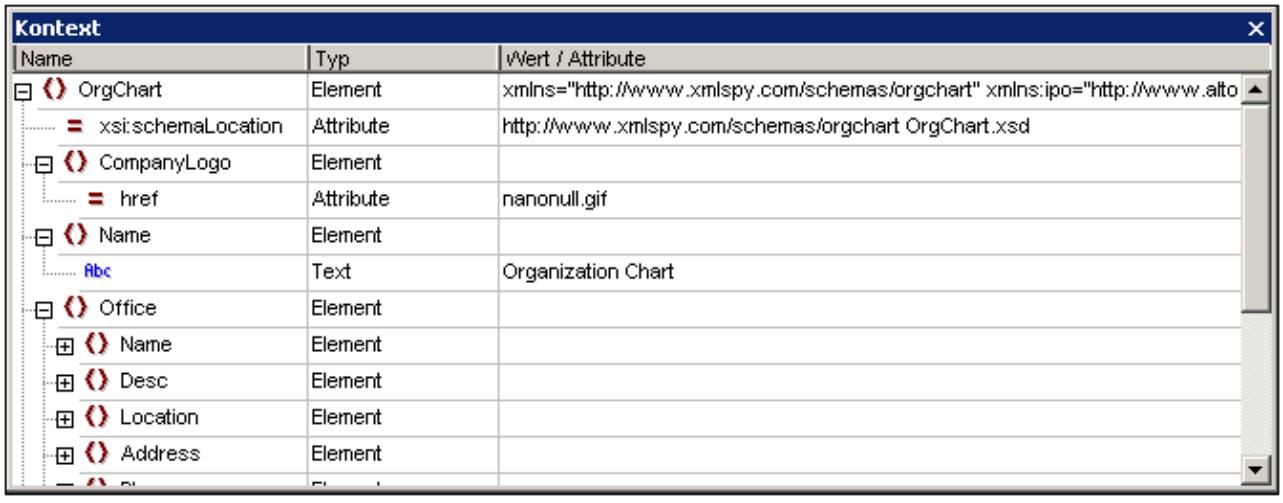
Im Standard-Layout gibt es zwei Fenstergruppen, jeweils mit Registern für die verschiedenen Fenster darin. Ein Register darin ist jeweils aktiv. Wenn Sie z.B. Informationen über Variablen anzeigen möchten, klicken Sie in der ersten Informationsfenstergruppe auf das Register "Variablen". Daraufhin wird das Variablenfenster angezeigt, während die Informationsfenster "Kontext" und "XPath-Überwachung" ausgeblendet werden. Beachten Sie, dass Sie auf einigen Registern die darin angezeigten Informationen auch zum Navigieren verwenden können: Wenn Sie auf einen Eintrag klicken, wird das jeweilige Element/Attribut in der XML-, XSLT- oder XQuery-Datei angezeigt. Nähere Informationen dazu finden Sie in der Dokumentation zu den jeweiligen Informationsfenstern (Kontext 551, Call Stack 553, Templates 554).

Durch Ziehen der Umrandung der beiden Informationsfenstergruppen können Sie deren Größe anpassen. Einzelne Fenster können aus der Gruppe herausgezogen werden, indem Sie auf den Registernamen klicken und das Fenster aus der Gruppe herausziehen. Um ein Fenster zur Gruppe hinzuzufügen, ziehen Sie seine Titelleiste auf die Titelleiste der Gruppe. Beachten Sie, dass es keine Schaltfläche zum Zurücksetzen auf das Standard-Layout gibt.

9.1.5.1 Kontext

Das Fenster "Kontext" steht nur in XSLT Debugger-Sitzungen nicht aber in XQuery Debugger-Sitzungen zur Verfügung.

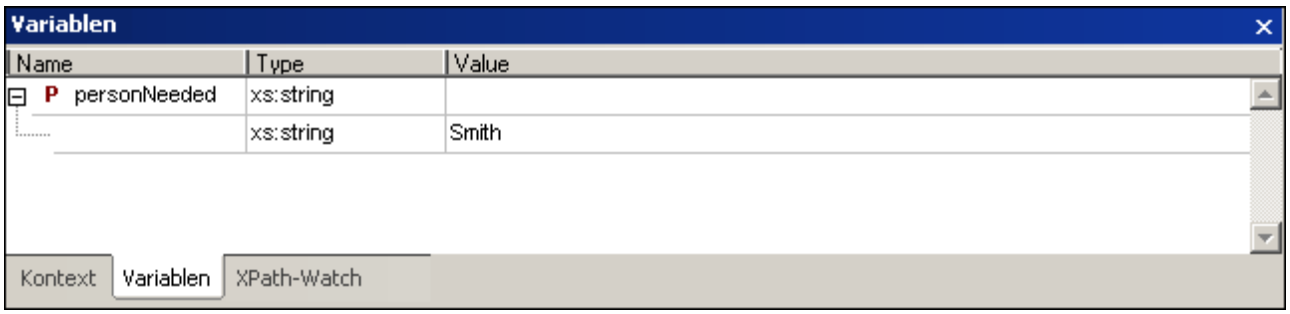
Während der Verarbeitung eines XSLT Stylesheet befindet sich der Prozessor-Kontext immer in einer Template, die mit einer Sequenz (von Nodes oder atomaren Werten) übereinstimmt. Im Kontextfenster wird der aktuelle Verarbeitungskontext angezeigt. Dabei kann es sich um eine Node-Sequenz, einen einzelnen Node oder einen atomaren Wert (wie z.B. einen String) handeln. Je nach Kontext-Eintrag, wird/werden der Wert bzw. das/die Attribut(e) angezeigt. Wenn es sich beim Kontext-Eintrag z.B. um ein Element handelt, werden die Attribute des Elements angezeigt. Wenn es sich beim Kontext-Eintrag um ein Attribut oder einen Text-Node handelt, wird der Wert des Node angezeigt.



Wenn Sie auf einen Eintrag im Kontextfenster klicken, wird diese Stelle im XML-Dokument angezeigt, Wenn das XML-Dokument derzeit auf der Benutzeroberfläche nicht angezeigt wird, wird ein Fenster dafür geöffnet.

9.1.5.2 Variablen

Das Fenster "Variablen" steht in XSLT und XQuery Debugger-Sitzungen zur Verfügung. Es werden darin die Variablen und Parameter und deren Werte angezeigt, die im XSLT/XQuery-Dokument verwendet werden, wenn Sie im Geltungsbereich liegen.

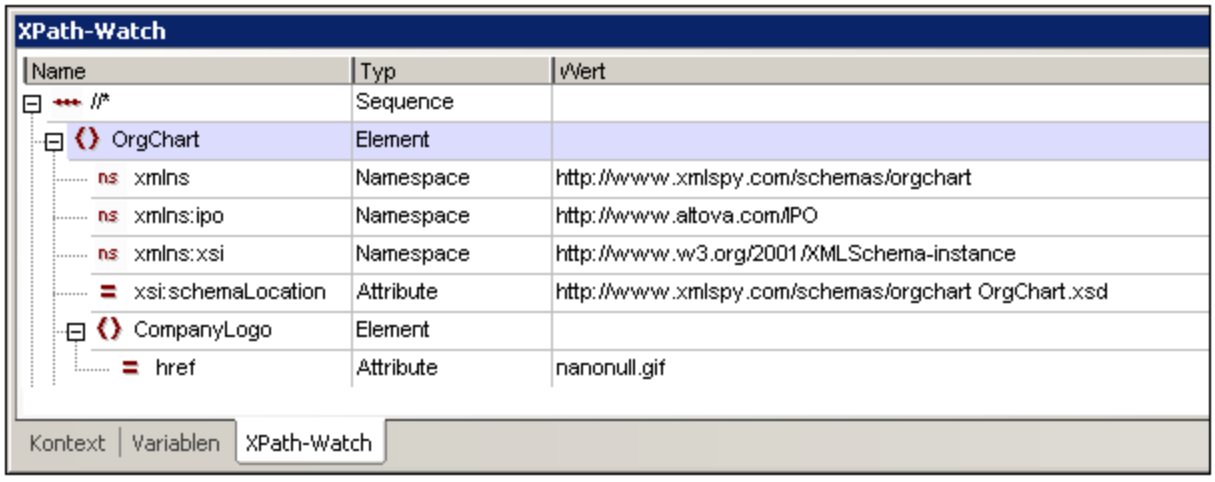


Parameter werden durch den Buchstaben **P** gekennzeichnet, globale Variablen (Variablen auf der obersten Ebene eines Stylesheet) durch ein **G** und lokale Variablen (innerhalb einer XSLT-Template deklarierte) durch ein **L**. Die Art des Werts von Variablen und Parameters wird im Feld "Value" ebenfalls durch Symbole gekennzeichnet. Es wird zwischen den folgenden Arten unterschieden: Node Set, Node Fragment, String, Number und Boolean.

9.1.5.3 XPath-Überwachung

Das Fenster "XPath-Überwachung" steht in XSLT- und XQuery-Debugger-Sitzungen zur Verfügung.

Sie können über dieses Fenster XPath-Ausdrücke eingeben, die Sie in einem oder mehreren Kontexten überprüfen möchten. Während Sie das XSLT-Dokument Schritt für Schritt überprüfen, wird der XPath-Ausdruck im aktuellen Kontext ausgewertet und das Ergebnis wird in der Spalte "Value" angezeigt.



© 2016-2022 Altova GmbH

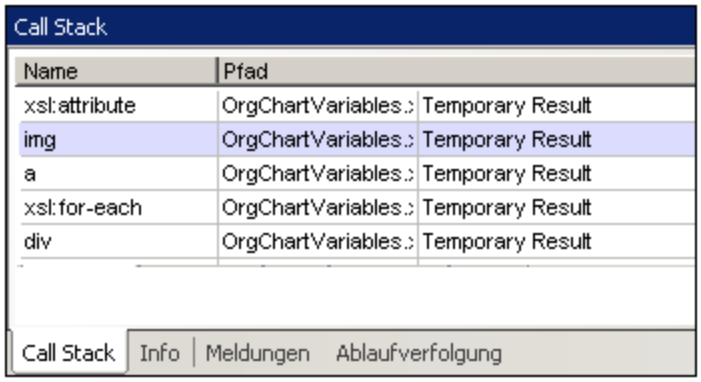
Um einen XPath-Ausdruck einzugeben, doppelklicken Sie in das Textfeld unterhalb der Spalte "Name" und geben Sie den XPath ein. Ziehen Sie alternativ dazu einen XPath-Ausdruck aus einer Datei in das Fenster "XPath-Überwachung". Verwenden Sie Ausdrücke, die gemäß der XPath-Version, die der XSLT-Version des XSLT Stylesheet entspricht (XPath 1.0 für XSLT 1.0, XPath 2.0 für XSLT 2.0 und XPath 3.0 für XSLT 3.0), korrekt sind.

Anmerkung: Wenn in der XML- bzw. XSLT-Datei Namespaces verwendet wurden, müssen Sie in Ihren XPath-Ausdrücken die richtigen Namespace-Präfixe verwenden.

9.1.5.4 Call Stack

Das Fenster "Call Stack" wird in XSLT- und XQuery Debugger-Sitzungen angezeigt.

Im Fenster "Call Stack" wird eine Liste aller zuvor verarbeiteten XSLT Templates und -Anweisungen angezeigt, wobei die aktuelle Template/Anweisung als erste in der Liste angezeigt wird.



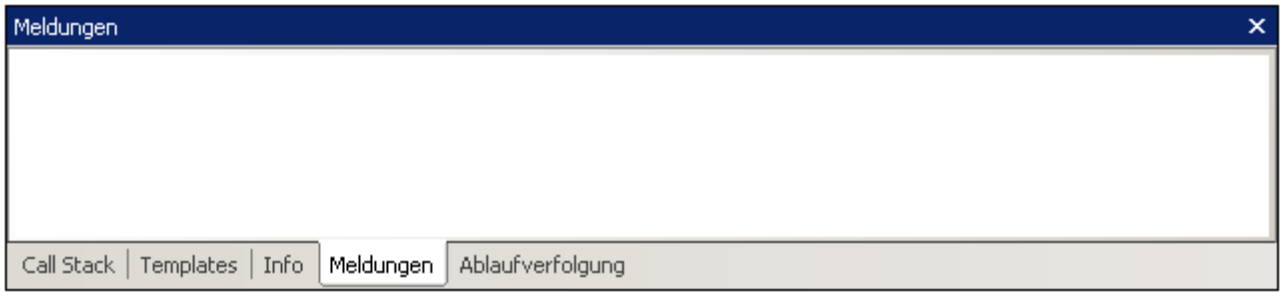
Wenn Sie in diesem Fenster auf einen Eintrag klicken, wird die jeweilige XSLT-Template/Anweisung im XSLT-Dokumentenfenster angezeigt. Wenn Sie auf eine Template/Anweisung klicken, die eine Built-in Template referenziert, wird die jeweilige built-in Template in einem separaten Fenster angezeigt, in dem alle Built-in Templates zu sehen sind.

9.1.5.5 Meldungen

Das Fenster "Meldungen" wird in XSLT- und XQuery-Debugger-Sitzungen angezeigt.

XSLT 1.0, XSLT 2.0 und XSLT 3.0

In XSLT Debugger-Sitzungen werden auf dem Register "Meldungen" Fehlermeldungen, die `xsl:message`-Anweisung(en) oder andere Fehlermeldungen, die beim Debuggen auftreten können, angezeigt.



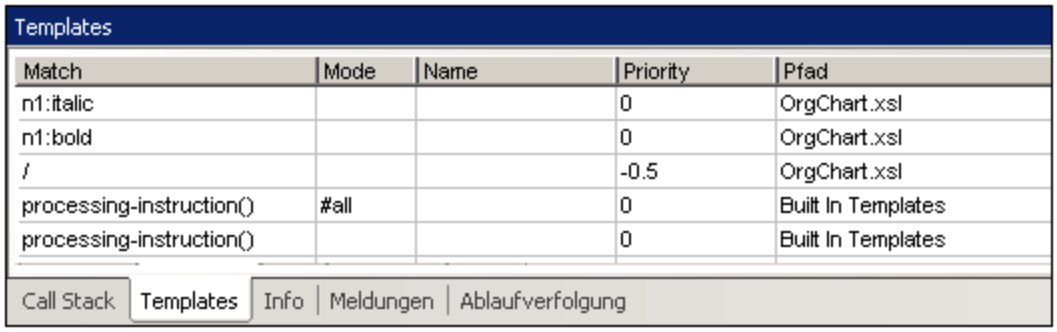
XQuery

In XQuery-Debugger-Sitzungen werden im Fenster "Meldungen" Fehlermeldungen angezeigt.

9.1.5.6 Templates

Das Fenster "Templates" (*siehe Abbildung unten*) ist nur in XSLT Debugger-Sitzungen verfügbar, nicht aber in XQuery Debugger-Sitzungen.

Im Fenster "Templates" werden die verschiedenen in XSLT Stylesheets verwendeten Templates angezeigt - einschließlich Built-in Templates und Named Templates. Matched Templates werden nach den Nodes aufgelistet, mit denen diese übereinstimmen. Named Templates werden nach Namen aufgelistet. Für beide Template-Arten werden Mode (Modus), Priority (Priorität) und Location (Pfad) angezeigt.



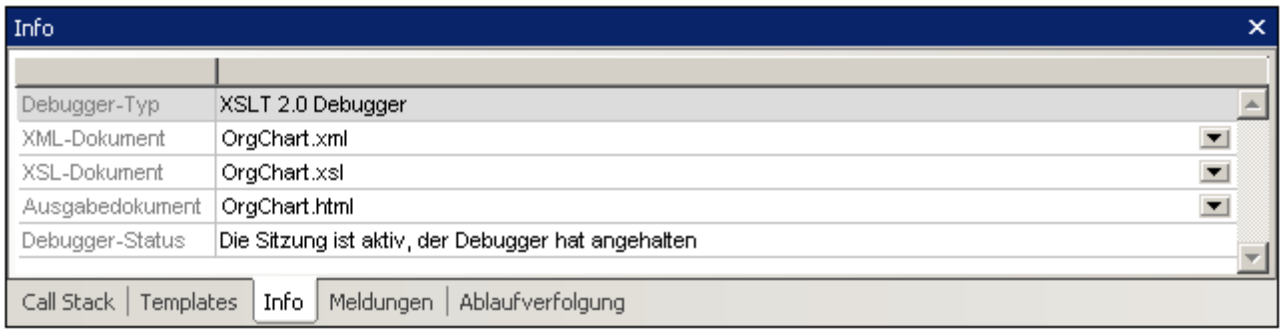
In oben gezeigten Screenshot sehen Sie drei gefundene Templates im XSLT-Stylesheet: Ein Template, das mit dem Dokument Node **/** übereinstimmt und Templates, die den Nodes `n1:italic` und `n1:bold` entsprechen. Alle anderen Templates sind Built-in Templates (die im Feld *Location* durch einen entsprechenden Eintrag gekennzeichnet sind).

Wenn Sie auf einen Eintrag in diesem Fenster klicken, wird die Template im XSLT-Dokumentenfenster markiert. Wenn Sie auf eine Built-in Template klicken, wird die Template in einem separaten Fenster, in dem alle Built-in Templates zu sehen sind, markiert.

© 2016-2022 Altova GmbH

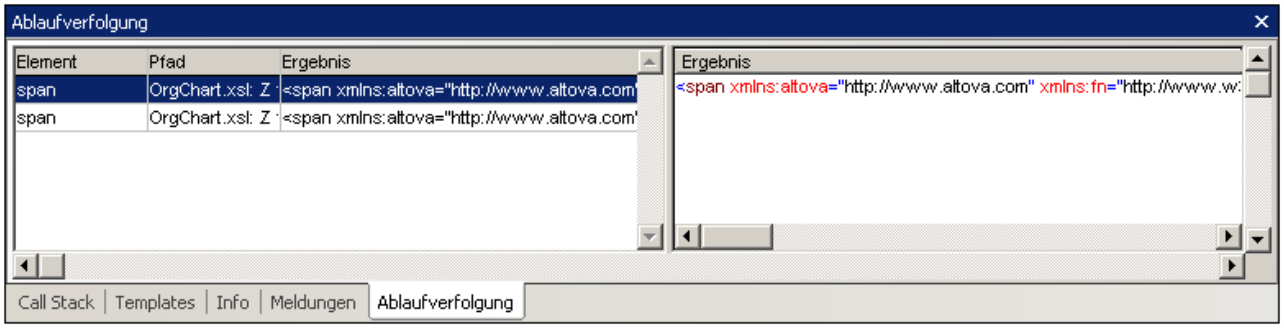
9.1.5.7 Info

Das Fenster "Info" (*siehe Abbildung unten*) ist in XSLT- und XQuery Debugger-Sitzungen verfügbar und enthält Informationen über die aktuelle Debugger-Sitzung wie u.a. Informationen zur Version des verwendeten Debuggers, die Namen des Quell- und Zieldokuments und den Status des Debuggers.



9.1.5.8 Ablaufverfolgung

Das Fenster "Ablaufverfolgung" wird in XSLT- und XQuery-Debugger-Sitzungen angezeigt.



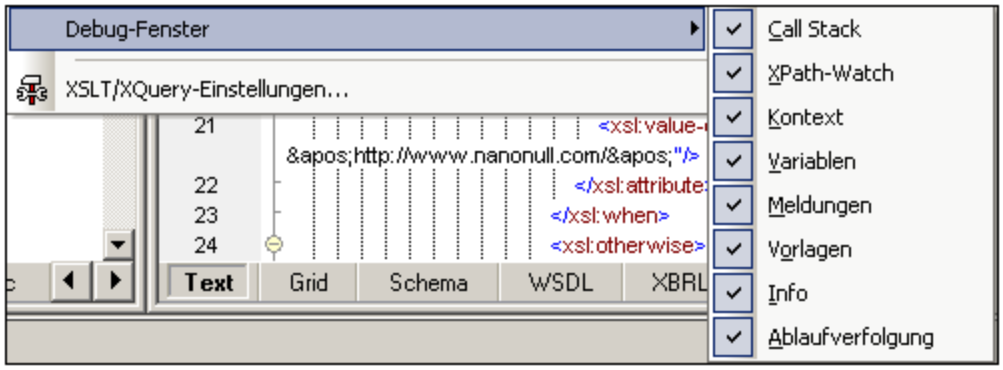
Das Fenster "Ablaufverfolgung" enthält das Element, für das der Tracepoint gesetzt wurde, seine Position im XSLT Stylesheet und das Ergebnis, das bei Ausführung des Elements generiert wird. Klicken Sie auf eine Zeile auf der linken Seite des Fensters um das vollständige Ergebnis auf der rechten Seite anzuzeigen.

9.1.5.9 Anordnen der Informationsfenster

Die Informationsfenster können an verschiedenen Stellen des XSLT and XQuery Debugger-Anwendungsfensters angeordnet werden. Fenster können angedockt werden, freischwebend angeordnet oder als mehrere Fenster in einem Fenster angezeigt werden. Zum Anordnen der Fenster stehen Ihnen die folgenden Möglichkeiten zur Verfügung.

Menü

Platzieren Sie den Cursor im Menü "XSL/XQuery" über den Eintrag "Debug-Fenster", um eine Liste von Info-Fenstern anzuzeigen. Durch Klicken auf den entsprechenden Eintrag können Sie bestimmte Fenster ein- oder ausblenden.



Damit blenden Sie das Fenster ein- bzw. aus.

Kontextmenü

Das Kontextmenü wird durch Rechtsklick auf ein Register oder die Titelleiste eines Fensters aufgerufen.



Klicken Sie auf die gewünschte Option, um das Fenster an- oder abzudocken oder auszublenden.

Drag-and-Drop

Sie können ein Fenster an seinem Register oder der Titelleiste an die gewünschte Stelle ziehen.

Außerdem können Sie das Fenster beim Ziehen mit Hilfe von Steuerelementen zur Platzierung an einem anderen Fenster oder im Anwendungsfenster andocken. Diese Ziehpunkte werden angezeigt, wenn Sie ein Fenster ziehen:

- Wenn Sie ein Fenster über ein anderes Fenster ziehen, erscheint ein kreisförmiges Steuerelement, das aus fünf Sektoren besteht (*siehe Screenshot unten*). Wenn Sie die Maustaste über einem dieser Sektoren loslassen, wird das Fenster, das Sie gerade gezogen haben, am entsprechenden Bereich des **Zielfensters** angedockt. Mit Hilfe der vier Pfeilsektoren wird das Fenster an der entsprechenden Seite des Zielfensters angedockt. Mit Hilfe der Schaltfläche in der Mitte wird das Fenster als Register des Zielfensters angedockt. Sie können ein Fenster auch als Register in einem anderen Fenster andocken, indem Sie es in die Registerleiste ziehen.

© 2016-2022 Altova GmbH



- Beim Ziehen eines Fensters erscheint ein Steuerelement bestehend aus vier Pfeilen. Jeder Pfeil entspricht einer Seite der **Debugger-Oberfläche**. Wenn Sie die Maustaste beim Ziehen eines Fensters über einem dieser Pfeile loslassen, wird das Fenster an einer Seite der Debugger-Oberfläche angedockt.



Außerdem können Sie zwischen der an- und der abgedockten Position eines Fensters wechseln, indem Sie auf die Titelleiste des Fensters doppelklicken.

9.1.6 Breakpoints

Der XSLT and XQuery Debugger gestattet Ihnen, Breakpoints in XSLT-, XQuery- und XML-Dokumenten zu definieren. Breakpoints werden als strichlierte rote Linie angezeigt (*siehe Abb. unten*).

Anmerkung: Sie können für dieselbe Anweisung einen Tracepoint und einen Breakpoint setzen. Angezeigt wird dies durch eine strichlierte blaurote Linie (*siehe Screenshot*).

```
<!– Local Variable reused –>
<xsl:variable name="OfficeName" select="n1:Name"/>
<!– Display the company name if the varaible is true ********* –>
<xsl:if test="$Show_Company_Name">
    <h3>
        <xsl:value-of select="$OfficeName"/>
    </h3>
    <xsl:message>
        <xsl:text>Company Named Displayed</xsl:text>
    </xsl:message>
</xsl:if>
```

Wenn Sie den Debugger in einer Debugger-Sitzung starten, wird der Debugger an jedem Breakpoint angehalten. Auf diese Art können Sie bestimmte Bereiche, auf die Sie Ihre Aufmerksamkeit im XSLT-, XQuery- und/oder XML-Dokument richten möchten, für das Debuggen kennzeichnen. Sie können beliebig viele Breakpoints definieren.

Anmerkung: Die für ein Dokument gesetzten Breakpoints bleiben erhalten, bis das Dokument geschlossen wird. Wenn Sie jedoch in die Schema-Ansicht wechseln (z.B: bei XSD-Dokumenten), werden die Breakpoints gelöscht; wenn Sie (von der Schema-Ansicht) in die Text- oder Grid-Ansicht zurückwechseln, gibt es dort keinen Breakpoint mehr.

Breakpoints in XML-Dokumenten

Sie können an jedem Node in einem XML-Dokument einen Breakpoint setzen. Die Verarbeitung wird am Beginn des Node angehalten.

Breakpoints in XSLT-Dokumenten

Sie können an den folgenden Stellen in einem XSLT-Dokument Breakpoints setzen:

- am Anfang von Templates und Template Instructions (z.B. `xsl:for-each`).
- an einem XPath-Ausdruck (XPath 1.0 oder XPath 2.0).
- an jedem Node in einem literal erstellten XML-Fragment. Die Verarbeitung wird am Beginn des Node unterbrochen.

Breakpoints in XQuery-Dokumenten

Sie können an den folgenden Stellen in einem XQuery-Dokument Breakpoints setzen:

- am Beginn von XQuery-Anweisungen.
- in einem XQuery-Ausdruck.
- an jedem Node in einem hartcodierten XML-Fragment. Die Verarbeitung wird am Beginn des Node unterbrochen.

Einfügen/Löschen von Breakpoints

So fügen Sie einen Breakpoint ein:

1. Setzen Sie den Cursor an die Stelle im Dokument, an der der Breakpoint gesetzt werden soll (*siehe oben*). In XSLT Debugger-Sitzungen können Sie sowohl in der Text- als auch in der Grid-Ansicht Breakpoints setzen. XQuery-Debugger-Sitzungen stehen nur in der Textansicht zur Verfügung.
2. Wählen Sie eine der folgenden Methoden:

- Wählen Sie **XSL/XQuery | Breakpoint einfügen/löschen**.
- Drücken Sie die Taste **F9**.
- Klicken Sie auf die rechte Maustaste und wählen Sie **Breakpoint einfügen/löschen**.

So löschen Sie einen Breakpoint:

1. Setzen Sie den Cursor an die Stelle im Dokument, die den Breakpoint enthält.
2. Wählen Sie eine der folgenden Methoden:

- Wählen Sie **XSL/XQuery | Breakpoint einfügen/löschen**.
- Drücken Sie die Taste **F9**.
- Klicken Sie auf die rechte Maustaste und wählen Sie **Breakpoint einfügen/löschen**.

Alternativ dazu können Sie einen Breakpoint auch über das Dialogfeld "Breakpoints" löschen:

© 2016-2022 Altova GmbH

1. Wählen Sie die Menüoption **XSL/XQuery | Breakpoints...**.
2. Klicken Sie im Dialogfeld auf den Breakpoint und klicken Sie auf **Löschen**.

Über die Schaltfläche **Alle löschen** löschen Sie alle Breakpoints aus dem Dialogfeld (und allen XSLT Stylesheets).

Deaktivieren/Aktivieren von Breakpoints:

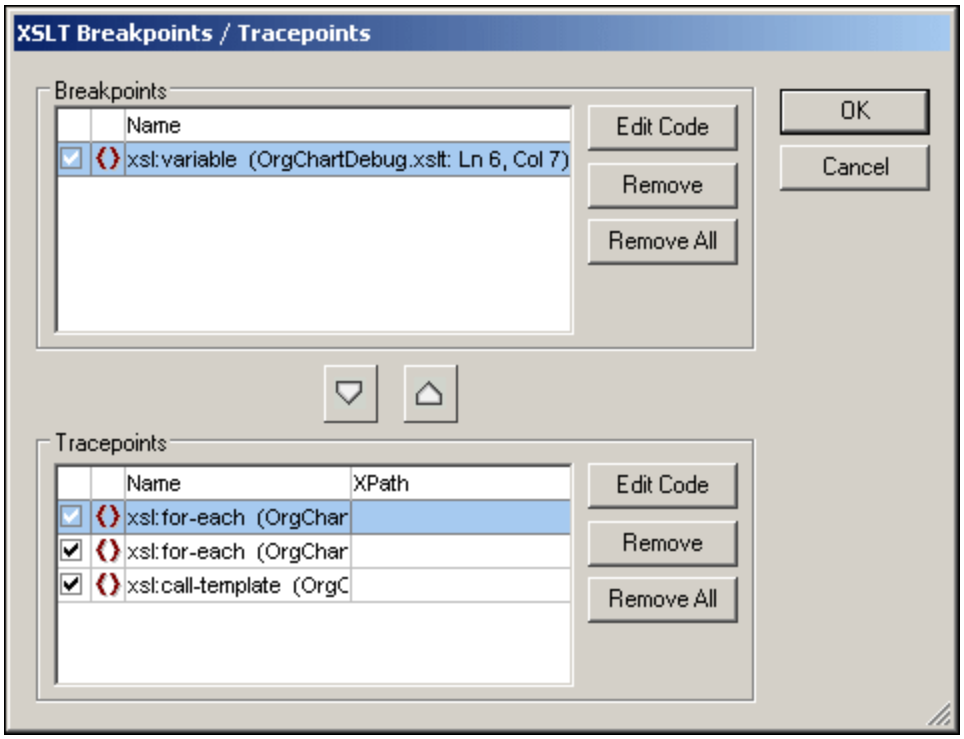
Sie können Breakpoints, die übersprungen werden sollen, auch deaktivieren ohne den Breakpoint zu löschen. Sie können diese Breakpoints später auch wieder aktivieren.

So deaktivieren Sie einen Breakpoint:

1. Setzen Sie den Cursor in den Node bzw. den Ausdruck, der den Breakpoint enthält.
2. Wählen Sie den Befehl **XSL/XQuery | Breakpoint aktivieren/deaktivieren** (Befehl kann auch mit Hilfe der Tastenkombination **Strg+F9** aufgerufen werden). Der Breakpoint ändert sich von rot in grau, was bedeutet, dass er deaktiviert wurde.

Alternativ dazu können Sie Breakpoints auch über das Dialogfeld "Breakpoints" deaktivieren. Gehen Sie dabei vor wie folgt:

1. Wählen Sie den Menübefehl **XSL/XQuery | Breakpoints/Tracepoints...**. Daraufhin wird das Dialogfeld "XSLT Breakpoints" geöffnet, in dem die aktuell definierten Breakpoints in allen XML-Quell- und XSLT Stylesheet-Dokumenten angezeigt werden.



2. Entfernen Sie das Häkchen bei den Breakpoints, die Sie deaktivieren möchten und klicken Sie auf OK. Die entsprechenden Breakpoints werden nun statt rot grau angezeigt.

So aktivieren Sie einen Breakpoint:

1. Setzen Sie den Cursor in den Node bzw. Ausdruck, der den Breakpoint enthält.
2. Wählen Sie den Befehl **XSL/XQuery | Breakpoint ein/aus** (oder drücken Sie **Strg+F9**). Die Farbe des Breakpoints ändert sich von grau auf rot, d.h. der Breakpoint wurde aktiviert.

Suchen eines bestimmten Breakpoint

So suchen Sie nach einem bestimmten Breakpoint:

1. Wählen Sie die Menüoption **XSL/XQuery | Breakpoints/Tracepoints...**. Daraufhin wird das Dialogfeld "XSLT Breakpoints/Tracepoints" angezeigt.
2. Klicken Sie auf den gewünschten Breakpoint in der Liste der Breakpoints.
3. Klicken Sie auf die Schaltfläche **Code bearbeiten**. Das Dialogfeld "Breakpoints" wird geschlossen und der Textcursor wird in der Textansicht direkt vor den Breakpoint gesetzt. In der Enhanced Grid-Ansicht wird die Tabellenzelle mit dem Breakpoint rot markiert.

Fortsetzen der Debugger-Sitzung nach dem Breakpoint

So fahren Sie nach dem Breakpoint mit der Debugger-Sitzung fort:

- Wählen Sie den Befehl **XSL/XQuery | Einsteigen** oder **XSL/XQuery | Debugger starten**.

9.1.7 Tracepoints

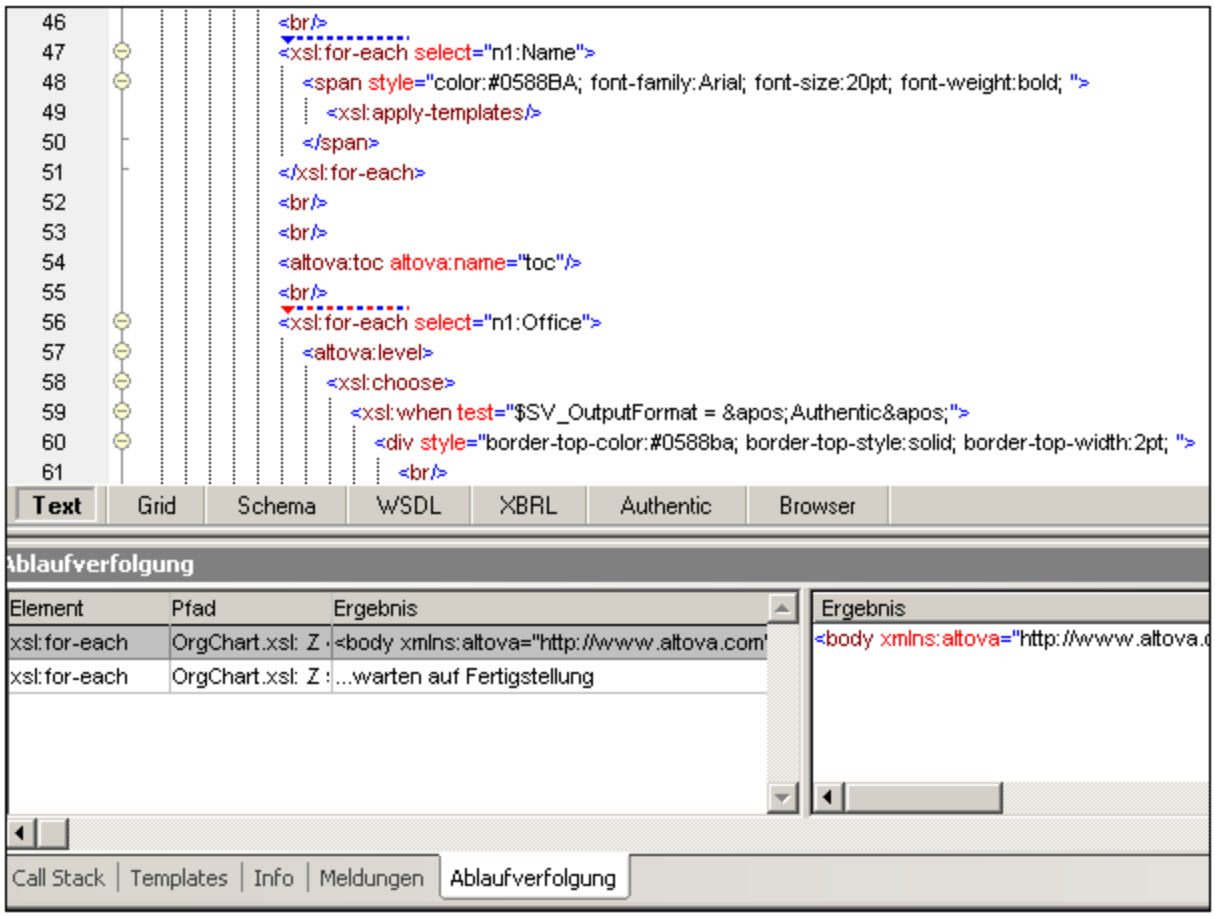
Mit Hilfe des XSLT and XQuery Debugger können Sie Tracepoints in XSLT-Dokumenten definieren.

Mit Hilfe von Tracepoints können Sie Inhalt, der von einer Anweisung generiert wird, auffinden oder das Ergebnis eines XPath-Ausdrucks an dem Punkt anzeigen, an dem der Tracepoint gesetzt wurde, ohne das XSLT Stylesheet z.B. durch Verwendung des `xsl:message` Elements zur Ausgabe von Debugging-Meldungen bearbeiten zu müssen.

Tracepoints werden durch eine strichlierte blaue Linie im Code des XSLT Stylesheet angezeigt (*siehe Screenshot unten*).

Anmerkung: Sie können innerhalb derselben Anweisung einen Tracepoint und einen Breakpoint setzen. Dies wird in Form einer blaurot gestrichelten Linie dargestellt (*siehe Screenshot*).

© 2016-2022 Altova GmbH



Der Debugger gibt die Ergebnisse der einzelnen Anweisungen, in denen ein Tracepoint gesetzt wurde, aus. Dieses Ergebnis wird im Fenster "Ablaufverfolgung" angezeigt. Sie können in einem XSLT Stylesheet beliebig viele Tracepoints setzen.

Anmerkung: Die Tracepoints bleiben in einem Dokument erhalten, bis es geschlossen wird.

Tracepoints in XSLT-Dokumenten

Sie können in einem XSLT Stylesheets in XSLT-Anweisungen und literalen Ergebnissen Tracepoints setzen.

Tracepoints in XML- und XQuery-Dokumenten

Sie können in XML- und XQuery-Dokumenten Tracepoints setzen.

Einfügen/Löschen von Tracepoints

So fügen Sie einen Tracepoint ein:

1. Platzieren Sie den Cursor an dem Punkt im XSLT-Dokument, an dem Sie den Tracepoint einfügen möchten. Sie können Tracepoints sowohl in der Text- als auch in der Grid-Ansicht setzen.
2. Wählen Sie eine der folgenden Methoden:

- Wählen Sie **XSL/XQuery | Tracepoint einfügen/löschen**.
- Drücken Sie **Umschalt+F9**.
- Rechtsklicken Sie und wählen Sie den Befehl **Tracepoint einfügen/löschen.**

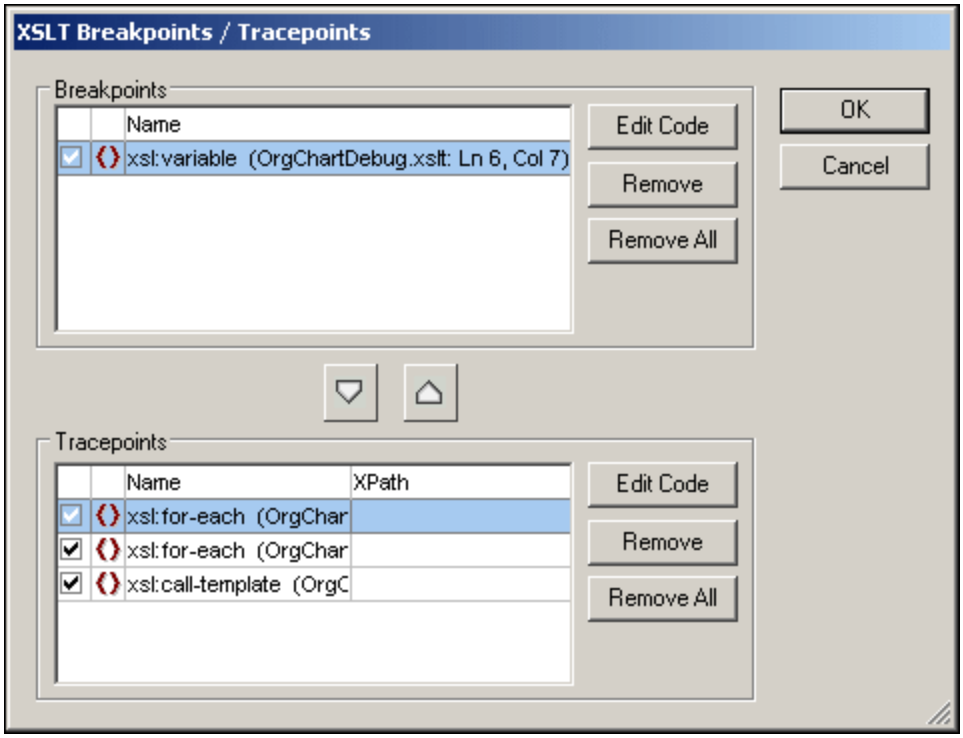
So löschen Sie einen Tracepoint:

1. Platzieren Sie den Cursor an die Stelle im XSLT-Dokument, die den Tracepoint enthält.
2. Wählen Sie eine der folgenden Methoden:

   - Wählen Sie **XSL/XQuery | Tracepoint einfügen/löschen**.
   - Drücken Sie **Umschalt+F9**.
   - Rechtsklicken Sie und wählen Sie den Befehl **Tracepoint einfügen/löschen**.

Alternativ dazu können Sie Tracepoints auch über das Dialogfeld "XSLT Breakpoints / Tracepoints" löschen:

1. Wählen Sie die Menüoption **XSL/XQuery | Breakpoints/Tracepoints...**.
2. Klicken Sie auf den Tracepoint im Dialogfeld und klicken Sie auf **Löschen**.



Mit Hilfe der Schaltfläche **Alle löschen** löschen Sie die Tracepoints aus dem Dialogfeld (und allen XSLT Stylesheets).

Definieren eines XPath als Tracepoint

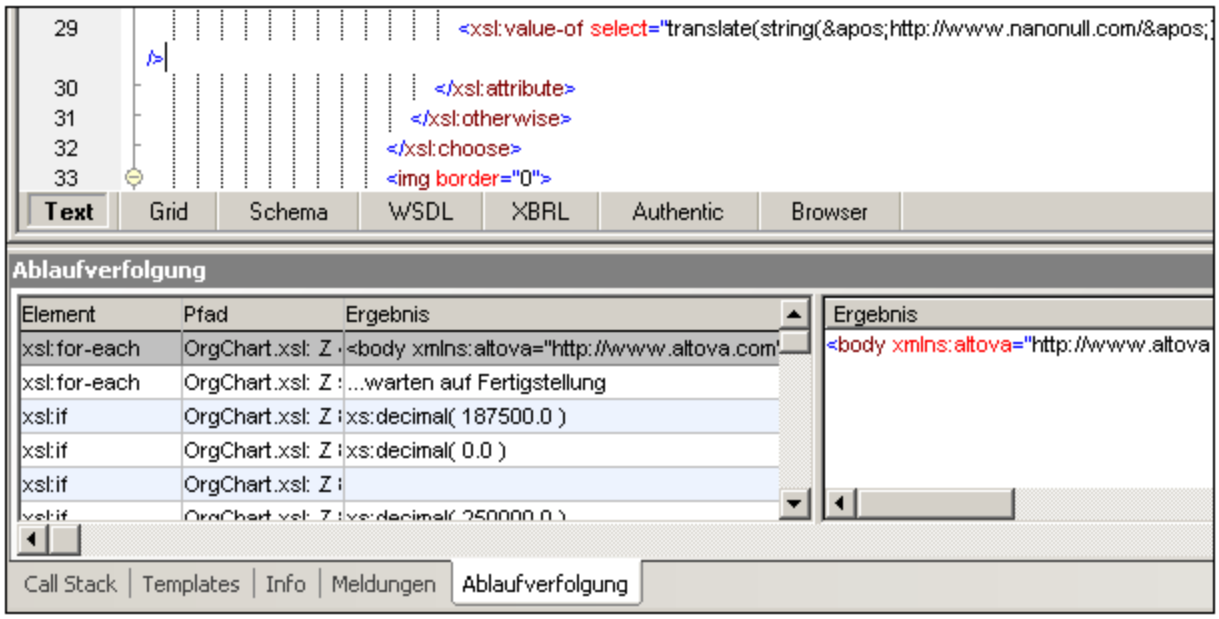
Sie können einen XPath als Tracepoint definieren. Wenn Sie einen XPath als Tracepoint definieren, wird im Fenster "Ablaufverfolgung" anstelle des Inhalts, der durch die Anweisung, für die der Tracepoint gesetzt wurde, generiert wird, das Ergebnis der XPath-Auswertung angezeigt. Der XPath wird relativ zum Kontext-Node an der Stelle, an der der Tracepoint gesetzt wurde, ausgewertet.

© 2016-2022 Altova GmbH

So definieren Sie einen XPath als Tracepoint:

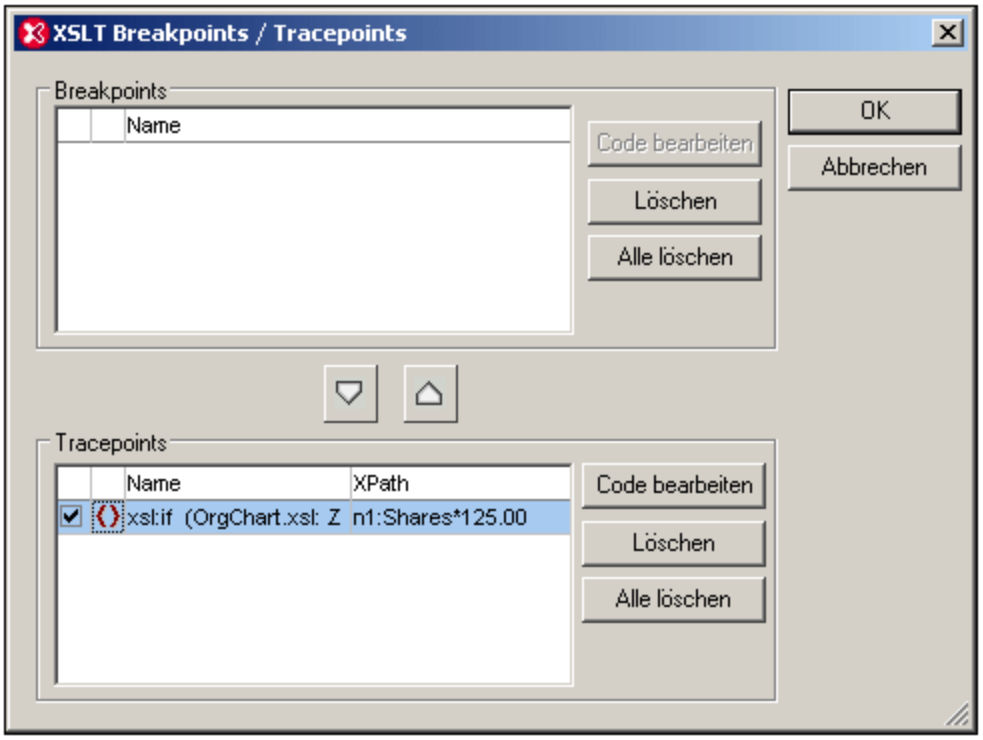
1. Wählen Sie die Menüoption **XSL/XQuery | Breakpoints/Tracepoints...**. Daraufhin wird das Dialogfeld "XSLT Breakpoints / Tracepoints" geöffnet, in dem alle derzeit definierten Tracepoints in allen offenen XSLT Stylesheets angezeigt werden.
2. Geben Sie den XPath in die Spalte **XPath** in der Zeile für den entsprechenden Tracepoint ein.

Beispiel:
Im unten gezeigten Beispiel wird der Tracepoint so gesetzt, dass der Kontext Node "Person" ist. Das Element "Person" enthält ein Element "Shares". Wir möchten die Anzahl der Aktien (Shares), die jede Person hat, multipliziert mit 125 (dem Wert einer einzelnen Aktie) anzeigen.



Gehen Sie folgendermaßen vor:

1. Setzen Sie in der Zeile direkt nach der `xsl:for-each`-Anweisung, die das Element `n1:Person` auswählt (in der Abbildung oben in Zeile 555) einen Tracepoint.
2. Öffnen Sie das Dialogfeld "XSLT Breakpoints / Tracepoints" und geben Sie den XPath `n1:Shares*125.00` für diesen Tracepoint ein.

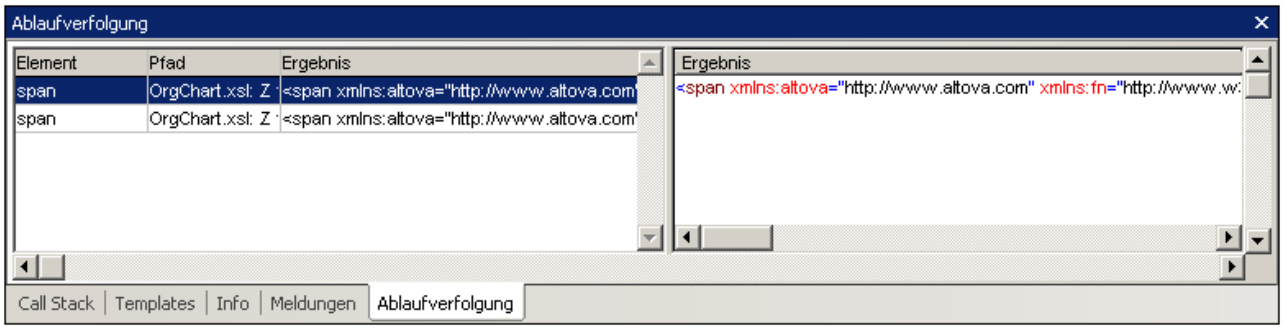


3. Starten Sie den Debugger. Das Ergebnis des XPath, den Sie als Tracepoint eingegeben haben, wird im Fenster "Ablaufverfolgung" angezeigt.

Das Fenster "Ablaufverfolgung"

Wählen Sie den Befehl **XSL/XQuery | Debugger starten,** um den Debugger zu starten. Das Ergebnis der Anweisungen, für die Tracepoints gesetzt wurden, wird im Fenster Ablaufverfolgung[555] (*siehe Screenshot*) angezeigt. Klicken Sie auf eine Zeile im Fenster "Ablaufverfolgung", um das vollständige Ergebnis der Anweisung auf der rechten Seite des Fensters (*siehe Screenshot*) anzuzeigen.

Anmerkung: Die Ergebnisse werden erst dann im Fenster "Ablaufverfolgung" angezeigt, wenn diese Ablaufverfolgung-Anweisung fertig ausgeführt wurde.



 © 2016-2022 Altova GmbH

Aktivieren/Deaktivieren von Tracepoints:

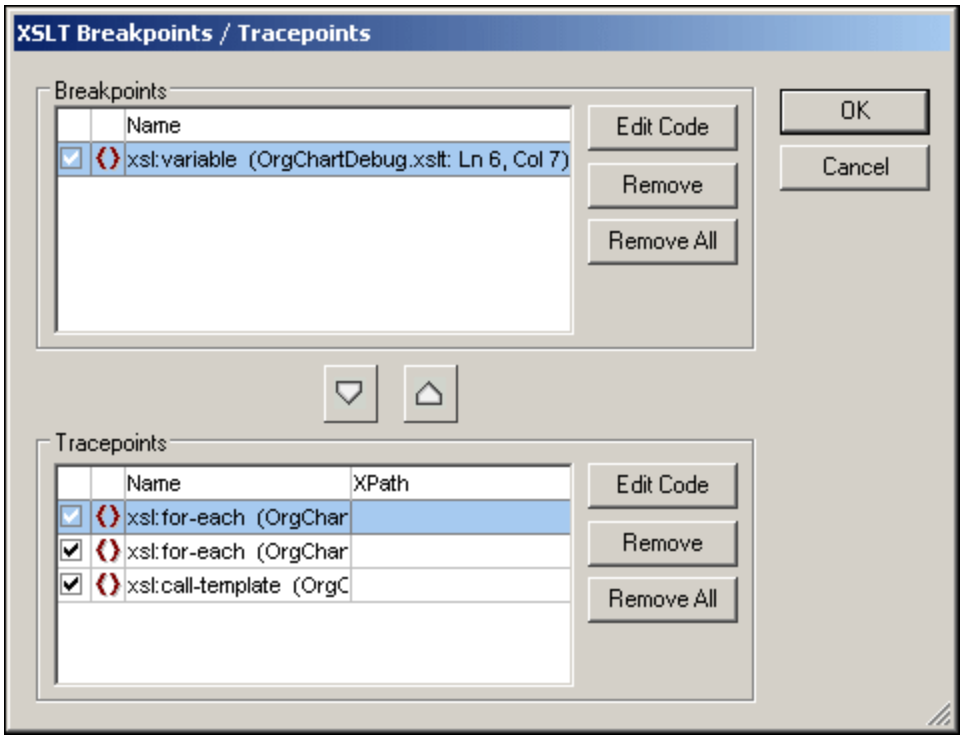
Sie können Tracepoints, nachdem Sie sie eingefügt haben, deaktivieren, wenn diese übersprungen aber nicht gelöscht werden sollen. Bei Bedarf können Sie diese jederzeit wieder aktivieren.

So deaktivieren Sie einen Tracepoint:

1. Platzieren Sie den Cursor an die Stelle im XSLT Stylesheet, die den Tracepoint enthält.
2. Wählen Sie **XSL/XQuery | Tracepoint aktivieren/deaktivieren** (oder drücken Sie **Strg+F9**). Die Farbe des Tracepoints ändert sich von Blau in Grau, wodurch gekennzeichnet wird, dass er deaktiviert wurde.

Alternativ dazu können Sie einen Breakpoint über das Dialogfeld "XSLT Breakpoints / Tracepoints" deaktivieren:

1. Wählen Sie die Menüoption **XSL/XQuery | Breakpoints/Tracepoints...**. Daraufhin wird das Dialogfeld XSLT Breakpoints / Tracepoints geöffnet, in dem Sie die aktuell definierten Breakpoints in allen offenen XML-Quelldokumenten und XSLT Stylesheets sehen.



2. Entfernen Sie das Häkchen für die Tracepoints, die Sie deaktivieren möchten und klicken Sie zur Bestätigung auf **OK**. Diese Tracepoints werden nun statt blau grau angezeigt, d.h. sie sind deaktiviert.

So aktivieren Sie einen Tracepoint:

1. Platzieren Sie den Cursor in den Node oder Ausdruck, der den Tracepoint enthält.
2. Wählen Sie **XSL/XQuery | Tracepoint aktivieren/deaktivieren** (oder drücken Sie **Strg+F9**). Der Tracepoint wird nun statt grau wieder blau angezeigt, was bedeutet, dass er aktiviert wurde.

Suchen eines bestimmten Tracepoint

So suchen Sie einen bestimmten Tracepoint:

1. Wählen Sie die Menüoption **XSL/XQuery | Breakpoints/Tracepoints...**. Daraufhin wird das Dialogfeld "XSLT Breakpoints / Tracepoints" angezeigt.
2. Klicken Sie auf den gewünschten Tracepoint in der Liste.
3. Klicken Sie auf die Schaltfläche **Code bearbeiten**. Das Dialogfeld "XSLT Breakpoints / Tracepoints" wird geschlossen und der Textcursor erscheint direkt vor dem Tracepoint in der Textansicht des XSLT-Dokuments. In der Enhanced Grid-Ansicht wird die Tabellenzelle, die den Breakpoint enthält, rot markiert.

9.1.8 Debugger-Tastaturkürzel

In der Liste unten finden Sie die Tastaturkürzel für den Debugger.

- *Debugger-Befehle*

F9	Breakpoint einfügen/löschen
F9 + Umschalt	Tracepoint einfügen/löschen
F9 + STRG	Breakpoint aktivieren/deaktivieren
F9 + Umschalt + STRG	Tracepoint aktivieren/deaktivieren
F11	Einsteigen
F11 + Umschalt	Aussteigen
F11 + STRG	Überspringen
F11 + Alt	Debugger starten

© 2016-2022 Altova GmbH

9.2 XSLT/XQuery Profiler

Altova Website: XSLT Profiler, XQuery Profiler
Der XSLT/XQuery Profiler ist ein Tool zum Analysieren der Ausführungszeiten von XSLT (1.0 und 2.0) Stylesheets und XQuery-Dokumenten innerhalb von XMLSpy. Er gibt Auskunft darüber, wie lange die Ausführung der einzelnen Anweisungen in einem XSLT Stylesheet oder einem XQuery-Dokument jeweils dauert. Anhand dieser Informationen können Sie die Ausführungsdauer für diese Dateien optimieren.

Mit Hilfe des Profilers können Sie die Anweisungen ermitteln, deren Ausführung insgesamt am längsten dauert. Anhand dieser Informationen können Sie die Anweisungen optimieren. Die Ausführung mancher Anweisungen kann aus einem oder mehreren der folgenden Gründe lange dauern:

- die Anweisung ist zeitintensiv
- die Anweisung wird oft überprüft (hohe Trefferquote)

Profilerstellung für Trefferquote und Callgraph

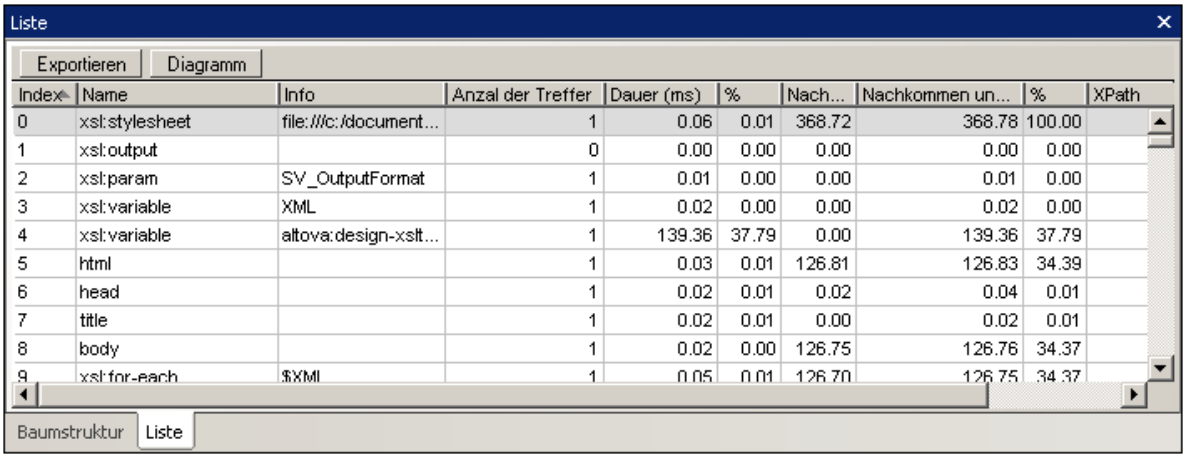
Sie können im Profiler zwischen *Trefferquote* und *Callgraph* Profilerstellung auswählen. Bei beiden Profilerstellungsarten wird eine Statistik zur Ausführungszeit der einzelnen Anweisungen angezeigt.
Zu Optimierungszwecken wird normalerweise die Trefferquote-Profilerstellung verwendet. Dabei wird für jede Anweisung im Profiler eine Zeile angezeigt. Bei der Callgraph-Profilerstellung sehen Sie eine Aufzeichnung des Verlaufs der Ausführung der gesamten XSLT-Transformation oder XQuery-Ausführung. Hier sehen Sie, welche Templates/Funktionen während der Transformation aufgerufen werden und in welcher Reihenfolge dies geschieht. Im Ergebnis der Callgraph Profilerstellung wird jedesmal, wenn eine Anweisung aufgerufen wird, eine Zeile angezeigt, anstatt dass für jede Anweisung eine Zeile angezeigt wird.

Informationen zur Verwendung des XSLT/XQuery Profilers finden Sie unter Erstellen von XSLT-Profilen[572] oder Erstellen von XQuery-Profilen[576].

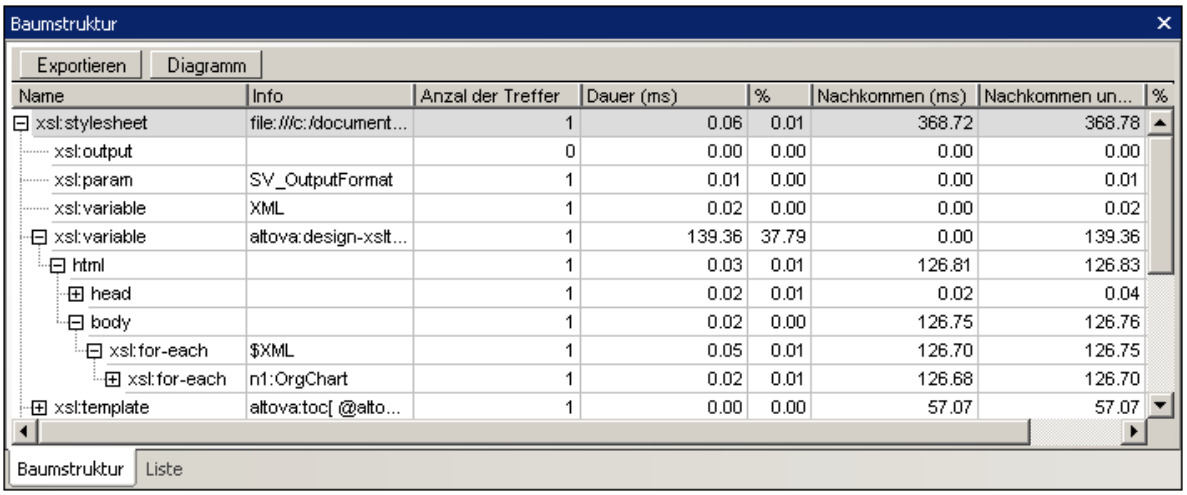
Profiler-Ansichten

Die Ergebnisse der Analyse können in einer der folgenden Ansichten angezeigt werden. Klicken Sie dazu auf das entsprechende Register:

- **Listenansicht:** Die Statistik wird in Form einer Liste angezeigt, die z.B. nach Dauer der Anweisungsausführung oder der Dauer der Ausführung der Anweisungen und ihrer untergeordneten Nodes (Nachkommen) sortiert werden kann.



- **Baumstrukturansicht:** Die Statistik wird in Form einer Baumstruktur angezeigt. Sie sehen darin z.B. wie lange für die Ausführung einer Funktion gebraucht wurde. Sie können die Struktur für diese Funktion anschließend erweitert anzeigen lassen und überprüfen, wie viel Zeit die Ausführung der einzelnen Anweisungen in Anspruch nahm und wie oft sie ausgeführt wurden.



Name	Info	Anzal der Treffer	Dauer (ms)	%	Nachkommen (ms)	Nachkommen un...	%
xsl:stylesheet	file:///c:/document...	1	0.06	0.01	368.72	368.78	
xsl:output		0	0.00	0.00	0.00	0.00	
xsl:param	SV_OutputFormat	1	0.01	0.00	0.00	0.01	
xsl:variable	XML	1	0.02	0.00	0.00	0.02	
xsl:variable	altova:design-xslt...	1	139.36	37.79	0.00	139.36	
html		1	0.03	0.01	126.81	126.83	
head		1	0.02	0.01	0.02	0.04	
body		1	0.02	0.00	126.75	126.76	
xsl:for-each	$XML	1	0.05	0.01	126.70	126.75	
xsl:for-each	n1:OrgChart	1	0.02	0.01	126.68	126.70	
xsl:template	altova:toc[@alto...	1	0.00	0.00	57.07	57.07	

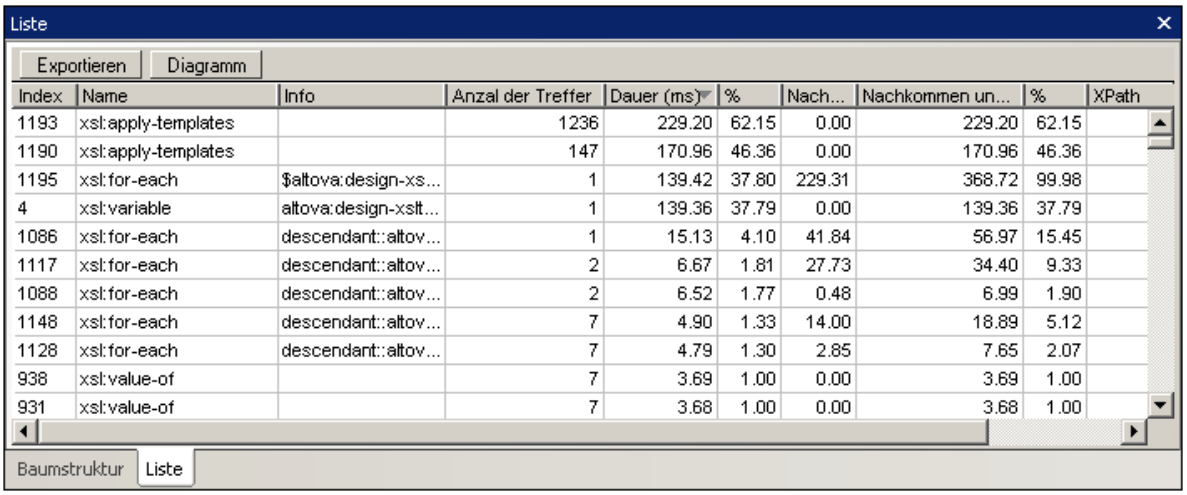
Sortieren von Ergebnissen

Nach Ausführung des Profilers können Sie die Ergebnisse nach der Ausführungsdauer der einzelnen Anweisungen oder nach der Anzahl der Aufrufe der Anweisung sortieren.

So sortieren Sie die Informationen im Profiler:

1. Klicken Sie auf das Register **Liste**.
2. Klicken Sie auf den Spaltenkopf der Spalte, nach der die Sortierung durchgeführt werden soll (z.B., **Anzahl der Treffer,** um nach der Anzahl der Aufrufe einer Anweisung zu sortieren oder **Dauer,** um nach der Ausführungsdauer der Anweisungen zu sortieren).

In diesem Screenshot sehen Sie den Inhalt des Profilers nach Anweisungsdauer in absteigender Reihenfolge sortiert.



Index	Name	Info	Anzal der Treffer	Dauer (ms)	%	Nach...	Nachkommen un...	%	XPath
1193	xsl:apply-templates		1236	229.20	62.15	0.00	229.20	62.15	
1190	xsl:apply-templates		147	170.96	46.36	0.00	170.96	46.36	
1195	xsl:for-each	$altova:design-xs...	1	139.42	37.80	229.31	368.72	99.98	
4	xsl:variable	altova:design-xslt...	1	139.36	37.79	0.00	139.36	37.79	
1086	xsl:for-each	descendant::altov...	1	15.13	4.10	41.84	56.97	15.45	
1117	xsl:for-each	descendant::altov...	2	6.67	1.81	27.73	34.40	9.33	
1088	xsl:for-each	descendant::altov...	2	6.52	1.77	0.48	6.99	1.90	
1148	xsl:for-each	descendant::altov...	7	4.90	1.33	14.00	18.89	5.12	
1128	xsl:for-each	descendant::altov...	7	4.79	1.30	2.85	7.65	2.07	
938	xsl:value-of		7	3.69	1.00	0.00	3.69	1.00	
931	xsl:value-of		7	3.68	1.00	0.00	3.68	1.00	

© 2016-2022 Altova GmbH

Optimieren Ihrer XSLT Stylesheets und XQuery-Dokumente

Beachten Sie beim Optimieren der Ausführungsdauer von Anweisungen in XSLT Stylesheets und XQuery-Dokumenten die folgenden Richtlinien:

- Verwenden Sie in einer Anweisung nach Möglichkeit keine Variablen, wenn die Variable nur einmal verwendet wird, da die Initialisierung einer Variable zeitaufwändig sein kann.

  In den folgenden XSLT Code-Fragmenten sehen Sie ein Beispiel, wie man Code durch Entfernen unnötiger Variablen optimieren kann. Beide Beispiele haben dasselbe Ergebnis. Im zweiten Beispiel werden jedoch die Variablen `name` und `containsResult` nicht verwendet:

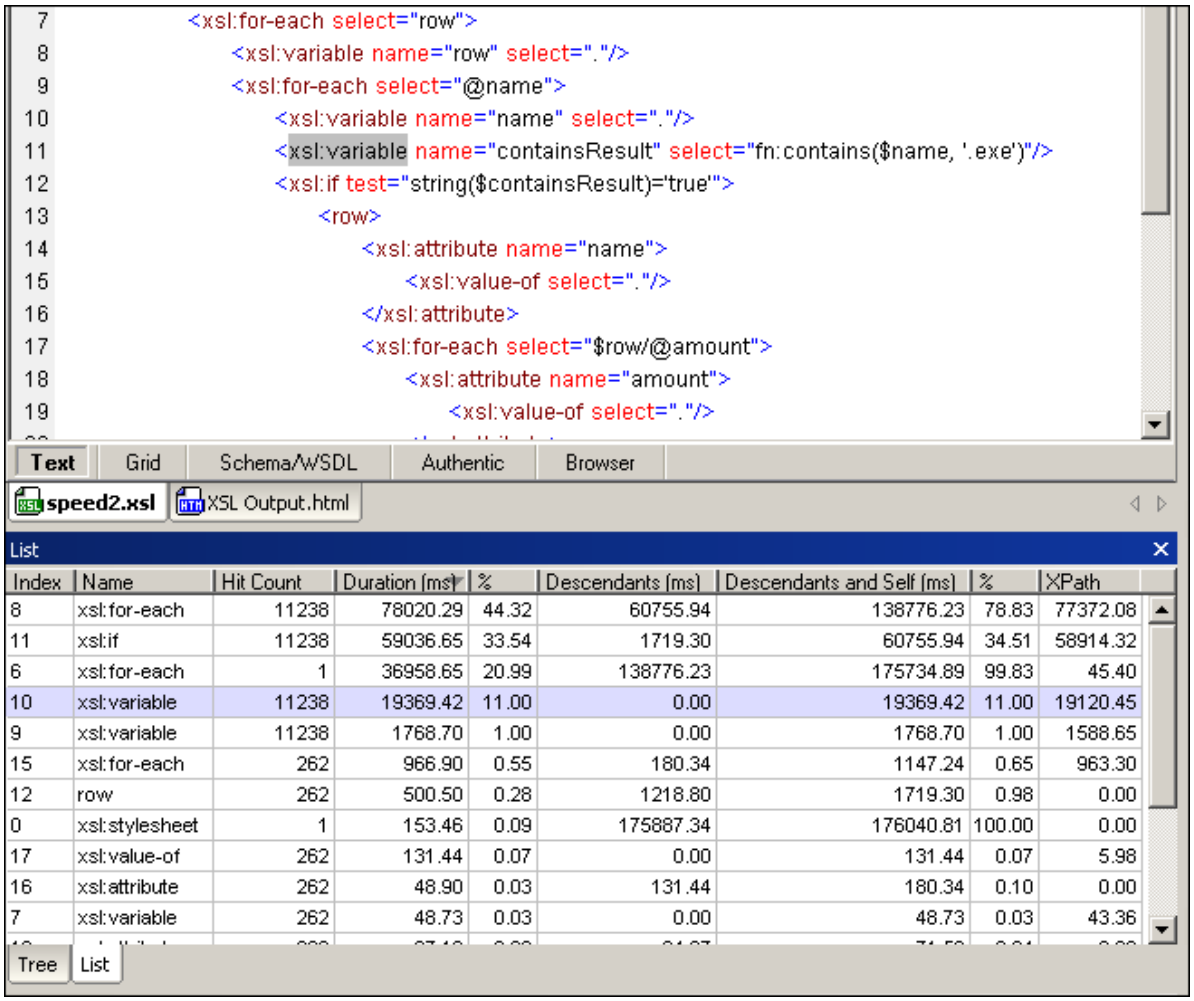
  Code-Fragment 1:

```
<xsl:for-each select="row">
      <xsl:variable name="row" select="."/>
      <xsl:for-each select="@name">
            <xsl:variable name="name" select="."/>
            <xsl:variable name="containsResult" select="fn:contains($name,
'.exe')"/>
            <xsl:if test="string($containsResult)='true'">

            ...

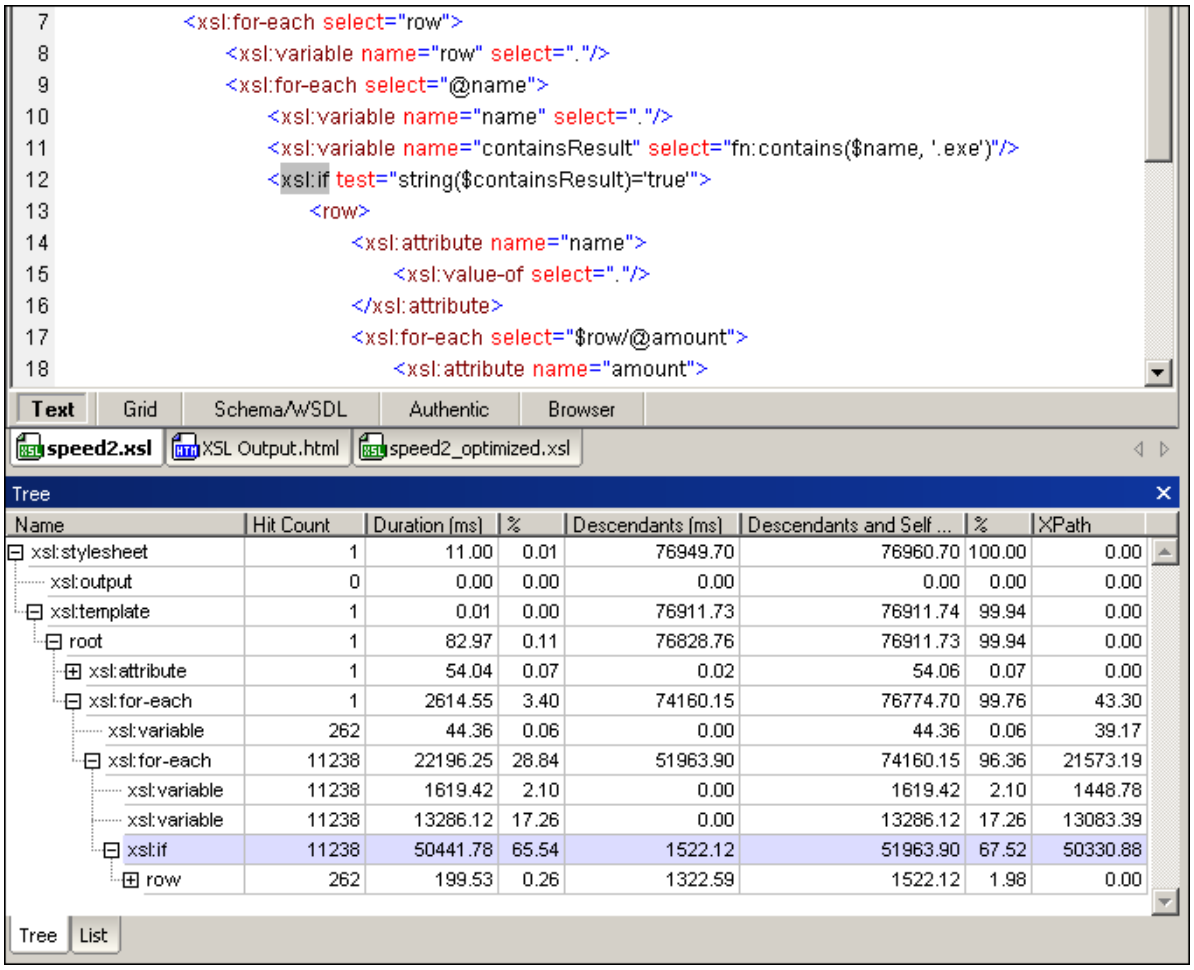
            </xsl:if>
      </xsl:for-each>
</xsl:for-each>
```

  Im Screenshot unten sehen Sie das Ergebnis der Analyse der Datei, die dieses Codefragment enthält und zwar sortiert nach der Dauer der Ausführung der Anweisungen. Die Anweisung, in der die Variable containsResult initialisiert wird, benötigt etwa 19 Sekunden Gesamtausführungszeit.

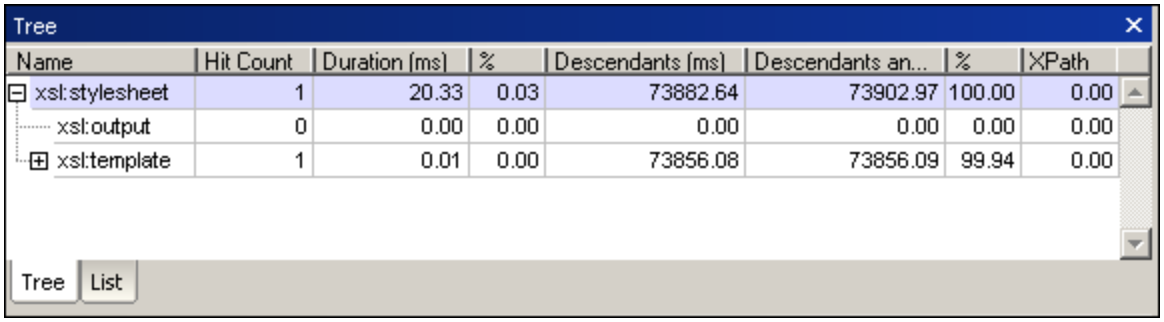


Im Screenshot unten sehen Sie die Ergebnisse in der Baumstruktur. Hier sehen wir, dass die if-Anweisung, die die Variable containsResult verwendet, etwa 50 Sekunden Gesamtausführungszeit benötigt:

© 2016-2022 Altova GmbH



Die XSLT-Transformation dauert insgesamt etwa 74 Sekunden:



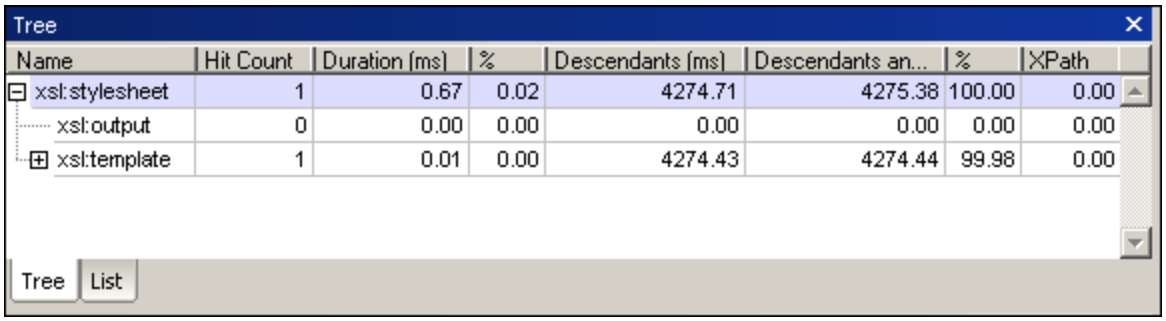
Code-Fragment 2:

```
<xsl:for-each select="row">
      <xsl:variable name="row" select="."/>
      <xsl:for-each select="@name">
            <xsl:if test="fn:contains(., '.exe')">

            ...
```

```
            </xsl:if>
        </xsl:for-each>
</xsl:for-each>
```

Nachdem das Stylesheet neu geschrieben wurde, ohne dass diese Variablen verwendet wurden, betrug die Gesamtausführungszeit nur mehr ca. 4,3 Sekunden:



- Verwenden Sie Variablen, wenn ein Wert oder ein Ausdruck mehrmals verwendet wird.
- Vermeiden Sie es, lokale Konstanten-Variablen in einer Funktion zu erstellen; erstellen Sie stattdessen globale Konstanten-Variablen.
- Vermeiden Sie es, Konstanten-Struktur-Fragmente in einer Funktion zu erstellen; erstellen Sie diese stattdessen global.
- Verwenden Sie möglichst wenige Prädikate, da die Filterung mit Prädikaten für jeden Node separat ausgewertet wird. Sie können die Anzahl der Aufrufe von Prädikaten z.B. durch Vorfiltern mit Hilfe von Namen reduzieren. In diesem Beispiel wird bei zwei Prädikaten `*` verwendet:

```
//*[node-name()=Book][author="Steve"]
```

In dieser Anweisung, die dasselbe Resultat hat, werden der Name `Book` und nur ein Prädikat verwendet:

```
//Book[@Author="Steve"]
```

- Teilen Sie Anweisungen so auf, dass die Teile der Anweisung, die nur einmal ausgeführt werden müssen, nur einmal ausgeführt werden. Erstellen Sie globale Variablen anhand von Teilen, die nur vom globalen Kontext abhängen.

9.2.1 Erstellen von XSLT-Profilen

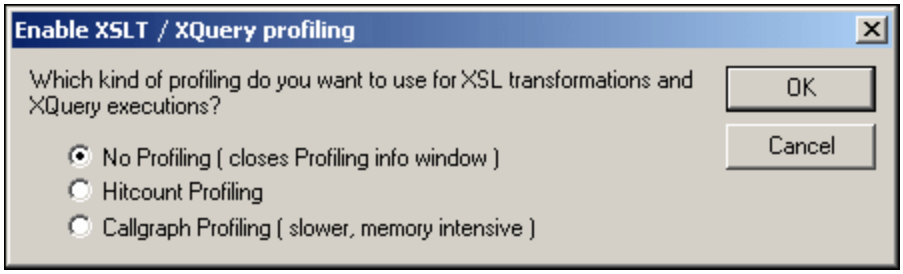
Starten des Profilers

Anmerkung: Die Ausführungsdauer, die im Profiler angezeigt wird, kann auch von anderen Applikationen, die auf Ihrem Computer ausgeführt werden, beeinflusst werden. Beim Analysieren mit dem Profiler sollte am besten nur die XMLSpy Applikation geöffnet sein.

So analysieren Sie ein XSLT Stylesheet:

© 2016-2022 Altova GmbH

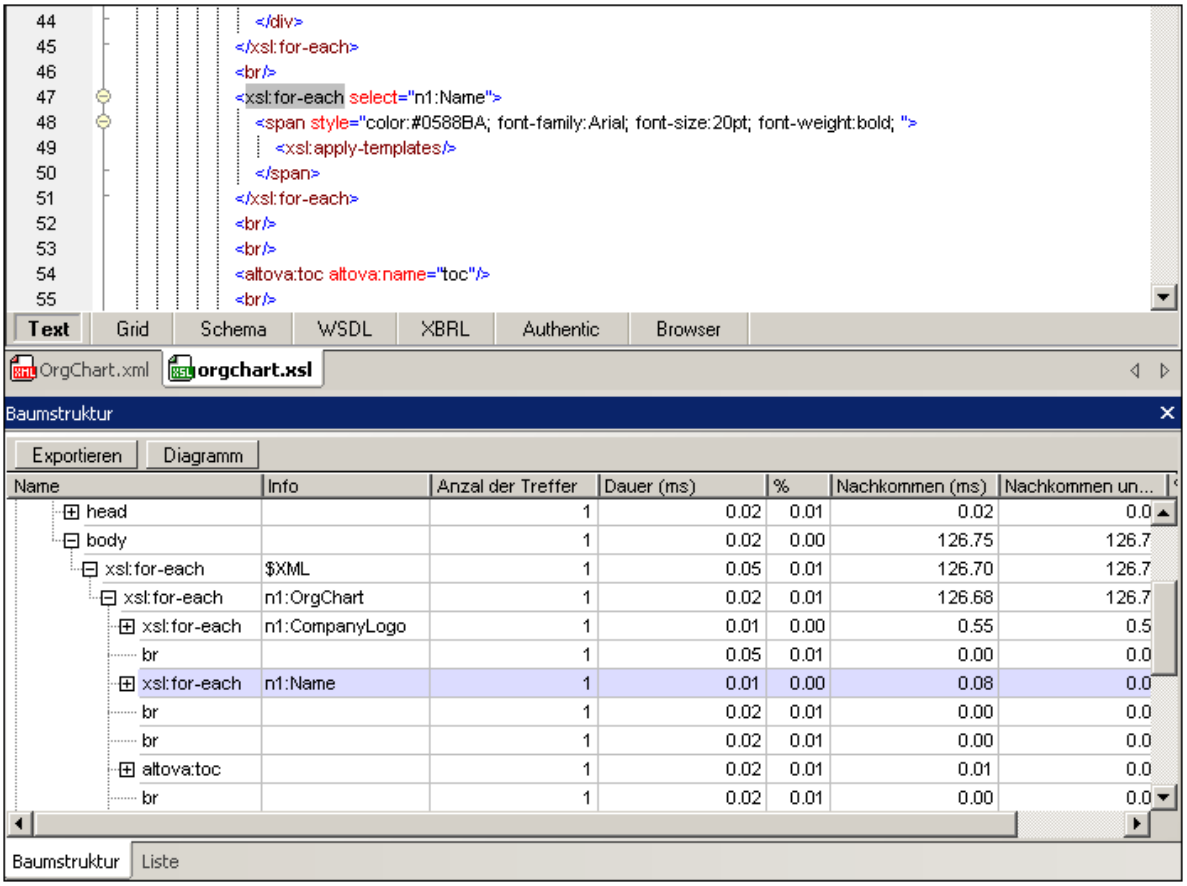
1. Öffnen Sie in XMLSpy die XML-Datei, die die Input-Daten für die XSLT-Transformation enthält.
2. Aktivieren Sie den Profiler durch Auswahl des Befehls **XSL/XQuery | XSLT/ XQuery Profilerstellung aktivieren**. Es wird ein Dialogfeld geöffnet.



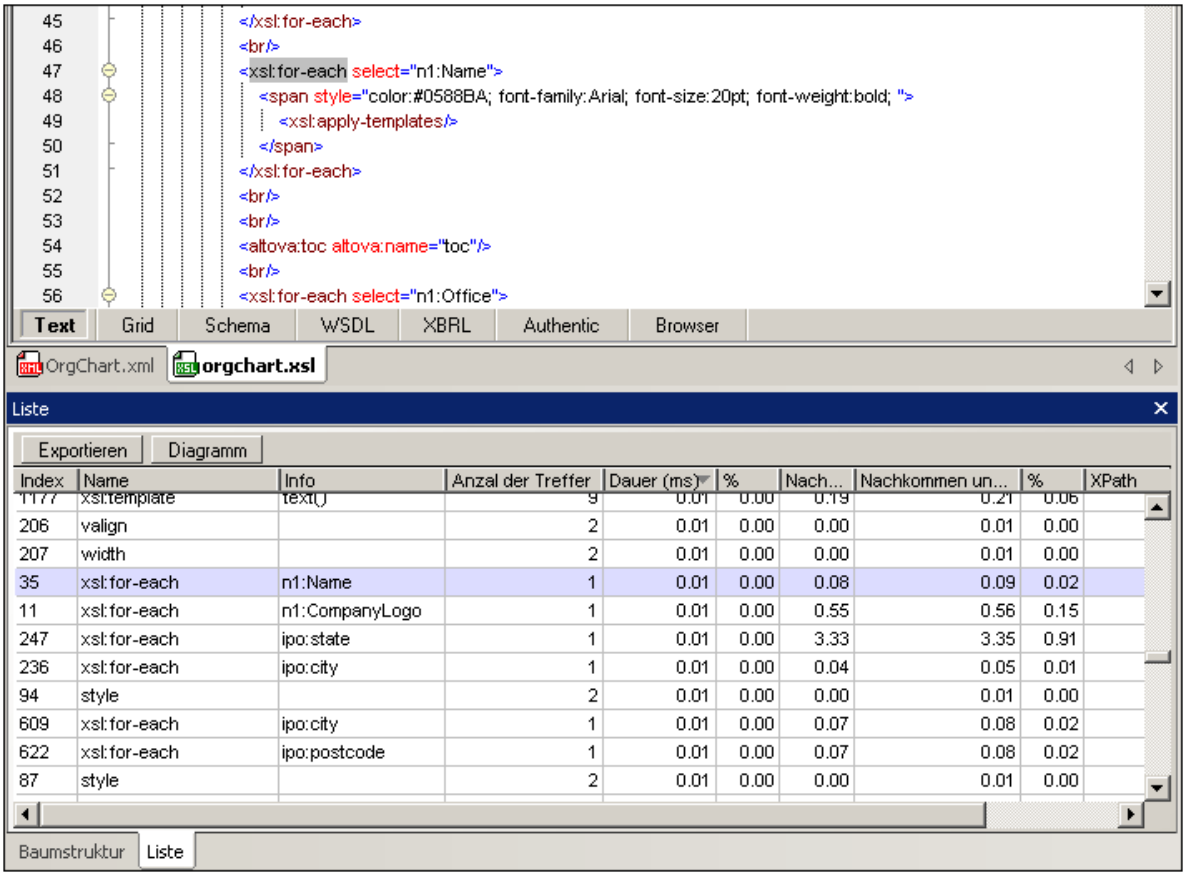
3. Wählen Sie die Option **Trefferquote Profilerstellung** oder **Callgraph Profilerstellung**. Klicken Sie zum Bestätigen auf **OK**. Es erscheint ein leeres Profiler-Fenster.
4. Führen Sie die XSL-Transformation aus (**XSL/XQuery | XSL-Transformation**). Es erscheint ein Dialogfeld, in dem Sie den Pfad zum XSLT Stylesheet angeben müssen, das Sie analysieren möchten. Nach Abschluss der Transformation wird im Profiler eine Statistik der Ausführungsdauer angezeigt.
5. Klicken Sie auf das "+"-Symbol, um die Zeilen im Profiler erweitert anzuzeigen, um die Ausführungsdauerstatistik für das XSLT Stylesheet zu sehen (*siehe Abbildung*). Beachten Sie, dass in diesen Abbildungen die Option "Trefferquote Profilerstellung" ausgewählt war.

Klicken Sie auf eine Zeile im Profiler, um die entsprechende Anweisung in der analysierten Datei zu markieren.

In diesem Screenshot sehen Sie die Baumstrukturansicht im Profiler:



Im folgenden Screenshot sehen Sie die Listenansicht im Profiler für dasselbe XSLT Stylesheet:



Verwenden der Informationen aus dem Profiler

Im Profiler werden die folgenden Informationen über die einzelnen Anweisungen im XSLT Stylesheet angezeigt:

- **Index:** Eine Nummer, die jeder Anweisung in der Reihenfolge, in der sie aufgerufen wurde, zugewiesen wird.
- **Name:** der Name der XSLT-Anweisung.
- **Anzahl der Treffer:** die Gesamtanzahl der Aufrufe der Anweisung während der Transformation.
- **Dauer (ms)** und **%:** Die Anzahl der Millisekunden, die für die Ausführung der Anweisung benötigt wurde, wobei dabei die Ausführungszeit für ihre untergeordneten Nodes nicht berücksichtigt wird, und der Prozentsatz an der Gesamtausführungsdauer.
- **Nachkommen (ms):** Die Anzahl der Millisekunden, die für die Ausführung der untergeordneten Nodes (Nachkommen) der Anweisung benötigt wurde.
- **Nachkommen und "selbst"** und **%:** Die Anzahl der Millisekunden, die für die Ausführung der Anweisung und der untergeordneten Nodes (Nachkommen) der Anweisung benötigt wurde und der Prozentsatz an der Gesamtausführungsdauer.
- **XPath:** Wenn die Anweisung eine XPath-Anweisung enthält, sehen Sie in dieser Spalte, wie lange die Ausführung der Anweisung dauerte.

Anmerkung: Bei Verwendung der Trefferquote-Profilerstellung entsprechen die Zeiten im Fenster "Profiler" der Gesamtausführungsdauer für alle Treffer in der Anweisung. Da bei Verwendung der Callgraph-Profilerstellung jeder Aufruf der Anweisung separat aufgelistet wird, beziehen sich die im Fenster "Profiler" angezeigten Zeiten auf die Dauer einer einzigen Ausführung der Anweisung.

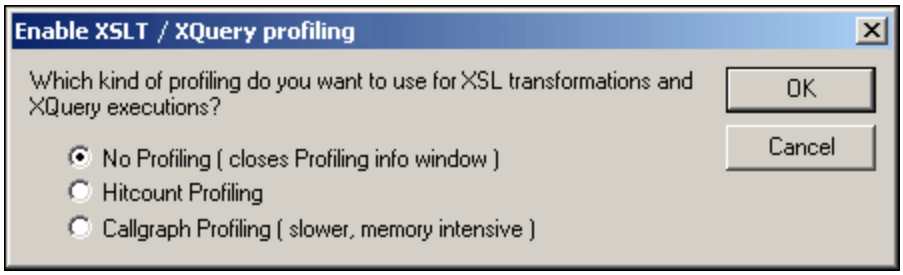
9.2.2 Erstellen von XQuery-Profilen

Starten des Profilers

Anmerkung: Die Ausführungsdauer, die im Profiler angezeigt wird, kann auch von anderen Applikationen, die auf Ihrem Computer ausgeführt werden, beeinflusst werden. Beim Analysieren mit dem Profiler sollte am besten nur die XMLSpy Applikation geöffnet sein.

So analysieren Sie ein XQuery-Dokument:

1. Öffnen Sie in XMLSpy die XML-Datei, die die Input-Daten für die XQuery-Ausführung enthält.
2. Aktivieren Sie den Profiler durch Auswahl des Befehls **XSL/XQuery | XSLT 2 aktivieren / XQuery Profilerstellung**. Es wird ein Dialogfeld geöffnet.

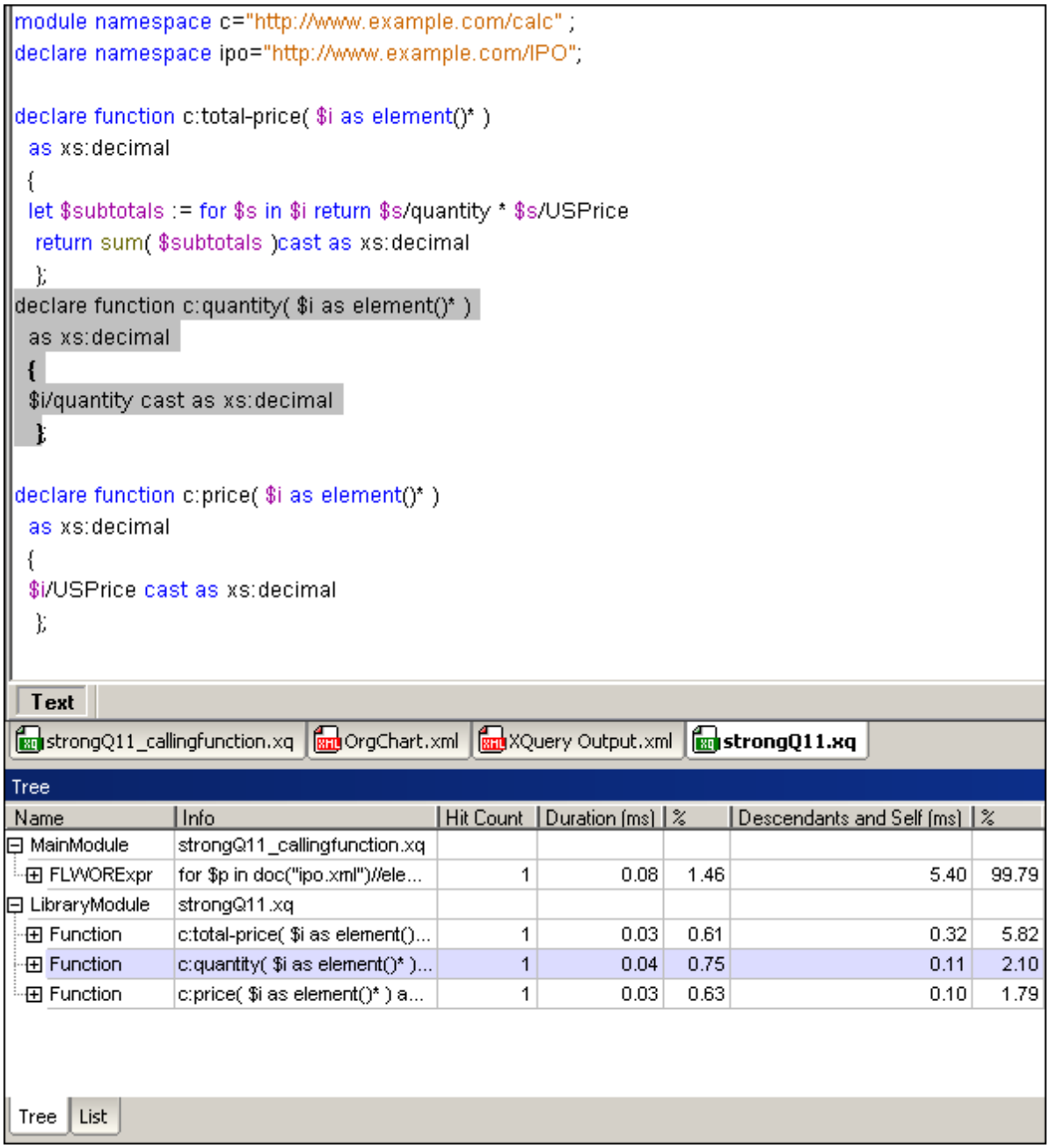


3. Wählen Sie die Option **Trefferquote Profilerstellung** oder **Callgraph-Profilerstellung**. Klicken Sie zum Bestätigen auf **OK**. Es erscheint ein leeres Profiler-Fenster.
4. Führen Sie die XQuery aus (**XSL/XQuery | XQuery-Ausführung**). Nach Abschluss der Ausführung wird im Profiler eine Statistik der Ausführungsdauer angezeigt.
5. Klicken Sie auf das "+"-Symbol, um die Zeilen im Profiler erweitert anzuzeigen, um die Ausführungsdauerstatistik für die Anweisungen im XQuery-Dokument zu sehen (*siehe Abbildung*). Beachten Sie, dass in diesen Abbildungen die Option **Trefferquote Profilerstellung** ausgewählt war.

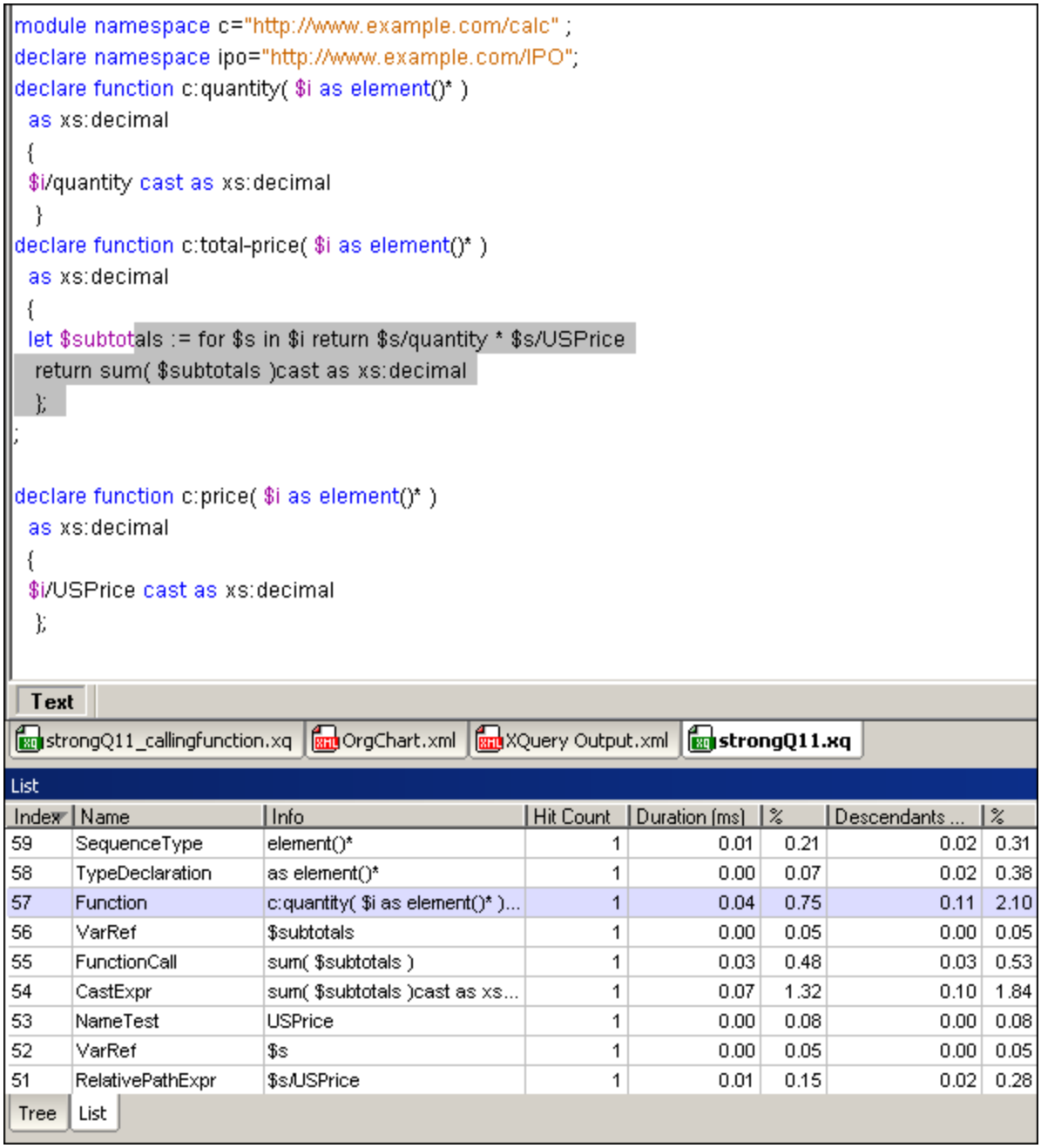
Klicken Sie auf eine Zeile im Profiler, um die entsprechende Anweisung in der analysierten Datei zu markieren.

In diesem Screenshot wird die Baumstrukturansicht im Profiler angezeigt.

© 2016-2022 Altova GmbH



Im folgenden Screenshot sehen Sie die Listenansicht für dasselbe XQuery-Dokument:



Verwendung der Informationen im Profiler

Im Profiler werden die folgenden Informationen über die einzelnen Anweisungen im XQuery-Dokument angezeigt:

- **Index:** Eine Nummer, die jeder Anweisung in der Reihenfolge, in der sie aufgerufen wurde, zugewiesen wird.
- **Name:** der Name der XQuery-Anweisung.

© 2016-2022 Altova GmbH

- **Info:** Informationen über die Anweisung. Wenn es sich z.B. bei der Anweisung um eine Variablendeklaration handelt, enthält diese Spalte den Namen der Variable und ihren Wert; handelt es sich um eine Funktion, so enthält diese Spalte den Namen und die Parameter der Funktion.
- **Anzahl der Treffer:** die Gesamtanzahl der Aufrufe der Anweisung während der Ausführung.
- **Dauer (ms)** und **%:** Die Anzahl der Millisekunden, die für die Ausführung der Anweisung ohne die untergeordneten Nodes benötigt wurde und der Prozentsatz an der Gesamtausführungsdauer.
- **Nachkommen und "selbst" (ms)** und **%:** Die Gesamtdauer der Ausführung der Anweisung und ihrer untergeordneten Nodes und der Prozentsatz an der Gesamtausführungsdauer.

Anmerkung: Bei Verwendung der Trefferquote-Profilerstellung entsprechen die Zeiten im Fenster "Profiler" der Gesamtausführungsdauer für alle Treffer in der Anweisung. Da bei Verwendung der Callgraph-Profilerstellung jeder Aufruf der Anweisung separat aufgelistet wird, beziehen sich die im Fenster "Profiler" angezeigten Zeiten auf die Dauer einer einzigen Ausführung der Anweisung.

9.2.3 Profiler-Ergebnisse: Exortieren und Diagramme

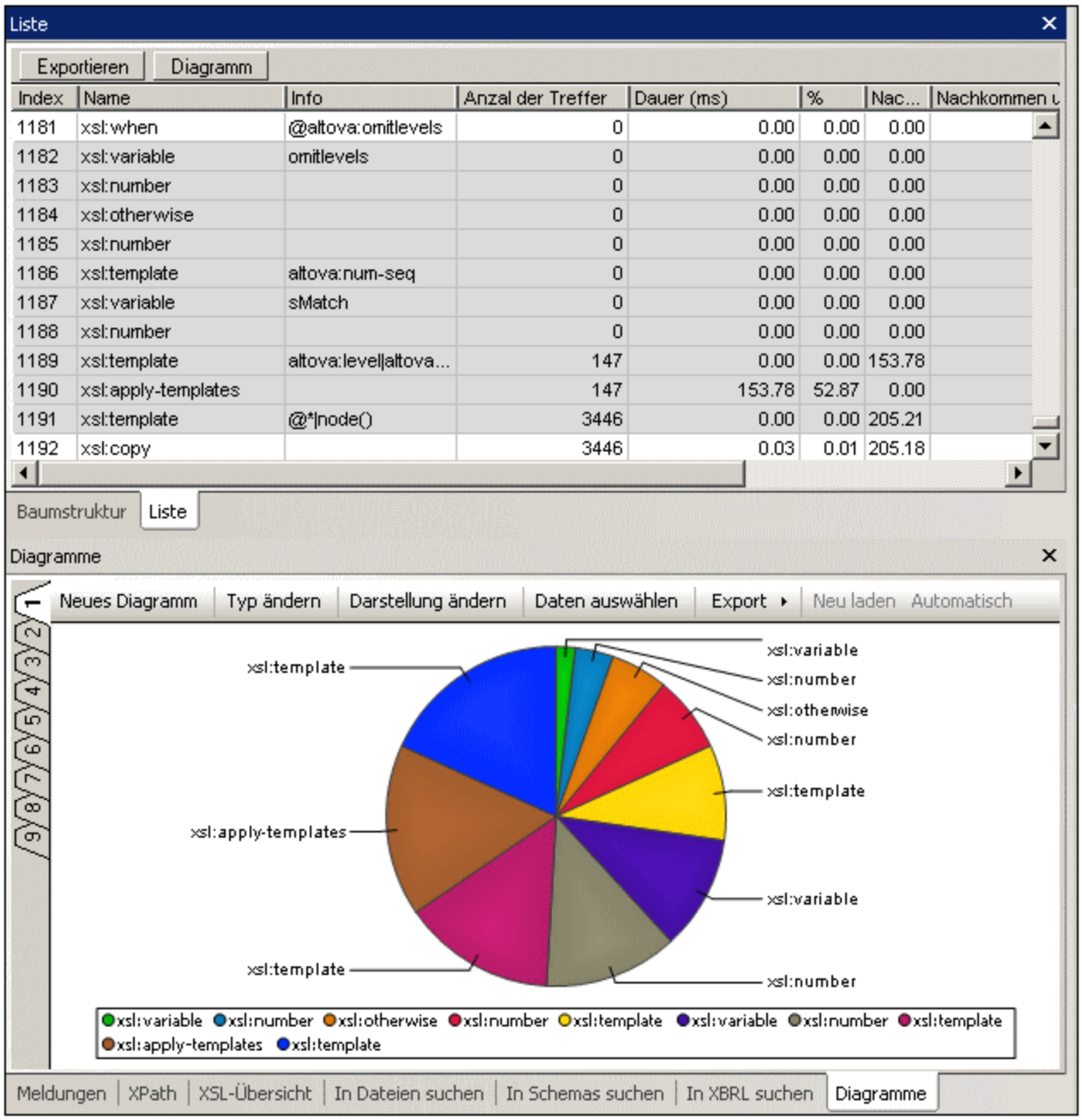
Nach Ausführung des XSLT/XQuery Profilers können die Ergebnisse in eine XML-Datei exportiert werden und es kann ein Diagramm davon als Bilddatei gespeichert werden.

Exportieren

Wenn Sie auf die Schaltfläche **Exportieren** klicken, werden Sie aufgefordert, einen Pfad und Dateinamen für die XML-Datei, in der die Profiler-Ergebnisse gespeichert werden sollen, auszuwählen. Um eine bessere Übersicht über die Struktur zu erhalten, zeigen Sie die XML-Datei am besten in der Grid-Ansicht an. Bei Anzeige eines XSLT-Profiler-Ergebnisses in der Grid-Ansicht sehen Sie, dass die Struktur des XML-Dokuments aus drei hierarchischen Ebenen besteht, von denen jede durch ein `node` Element identifiziert wird. Das erste `node` Element repräsentiert die Dokument-Root, das zweite das `xsl:stylesheet` Element und das dritte die globalen Elemente (wie z.B. `xsl:output` und `xsl:template`). Die Profilerstellungsdaten sind in de Attributen der einzelnen `node` Elemente gespeichert.

Diagramm

Nach Ausführung des XSLT/XQuery Profilers kann ein Diagramm der Ergebnisse oder einer Teilmenge der Ergebnisse generiert werden. Klicken Sie im Profiler-Fenster (*siehe Abbildung unten*) auf die Schaltfläche **Diagramm**, um ein Diagramm im Diagrammausgabefenster zu generieren (*siehe Abbildung unten*).

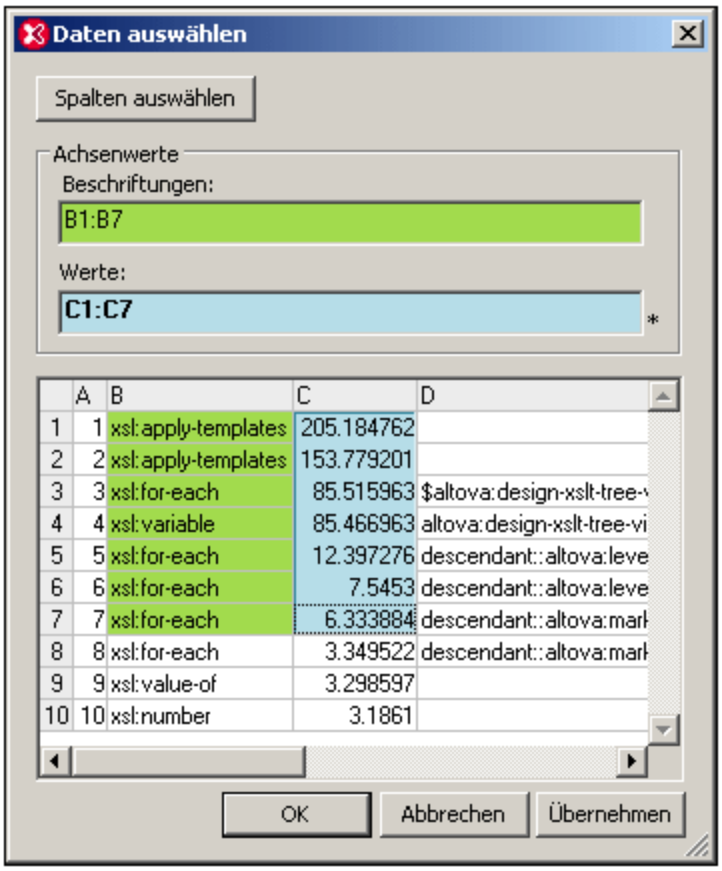


Beachten Sie die folgenden Punkte:

- Im Profiler-Fenster (Struktur- oder Listenansicht) können Sie durch Markieren einen Teil der Ergebnisse auswählen. Um mehrere Elemente zu markieren, klicken Sie auf die Einträge, während Sie die **Strg**- oder **Umschalttaste** gedrückt halten. In der Listenansicht können die Ergebnisse durch Klicken auf die entsprechende Spaltenüberschrift nach den Werten der Spalte sortiert werden. Dies kann z.B. nützlich sein, um die Ergebnisse nach den zeitintensivsten Werten zu sortieren und anschließend einige Werte zu markieren, die am meisten Transformationszeit beanspruchen. Durch Auswahl einer Teilmenge können unerwünschte Ergebnisse ausgefiltert werden. In der Abbildung oben wurden die markierten Ergebnisse ausgewählt.

 © 2016-2022 Altova GmbH

- Nachdem Sie ein Diagramm erstellt haben, können Sie seinen Typ (Kreisdiagramm, Balkendiagramm, Liniendiagramm usw.) durch Klicken auf die Diagrammausgabefenster-Schaltfläche **Typ ändern** 373 ändern. Die verschiedenen Diagrammtypen sind in dieser Dokumentation im Abschnitt Diagramme 350 näher beschrieben.
- Wenn Sie im Diagrammausgabefenster auf die Schaltfläche **Daten auswählen** klicken, wird das Dialogfeld "Daten auswählen" (*Abbildung unten)* aufgerufen. In diesem Dialogfeld können Sie aus der Datentabelle, die durch die Spaltenauswahl erzeugt wurde, Daten für die X- und die Y-Achse auswählen. Um Daten für die X-Achse auszuwählen, klicken Sie in das Textfeld "Achsenwerte" und geben Sie entweder den Bereich für die Tabellenwerte ein (z.B., `B1:B7`) oder ziehen Sie den Cursor vom Anfang des Bereichs bis zum Ende des Bereichs. Gehen Sie bei der Auswahl für die Y-Achse ebenso vor.



	A	B	C	D
1	1	xsl:apply-templates	205.184762	
2	2	xsl:apply-templates	153.779201	
3	3	xsl:for-each	85.515963	$altova:design-xslt-tree-v
4	4	xsl:variable	85.466963	altova:design-xslt-tree-vi
5	5	xsl:for-each	12.397276	descendant::altova:leve
6	6	xsl:for-each	7.5453	descendant::altova:leve
7	7	xsl:for-each	6.333884	descendant::altova:mark
8	8	xsl:for-each	3.349522	descendant::altova:mark
9	9	xsl:value-of	3.298597	
10	10	xsl:number	3.1861	

Wenn Sie auf die Schaltfläche **Spalten auswählen** klicken, können Sie die Datenauswahl für die Datentabelle ändern. Nähere Informationen zur Spaltenauswahl finden Sie unter Quell-XPath 357, Auswahl für die X-Achse 361 und Auswahl für die Y-Achse 366.

Nähere Informationen zu Diagrammen finden Sie in dieser Dokumentation im Abschnitt Diagramme 350.

10 XPath/XQuery-Ausdrücke

Mit Hilfe von XPath- und XQuery-Ausdrücken können Sie in XML-Strukturen navigieren. Mit Hilfe von XPath- und XQuery-Ausdrücken der Version 3.1, die zusätzlich noch Zuordnungen und Arrays unterstützt, können Sie auch in JSON-Strukturen navigieren. XPath ist eine Teilmenge von XQuery. Jeder Ausdruck, der in beiden Sprachen gültig ist, gibt in beiden Sprachen auch dasselbe Ergebnis zurück. Nähere Informationen zu den beiden Sprachen finden Sie in der XPath 3.1 Recommendation und der XQuery 3.1 Recommendation.

Auswertung von XPath/XQuery-Ausdrücken in XMLSpy

XMLSpy enthält umfangreiche Funktionen, um XPath/XQuery-Ausdrücke zu erstellen und diese anhand von XML- und JSON-Dokumenten auszuwerten. Auf diese Art können Sie Ausdrücke schnell anhand der gewünschten XML- oder JSON-Dokumente erstellen und testen.

Diese Analysefunktionen stehen im Fenster "XPath/XQuery", welches ein Ausgabefenster ist, das standardmäßig eines der Ausgabefenster[115] im unteren Bereich der Applikationsoberfläche bildet, zur Verfügung. Die Funktionen des XPath/XQuery-Fensters sind in den Unterabschnitten dieses Abschnitts beschrieben.

In einem typischen Anwendungsszenario würden Sie folgendermaßen vorgehen:

1. Öffnen des XML- oder JSON-Dokuments, für das Sie einen Ausdruck erstellen oder testen möchten.
2. Eingabe des XPath/XQuery-Ausdrucks in das Fenster "XPath/XQuery".
3. Ausführung der Auswertung oder des Debuggers, um die Ergebnisse zu sehen. Im Auswertungsfenster sehen Sie das Endergebnis, während Sie die Auswertung im Debugger Schritt für Schritt ausführen können, sodass Sie die Auswirkungen der Auswertung in jedem Schritt sehen.

In diesem Abschnitt

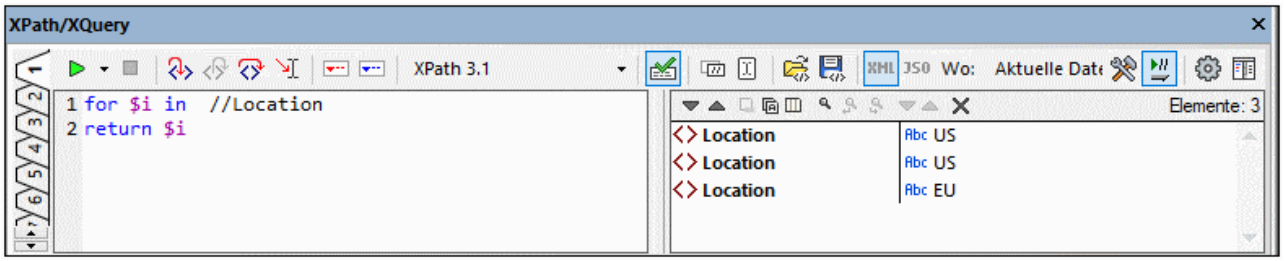
In diesem Abschnitt sind die Funktionalitäten des Fensters "XPath/XQuery" beschrieben. Er enthält die folgenden Kapitel:

- Das Fenster "XPath/XQuery"[583]
- Auswerten des Ausdrucks[585]: zur Auswertung von "XPath/XQuery"-Ausdrücken
- Debuggen des Ausdrucks[591]: zum Debuggen von "XPath/XQuery"-Ausdrücken
- Ausdruckserstellung[598]: zur Erstellung von "XPath/XQuery"-Ausdrücken
- XQuery-Ausdrücke für JSON[601]
- Anmerkungen[604]

© 2016-2022 Altova GmbH

10.1 Das Fenster "XPath/XQuery"

Im Fenster "XPath/XQuery" 124 (*Abbildung unten*) können Sie XPath- und XQuery-Ausdrücke anhand von XML- oder JSON-Dokumenten erstellen, auswerten und debuggen. (In XPath/XQuery 3.1 wurden Funktionen, die auch JSON-Abfragen ermöglichen, eingeführt. *Siehe auch JSON-Transformationen mit XSLT/XQuery* 769*)*

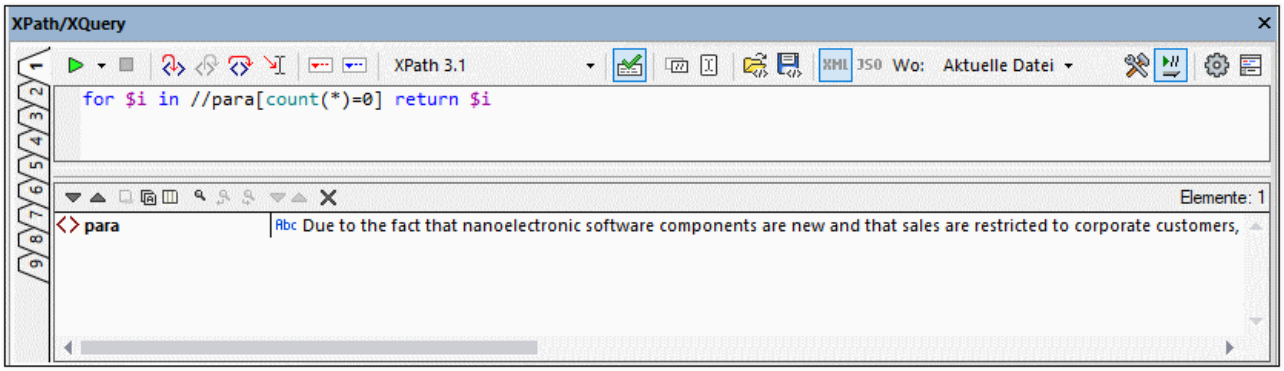


Die Grundzüge des Fensters und seiner Symbolleiste sind in der Beschreibung der Benutzeroberfläche unter XPath/XQuery-Fenster 124 erläutert.

Horizontales und vertikales Layout

In der rechten Ecke der Symbolleiste befindet sich eine Schaltfläche (*siehe der Abbildungen oben und unten*), über die Sie zwischen dem horizontalen und vertikalen Layout wechseln können. Sie können das Layout jederzeit und in jedem Modus wechseln (*siehe Auswertungsmodus und Debug-Modus* 584 *weiter unten*). In der Abbildung oben sehen Sie das vertikale Layout, welches sich eignet, wenn der XPath/XQuery-Ausdruck (in der Abbildung oben im linken Fenster) über mehrere Zeilen verläuft.

Das horizontale Layout (*Abbildung unten*) eignet sich in Fällen, in denen das Ergebnis Zeilen mit langer horizontaler Ausdehnung enthält.



Neun Register

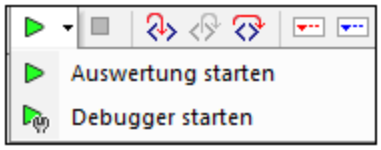
Das Fenster "XPath/XQuery" hat auf der linken Seite (*siehe Abbildungen oben*) neun Register. Dank mehrerer Register können Sie mit verschiedenen Ausdrücken auf verschiedenen Registern arbeiten und die Ergebnisse vergleichen. Klicken Sie auf den Ziehpunkt des Registers, wenn Sie zu diesem Register wechseln möchten.

Auswertungsmodus und Debug-Modus

Das Fenster "XPath/XQuery" [124] kann in zwei Modi verwendet werden:

- Im Auswertungsmodus [585] wird ein XPath- oder XQuery-Ausdruck anhand eines oder mehrerer XML/JSON-Dokumente ausgewertet. Der Ausdruck wird in das **Ausdrucksfenster** eingegeben und das Ergebnis wird im daneben liegenden Fenster "**Ergebnisse**" angezeigt. Wenn Sie auf Nodes im Ergebnis klicken, gelangen Sie zu diesem Node im XML- oder JSON-Dokument.
- Im Debug-Modus [591] können Sie einen XPath/XQuery-Ausdruck während der Anwendung am gerade aktiven XML-Dokument debuggen. Sie können Breakpoints und Tracepoints setzen und sich Schritt für Schritt durch die Auswertung bewegen. Bei jedem Schritt sehen Sie den Inhalt der Variablen. Außerdem können Sie benutzerdefinierte Überwachungsausdrücke festlegen, um zusätzliche Aspekte der Auswertung zu überprüfen.

Um zwischen den beiden Modi zu wechseln, wählen Sie den entsprechenden Befehl im Dropdown-Menü **Auswertung/Debugging starten** in der linken Ecke der Symbolleiste des Fensters (*siehe Abbildung unten*) .



Eine Anleitung zur Verwendung der beiden Modi finden Sie in den Unterabschnitten dieses Abschnitts.

Erstellung von XPath/XQuery-Ausdrücken

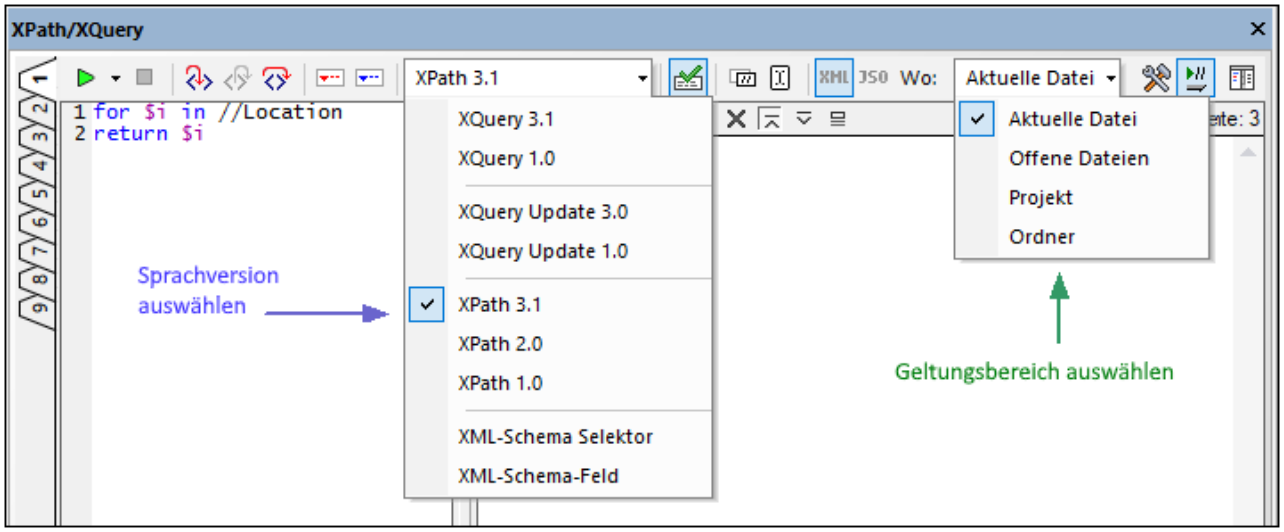
In beiden Modi [584] können Sie mit Hilfe der Ausdruckserstellung [598] syntaktisch korrekte Ausdrücke erstellen.

Über die **Erstellungsmodus**-Schaltfläche  in der Hauptsymbolleiste können Sie die Ausdruckserstellung [598] ein und ausschalten.

© 2016-2022 Altova GmbH

10.2 Auswerten des Ausdrucks

Im Fenster "XPath/XQuery" können Sie einen XPath- oder XQuery-Ausdruck (in einer von Ihnen ausgewählten Sprache; *siehe Abbildung unten*) erstellen und diesen dann in einem von Ihnen in der Option *Wo* (*siehe Abbildung*) ausgewählten Geltungsbereich auswerten. Der Ausdruck kann an der aktuellen Datei sowie an den folgenden Gruppen bestehend aus mehreren Dateien ausgewertet werden: (i) an allen gerade offenen Dokumenten; (ii) an Dateien des gerade aktiven XMLSpy Projekts[1080] oder (iii) an Dateien eines ausgewählten Ordners. Zum Testen des Ausdrucks genügt als Geltungsbereich eine einzige Datei. Sie können über das XPath/XQuery-Fenster jedoch auch bestimmte Daten in einer oder mehreren Dateien suchen und diese im Ergebnisfenster ausgeben lassen. Wählen Sie dazu im Geltungsbereich eine geeignete Dateigruppe aus.

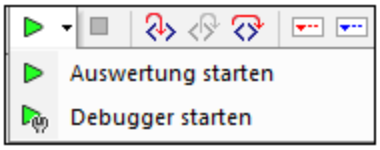


Das Fenster "XPath/XQuery" besteht aus einer Symbolleiste und zwei Fenstern - dem Ausdrucksfenster (*in der Abbildung oben links*) und dem Ergebnisfenster (*in der Abbildung oben rechts*).

Auswertung

Um einen XPath/XQuery-Ausdruck auszuwerten, gehen Sie folgendermaßen vor *(siehe auch Abbildung oben)*:

1. *Auswahl des Auswertungsmodus:* Wählen Sie im Dropdown-Menü des Befehls **Auswertung/Debugging starten (F5)** (in der Symbolleiste ganz oben links, *siehe Abbildung unten*) den Befehl **Auswertung starten** aus.

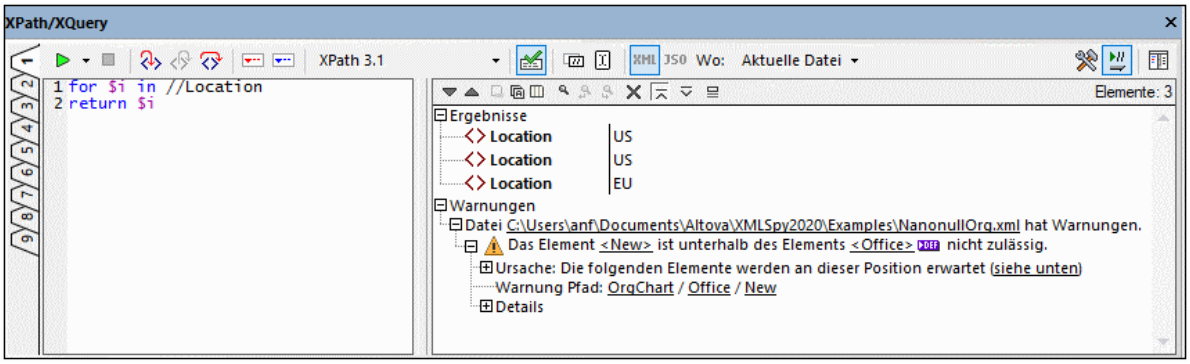


2. *Horizontales/Vertikales Layout:* Um das Layout zu wechseln, klicken Sie auf die Schaltfläche **Horizontales/Vertikales Layout** (in der Symbolleiste ganz oben rechts). Die Standardeinstellung ist das vertikale Layout.
3. *Auswahl der Sprachversion:* Wählen Sie in der Symbolleiste die Sprachversion für den gewünschten Ausdruck aus, z.B. XPath 3.1. Der Standardwert ist 3.1. Siehe auch die Anmerkung unten *XML-Schema-Selektor und XML-Schema-Feld*[587].

4. *Eingabe des Ausdrucks:* Geben Sie im Ausdrucksfenster den gewünschten Ausdruck ein. Als Hilfe bei der Erstellung eines Ausdrucks stehen die Eingabehilfen der Ausdruckserstellung[598] zur Verfügung. Nähere Informationen zu den Bearbeitungsfunktionen des Ausdrucksfensters finden Sie auch in der Anmerkung weiter unten mit dem Titel *Bearbeitung im Ausdrucksfenster*[587].
5. *Auswahl des Geltungsbereichs für die Auswertung:* Wählen Sie in der Symbolleistenoption *Wo:* die Datei(en), anhand derer der Ausdruck ausgewertet werden soll. Die Optionen sind: *Aktuelle Datei; Offene Dateien; (XMLSpy)-Projekt* und *Ordner*. Die Standardeinstellung ist *Aktuelle Datei*.



Wenn *Aktuelle Datei* ausgewählt ist, wird die aktuell aktive Datei abgefragt. Bei Auswahl der Option *Offene Dateien* wird der Ausdruck anhand aller gerade in XMLSpy geöffneten Dateien ausgewertet. "Projekt" bezieht sich auf das aktuell aktive XMLSpy-Projekt[1080]. (Die externen Ordner in einem XMLSpy-Projekt können durch Aktivierung des Symbols *Externe Ordner überspringen* von der Auswertung ausgenommen werden). Über die Option *Ordner* können Sie zum gewünschten Ordner navigieren. Der XPath-Ausdruck wird dann anhand der XML- oder JSON-Dateien in diesem Ordner ausgewertet.
6. *Auswahl einer XML- oder JSON-Auswertung:* Wenn der Geltungsbereich für die Auswertung die aktuelle Datei ist, wird der Auswertungsmodus (XML oder JSON) automatisch anhand des Konformitätstyps des Dateityps des Dokuments[1606] ermittelt (JSON-Modus für JSON-konforme Dateien, XML-Modus für Nicht-JSON-Dateien). Dieser automatisch ermittelte Modus kann nicht geändert werden und die Schaltflächen sind nicht aktiv. Wenn der Geltungsbereich mehrere Dateien umfasst, sind beide Schaltflächen aktiv und Sie können den gewünschten Auswertungsmodus auswählen. Als Standardmodus ist die Option ausgewählt, die zuletzt ausgewählt war.
7. *Definition des Kontext-Node:* Als Kontext-Node kann entweder (i) der Root-Node oder (ii) die aktuelle Auswahl im aktiven Dokument ausgewählt werden. Über die Symbolleistenschaltfläche **Aktuelle Auswahl im Dokument als Ausgangspunkt für XPath/XQuery festlegen** können Sie zwischen den beiden Einstellungen wechseln. Die Standardeinstellung ist der Root-Node.
8. *XML-Validierung:* Wenn die Symbolleistenschaltfläche **XML validieren** aktiv ist (Standardeinstellung), werden die ausgewerteten XML-Dateien validiert. Fehler werden als Warnungen behandelt und im Ergebnisfenster (*siehe Abbildung unten*) ausgegeben, doch wird die Auswertung fortgesetzt.



9. *Auswertung des Ausdrucks:* Wenn die Symbolleistenoption **Bei der Eingabe auswerten** ausgewählt ist, wird das Ergebnis der Auswertung, noch während Sie den Ausdruck eingeben, im Ergebnisfenster

 © 2016-2022 Altova GmbH

angezeigt. Wenn diese Option nicht ausgewählt ist, muss die Auswertung explizit durch Klick auf den Befehl **Auswertung/Debugging starten (F5)** (links oben in der Symbolleiste) gestartet werden.

Symbolleisten-Schaltflächen beim Auswertungsvorgang

	**Auswertung/Debugging starten (F5)**	Über diese Schaltfläche können Sie den Auswertungsmodus auswählen und die Auswertung starten.
	**Auswertung/Debugging beenden (Umschalt + F5)**	Ist während der Auswertung aktiv. Diese Schaltfläche ist nützlich, wenn die Auswertung sehr lange dauert oder in eine Endlosschleife gerät und Sie die Auswertung daher beenden möchten.
	**Horizontales/Vertikales Layout**	Wechselt zwischen dem horizontalen und dem vertikalen Layout.
	**Zum Erstellungsmodus wechseln**	Wechselt in den Ausdruckserstellungsmodus, in dem Ihnen bei der Erstellung von Ausdrücken kontextsensitive Eingabehilfen zur Verfügung stehen.
	**XPath der aktuellen Auswahl kopieren**	Kopiert den Locator-Pfad des Node im XML-Dokument im Ausdrucksfenster an die letzte Cursor-Position.
	**XML/JSON-Auswertungsmodus** (wechselt zwischen dem XML_ und dem JSON-Auswertungsmodus)	Das markierte Symbol kennzeichnet die aktive Option. Wenn der Geltungsbereich mehrere Dateien umfasst, sind beide Symbole aktiv und Sie können eines davon auswählen. Andernfalls wird der Auswertungsmodus wird automatisch anhand des Dateityps ermittelt; das andere Symbol ist deaktiviert.
	**Aktuelle Auswahl als Kontext festlegen**	Wechselt beim Auswertungskontext zwischen dem Root-Node und der aktuellen Auswahl.
	**Bei der Eingabe auswerten**	Aktiviert die Auswertung des Ausdrucks während der Eingabe.
	**XML validieren**	Wenn diese Option aktiviert ist, wird/werden das/die XML-Dokument(e) validiert.

XML-Schema-Selektor und XML-Schema-Feld
Die Optionen *XML-Schema-Selektor* und *XPath-Schema-Feld* können für eine eingeschränkte Untergruppe bestimmter XPath 1.0 Fälle verwendet werden und sind hilfreich, wenn im XML-Schema eindeutige Identity Constraints definiert wurden. Wenn eine dieser Optionen ausgewählt wurde, sind im XPath-Ausdruck nur Namensüberprüfungen (und das Platzhalterzeichen `*`) zulässig. Prädikate und XPath-Funktionen können in diesem Fall nicht verwendet werden. Auch sind bei Auswahl der Option *XML-Schema-Selektor* nur Ausdrücke auf der Child-Ebene zulässig; bei der Option *XML-Schema-Feld* sind Ausdrücke auf der Child- und der Attribut-Ebene zulässig. Nähere Informationen dazu finden Sie unter der XML Schema: Structures Recommendation.

Bearbeitung im Auswertungsfenster
Beachten Sie bei der Bearbeitung von Ausdrücken im Ausdrucksfenster die folgenden Punkte:

- Um den Ausdruck aus Gründen der Lesbarkeit in mehreren Zeilen zu erstellen, drücken Sie die **Eingabetaste**.

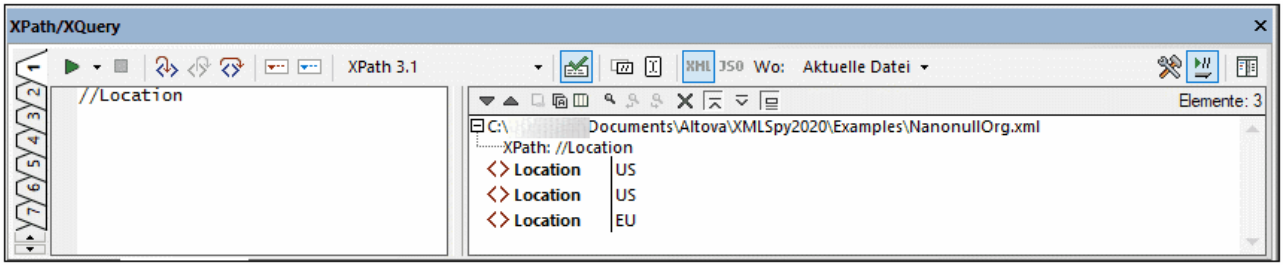
- Um den Text im Ausdrucksfeld zu vergrößern/zu verkleinern, klicken Sie in das Ausdrucksfeld und drehen Sie bei gedrückter **Strg**-Taste das Mausrad. *Beachten Sie, dass dies auch für den Ergebnisbereich gilt.*
- Anstatt den Locator-Pfad-Ausdruck eines Node manuell einzugeben, können Sie dies von XMLSpy erledigen lassen. Gehen Sie dazu folgendermaßen vor: (i) Platzieren Sie den Cursor an die Stelle im XPath-Ausdruck, an der der Locator-Pfad eingegeben werden soll; (ii) Setzen Sie den Cursor in Node, der als Ziel-Node verwendet werden soll; (iii) Klicken Sie in der Symbolleiste auf die Schaltfläche **XPath von der aktuellen Auswahl in das Bearbeitungsfeld kopieren**, um den Locator-Pfad des ausgewählten Node in den Ausdruck einzugeben. Der Locator-Pfad wird als absoluter Pfad, der am Root-Node des Dokuments beginnt, eingegeben.

XQuery- und JSON-Auswertungen

- Informationen zur XQuery-Auswertung finden Sie im Abschnitt XQuery-Auswertung[523]. (Die XQ-Symbole sind für die XQuery-Auswertung[523]; die XQU Symbole sind für XQuery Update-Auswertungen[528].)
- Eine Anleitung zur Abfrage von JSON-Dokumenten finden Sie unter XQuery-Ausdrücke für JSON[601] und JSON-Transformationen mit XSLT/XQuery[718].

Ergebnisfenster

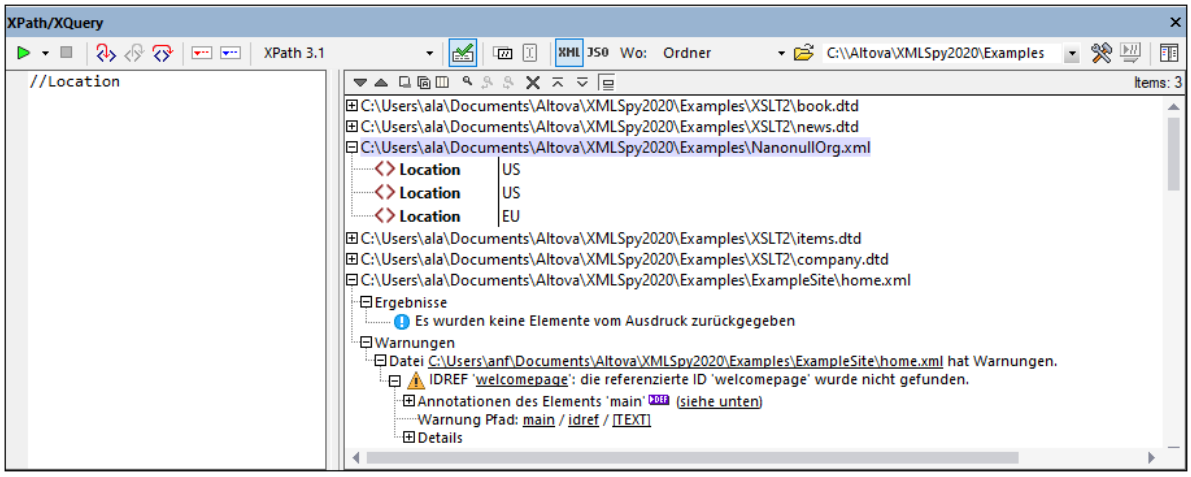
In der Abbildung unten sehen Sie das Ergebnisfenster auf der rechten Seite. Beachten Sie, dass dieser eine eigene Symbolleiste hat.



Der Ergebnisbereich verfügt über die folgenden Funktionalitäten:

- Aktivieren Sie das Symbol *Kopfzeile in der Ausgabe anzeigen*, um den Pfad zur XML-Datei und den XPath-Ausdruck anzuzeigen (*siehe Abbildung unten*).
- Die Ergebnisliste besteht aus zwei Spalten: (i) einem Node-Namen oder Datentyp; (ii) dem Inhalt des Node.
- Wenn der XPath-Ausdruck Nodes zurückgibt - wie z.B. Elemente oder Attribute - können Sie auswählen, ob der gesamte Inhalt der Nodes als Wert des Node angezeigt werden soll. Klicken Sie dazu auf die Schaltfläche *Gesamtes Ergebnis anzeigen.*
- Wenn das Ergebnis - im Gegensatz zu mit einem Ausdruck generierten Literalen - einen Node enthält (auch einen Text-Node), wird der Node im XML-Dokument im Hauptfenster markiert, wenn Sie im Ergebnisfenster darauf klicken.
- Wenn die Auswertung an mehreren (*in der* Wo-*Option definierten*) Dateien durchgeführt wird, so werden die Ergebnisse für jede einzelne Datei separat unter dem Pfad dieser Datei aufgelistet (*siehe Abbildung unten*). Wenn der Auswertungsmodus XML ist, werden XML-konforme Dateien ausgewertet und andere Dateitypen übersprungen. Wenn der Auswertungsmodus JSON ist, werden JSON-konforme Dateien ausgewertet und andere Dateitypen übersprungen.

© 2016-2022 Altova GmbH



- Wenn das Ergebnis mehrere Dateien umfasst, wird die entsprechende Datei bei Klick auf einen Dateinamen in der Ergebnisliste in XMLSpy geöffnet und zur aktiven Datei gemacht.
- Sie können beide Spalten einer Ergebnis-Subzeile oder nur die Wertespalte kopieren. Um alle Spalten zu kopieren, klicken Sie mit der rechten Maustaste auf eine Subzeile und aktivieren Sie den Befehl **Beim Kopieren werden alle Spalten inkludiert**. (Alternativ dazu können Sie diesen Befehl über die Symbolleisten-Schaltfläche im Ergebnisbereich ein- und ausschalten.) Klicken Sie anschließend mit der rechten Maustaste auf die gewünschte Subzeile und wählen Sie entweder **Subzeile kopieren** (für diese Subzeile) oder **Alles kopieren** (für alle Subzeilen).

Symbolleiste des Ergebnisfensters

Die Symbolleiste des Ergebnisfensters enthält Symbole für Navigations-, Such- und Kopierfunktionen. In der unten stehenden Tabelle finden Sie eine Beschreibung der Symbole von links nach rechts. Die dazugehörigen Befehle stehen auch über das Kontextmenü von Einträgen der Ergebnisliste zur Verfügung.

Symbol	Funktion
Weiter, Zurück	Wählt jeweils den nächsten bzw. vorhergehenden Eintrag in der Ergebnisliste aus
Ausgewählte Textzeile in die Zwischenablage kopieren	Kopiert die Wertespalte des ausgewählten Ergebniseintrags in die Zwischenablage. Um alle Spalten zu kopieren, aktivieren Sie den Befehl *Beim Kopieren werden alle Spalten inkludiert (siehe unten)*
Alle Nachrichten in die Zwischenablage kopieren	Kopiert die Wertespalte aller Ergebniseinträge, darunter auch leere Werte, in die Zwischenablage. Jeder Eintrag wird als separate Zeile kopiert.
Beim Kopieren werden alle Spalten inkludiert	Wechselt zwischen der Funktion zum Kopieren (i) aller Spalten und (ii) nur der Wertespalte. Das Spaltentrennzeichen ist ein einziges Leerzeichen.
Suchen	Öffnet ein *Suchdialogfeld* zum Suchen nach einem beliebigen String - darunter auch von Sonderzeichen - in der Ergebnisliste.
Vorheriges suchen	Sucht nach der vorhergehenden Instanz des zuletzt im *Suchdialogfeld* gesuchten Eintrags.
Weitersuchen	Sucht nach der nächsten Instanz des zuletzt im *Suchdialogfeld* gesuchten Eintrags.

Zurücksetzen	Löscht die Ergebnisliste.
Mehrzeilige Ergebnisse auf eine einzige Zeile reduzieren	Wenn die Wertespalte eines Ergebniseintrags mehrzeiligen Text (Text der Zeilenvorschubzeichen beinhaltet) enthält, können Sie zwischen einer mehrzeiligen und einer einzeiligen Anzeige wechseln.
Vollständiges Ergebnis anzeigen	Zeigt den gesamten Inhalt von Nodes als Wert des Node an.
Kopfzeile in der Ausgabe anzeigen	Zeigt den Pfad der XML-Datei(en) und den XPath-Ausdruck an.

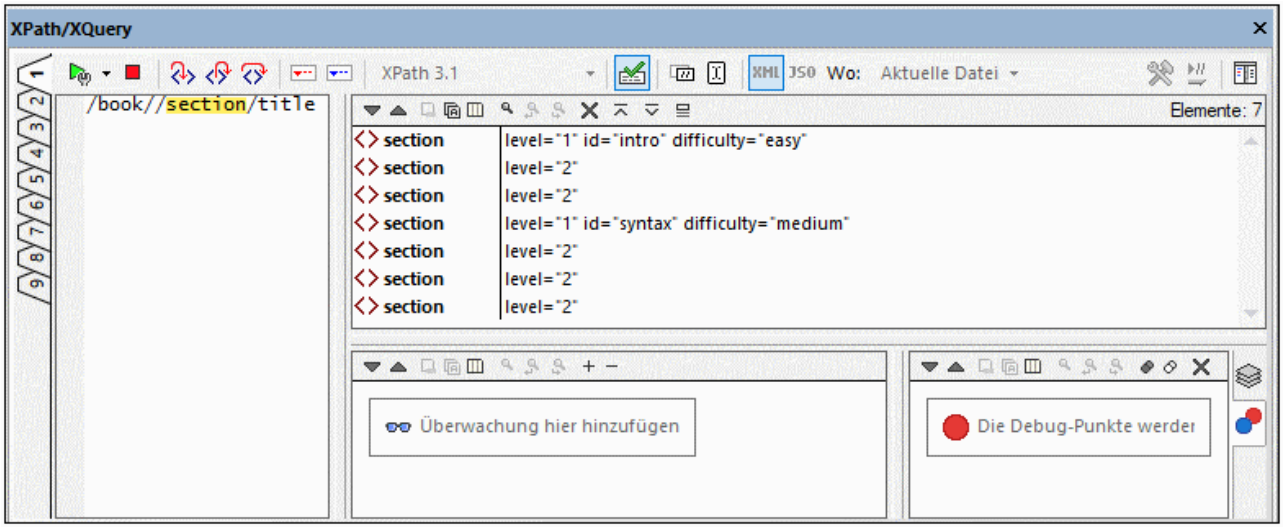
© 2016-2022 Altova GmbH

10.3 Debuggen des Ausdrucks

Im Debug-Modus des "XPath/XQuery"-Fensters (*Abbildung unten*) können Sie einen XPath/XQuery-Ausdruck im Kontext der aktiven Datei debuggen.

Im Debug-Modus werden zum Ergebnisbereich zwei zusätzliche Bereiche hinzugefügt *(siehe Abbildung unten)*:

- der Call Stack- und der Debug-Punkte-Bereich. Jeder davon hat ein eigenes Register im Fenster.
- der Bereich für Variablen und Überwachungsausdrücke; sowohl Überwachungsausdrücke als auch Variablen werden im selben Bereich angezeigt



Die Bereiche für Ausdrücke und Ergebnisse können nebeneinander oder übereinander angeordnet werden. Um zwischen diesen Layouts zu wechseln, klicken Sie (rechts oben in der Symbolleiste des Fensters) auf die Schaltfläche **Horizontales/Vertikales Layout**.

Im Debugger-Modus stehen die folgenden Funktionalitäten zur Verfügung:

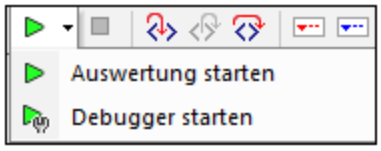
- Hier können Sie in den XPath-Auswertungsvorgang einsteigen und Schritt für Schritt sehen, wie der XPath-Ausdruck ausgewertet wird. Verwenden Sie dazu die Symbolleisten-Schaltfläche **Einsteigen (F11)**. Bei jedem Auswertungsschritt wird der Teil des Ausdrucks, der soeben ausgewertet wird, gelb markiert (*siehe Abbildung oben*), während das Ergebnis der Auswertung dieses Schritts im Ergebnisbereich angezeigt wird. So wurden z.B. in der Abbildung oben alle `section`-Nachfahren des Elements `book` ausgewählt.
- Sie können Breakpoints setzen, an denen die Auswertung angehalten werden soll, damit Sie Ergebnisse an diesen Stellen überprüfen können. Sie können die Auswertung durchlaufen lassen, indem Sie nur an den Breakpoints anhalten. Verwenden Sie dafür die Symbolleisten-Schaltfläche **Debugger starten** (F5). Dies geht schneller, als wenn Sie bei jedem Schritt mit **Einsteigen (F11)** anhalten.
- Sie können Tracepoints setzen, um einen Bericht über die Ergebnisse an den als Tracepoints markierten Stellen zu sehen. Die Auswertung wird dabei nicht angehalten (außer bei Breakpoints), sondern die Tracepoint-Ergebnisse werden in einer Liste im Ergebnisbereich angezeigt.
- Mit Hilfe von Überwachungsausdrücken können bestimmte Informationen (wie Dokumentdaten oder Auswertungsaspekte) überprüft werden. Dies ist vor allem an Breakpoints nützlich.

- Variablen, die sich im Geltungsbereich befinden, werden einschließlich ihrer Werte im Bereich für Variablen und Überwachungsausdrücke angezeigt.
- Prozessoraufrufe eines Auswertungsschritts werden auf dem Register "Call Stack" des Call Stack- und Debug-Punkte-Bereichs angezeigt.
- Wenn Breakpoints und Tracepoints gesetzt wurden, werden diese auf dem Register "Debug-Punkte" des Call Stack- und Debug-Punkte-Bereichs angezeigt.

Nähere Informationen zu diesen Funktionalitäten finden Sie in der jeweiligen Beschreibung dazu weiter unten.

Einrichten des Debug-Modus

Um in den Debug-Modus zu wechseln, klicken Sie im Dropdown-Menü des Befehls **Auswertung/Debugging starten (F5)** (links oben in der Symbolleiste; *siehe Abbildung unten*) auf **Debugger starten**. Im Debug-Modus werden die Bereiche "Überwachungsausdrücke", "Call Stack" und "Debug-Punkte" angezeigt. Um mit dem Debuggen des aktuellen Ausdrucks zu beginnen, klicken Sie auf **Debugger starten** oder drücken Sie **F5**.



Beachten Sie, dass der Debug-Modus nur bei der aktuellen Datei funktioniert; er kann nicht mit mehreren Dateien verwendet werden. Infolgedessen wird die Option *Wo* automatisch auf *Aktuelle Datei* gesetzt, wenn Sie in diesen Modus wechseln.

Schaltflächen für die Konfiguration des Debug-Modus

	**Auswertung/Debugging starten (F5)**	Startet den Debugger
	**Horizontales/Vertikales Layout**	Wechselt zwischen dem horizontalen und dem vertikalen Layout.
	**Zum Erstellungsmodus wechseln**	Wechselt in den Ausdruckserstellungsmodus, in dem Ihnen bei der Erstellung von Ausdrücken kontextsensitive Eingabehilfen zur Verfügung stehen.
	**XPath der aktuellen Auswahl kopieren**	Kopiert den Locator-Pfad des Node im XML-Dokument im Ausdrucksfenster an die letzte Cursor-Position.
	**XML/JSON-Auswertungsmodus**	Das markierte Symbol kennzeichnet die aktive Option. Der Auswertungsmodus wird automatisch anhand des Dateityps ermittelt; das andere Symbol ist deaktiviert.
	**Aktuelle Auswahl als Kontext festlegen**	Wechselt beim Auswertungskontext zwischen dem Root-Node und der aktuellen Auswahl.
	**Bei der Eingabe auswerten**	Aktiviert die Auswertung des Ausdrucks während der Eingabe.

© 2016-2022 Altova GmbH

	**XML validieren**	Wenn diese Option aktiviert ist, wird/werden das/die XML-Dokument(e) validiert.

Ausführen des Debuggers

Das Debuggen eines XPath/XQuery-Ausdrucks erfolgt in den folgenden Schritten:

1. Machen Sie die XML/JSON-Datei, an der Sie den Ausdruck ausführen möchten, zur aktiven Datei.
2. Geben Sie den XPath/XQuery-Ausdruck in das Ausdrucksfenster ein.
3. Setzen Sie die gewünschten Breakpoints oder Tracepoints. Ein Breakpoint ist ein Punkt, an dem die Auswertung unterbrochen wird. Ein Tracepoint ist ein Punkt in der Auswertung, der aufgezeichnet wird, sodass Sie einen zurückverfolgbaren Pfad der Auswertungsergebnisse zur Verfügung haben.
4. Wenn Sie auf **Debugger starten** klicken, wird die Auswertung in einem einzigen Schritt bis zum Ende ausgeführt, es sei denn, es wurde ein Breakpoint im Ausdruck markiert. Klicken Sie wiederholt auf **Debugger starten,** um nach jedem Breakpoint fortzufahren und bis zum Ende der Auswertung zu gelangen.
5. Mit Hilfe der Funktionen Einsteigen/Aussteigen/Überspringen können Sie sich Schritt für Schritt durch die Auswertung bewegen.

Schaltflächen für das Debuggen

	**Debugger starten (F5)**	Startet den Debugger. Die Auswertung wird direkt bis zum Ende durchgeführt.
	**Debugging beenden (Umschalt+F5)**	Beendet die Auswertung und stoppt den Debugger.
	**Einsteigen (F11)**	Steigt Schritt für Schritt durch die Auswertung.
	**Aussteigen (Umschalt+F11)**	Steigt aus dem aktuellen Auswertungsschritt aus und geht zum übergeordneten Schritt.
	**Überspringen (Strg+F11)**	Überspringt untergeordnete Schritte.
	**Breakpoint einfügen/löschen (F9)**	Fügt an dem Ausführungsschritt, in den Sie den Cursor platzieren, einen Breakpoint ein, bzw. entfernt diesen.
	**Tracepoint einfügen/löschen (Umschalt+F9)**	Fügt an dem Ausführungsschritt, in den Sie den Cursor platzieren, einen Tracepoint ein, bzw. entfernt diesen.

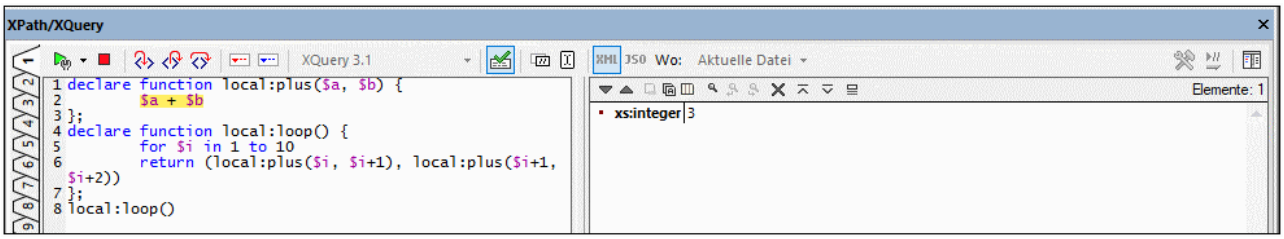
Einsteigen, Aussteigen und Überspringen von Auswertungsschritten

Mit der Funktion *Einsteigen* können Sie die Auswertung Schritt für Schritt durchgehen. Jeder Klick leitet Sie durch den entsprechenden Schritt der Auswertung. Der aktuelle Schritt wird im Ausdruck markiert (*siehe Abbildung unten*). Mit der Funktion *Aussteigen* gelangen Sie zur nächsthöheren Ebene und mit *Überspringen* werden Schritte auf niedrigeren Ebenen übersprungen und Sie gelangen zum nächsten Schritt auf derselben Ebene. Um die *Schrittfunktionen* zu probieren, kopieren Sie den unten stehenden XQuery 3.1-Ausdruck in das Ausdrucksfenster und klicken Sie auf die drei *Schritt*-Schaltflächen, um zu sehen, wie diese funktionieren.

XQuery 3.1-Ausdruck zum Testen der Einsteigen-, Aussteigen und Überspringen-Funktion

```
declare function local:plus($a, $b) {
        $a + $b
};
declare function local:loop() {
        for $i in 1 to 10
        return (local:plus($i, $i+1), local:plus($i+1, $i+2))
};
local:loop()
```

In der Abbildung unten sehen Sie die Auswertung an der Stelle, an der die Verarbeitung bei Erreichen des Additionsschritts `$a + $b` während der ersten Verarbeitung der Schleife, d.h. wenn `$i=1`, angehalten wird. In diesem Additionsschritt wird als Ergebnis (als Folge von `1+2`) `3` angezeigt.



Breakpoints

Breakpoints sind Punkte, an denen der Debugger angehalten werden soll, nachdem er mit **Debugger starten** gestartet wurde. Sie eignen sich dazu, einen bestimmten Teil eines Ausdrucks zu analysieren. Wenn der Debugger an einem Breakpoint anhält, können Sie das Ergebnis überprüfen. Mit Hilfe der Funktion **Einsteigen** können Sie anschließend die Ergebnisse des nächsten Schritts der Auswertung anzeigen. Um einen Breakpoint zu setzen, platzieren Sie den Cursor an der Stelle in den Ausdruck, an der Sie den Breakpoint setzen möchten und klicken Sie auf die Symbolleisten-Schaltfläche **Breakpoint einfügen/löschen (F9)**. Der Breakpoint wird daraufhin mit einem rot strichlierten Überstrich markiert. Um einen Breakpoint zu löschen, wählen Sie ihn aus und klicken Sie auf **Breakpoint einfügen/löschen (F9).**

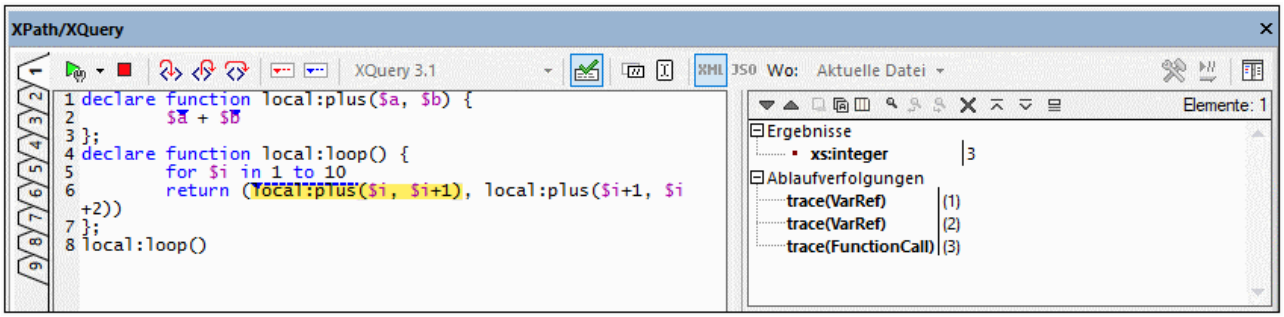
Siehe auch Debug-Punkte 596 weiter unten.

Tracepoints

Tracepoints sind Breakpoints, an denen die Ergebnisse aufgezeichnet und in der *Ablaufverfolgungsstruktur* des *Ergebnisregisters* (*siehe Abbildung unten*) angezeigt werden. Dadurch sehen Sie alle Auswertungsergebnisse bestimmter Teile des Ausdrucks. So wurden z.B. in der Abbildung unten Tracepoints bei `$a`, `$b`, `local:plus($i, $i+1)` gesetzt. Die Ergebnisse dieser Tracepoints werden während der ersten Iteration durch die Schleife in der *Ablaufverfolgungsstruktur* angezeigt: `$a=1`, `$b=2`, und `local:plus($i, $i+1)=3`.

Um einen Tracepoint zu setzen, platzieren Sie den Cursor an der Stelle in den Ausdruck, an der Sie den Tracepoint setzen möchten und klicken Sie auf die Symbolleisten-Schaltfläche **Tracepoint einfügen/löschen (Umschalt+F9)**. Der Tracepoint wird daraufhin mit einem blau strichlierten Überstrich markiert (*siehe Abbildung unten*). Um einen Tracepoint zu löschen, wählen Sie ihn aus und klicken Sie auf **Tracepoint einfügen/löschen (F9).**

© 2016-2022 Altova GmbH

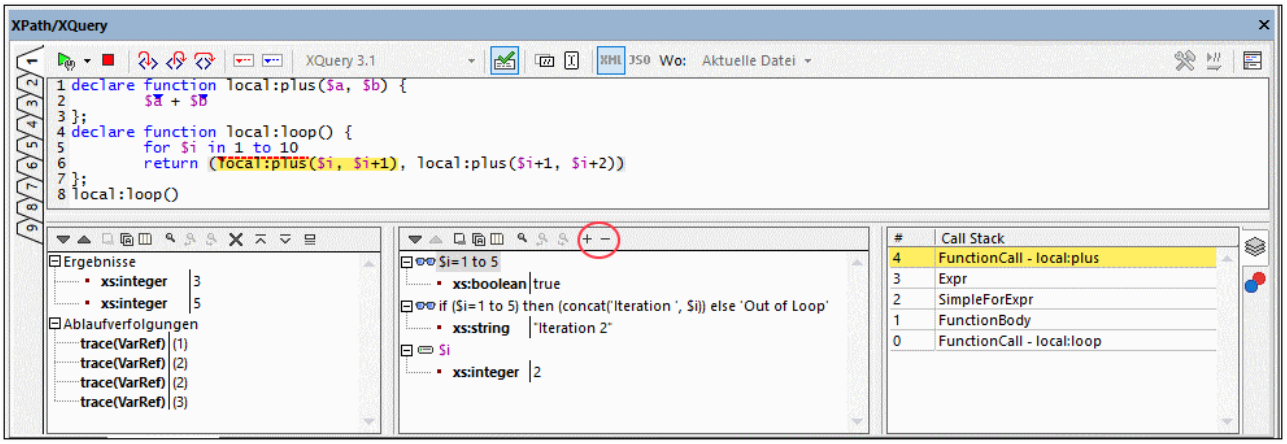


Anmerkung: Wenn an einer Stelle eines Ausdrucks sowohl ein Breakpoint als auch ein Tracepoint gesetzt wurde, so besteht der Überstrich aus abwechselnd gesetzten roten und blauen Strichen.

Siehe auch Debug-Punkte 596 weiter unten.

Variablen, Überwachungsausdrücke und Call Stack

Variablen und Überwachungsausdrücke werden im Variablen- und Überwachungsausdrucksbereich *(mittlerer unterer Bereich in der Abbildung unten) angezeigt*.



Variablen

Variablen, die im Ausdruck deklariert wurden und sich im aktuellen Auswertungsschritt im Geltungsbereich befinden, werden zusammen mit den jeweiligen aktuellen Werten angezeigt. So wurde z.B. in der Abbildung oben der Aufruf der Funktion `local:plus` erreicht. Die Variable `$i` befindet sich in der Funktion `local:loop` im Geltungsbereich und wurde soeben auf `2` inkrementiert, daher wird `$i` mit ihrem aktuellen Wert angezeigt. Die Variablen `$a` und `$b` befinden sich nicht in `local:loop` und somit auch nicht im Geltungsbereich. Sie werden daher auch nicht angezeigt.

Überwachungsausdrücke

Überwachungsausdrücke sind Ausdrücke, die Sie entweder vor Beginn einer Auswertung oder während einer Unterbrechung der Auswertung eingeben können. Sie können für folgende Zwecke verwendet werden:

- um bestimmte Bedingungen zu überprüfen. So wird z.B. in der Abbildung unten mit dem Überwachungsausdruck `$i=1 to 5` überprüft, ob die Variable `$i` einen Wert im angegebenen Bereich hat. Das Ergebnis `true` besagt, dass diese Bedingung erfüllt wird.

- um Daten in einem bestimmten Kontext zu suchen. So könnten wir z.B. im Kontext eines Elements `Company` den Überwachungsausdruck `@id` eingeben, um den Kundencode des Unternehmens im XML-Zieldokument nachzuschlagen.
- um zusätzliche Daten zu generieren. So könnte etwa mit dem Überwachungsausdruck `if ($i=1 to 5) then (concat("Iteration ", $i)) else "Out of Loop"` in der Abbildung oben ein passender String generiert werden, mit dem angegeben wird, in welcher Iteration der Schleife sich die Auswertung gerade befindet.

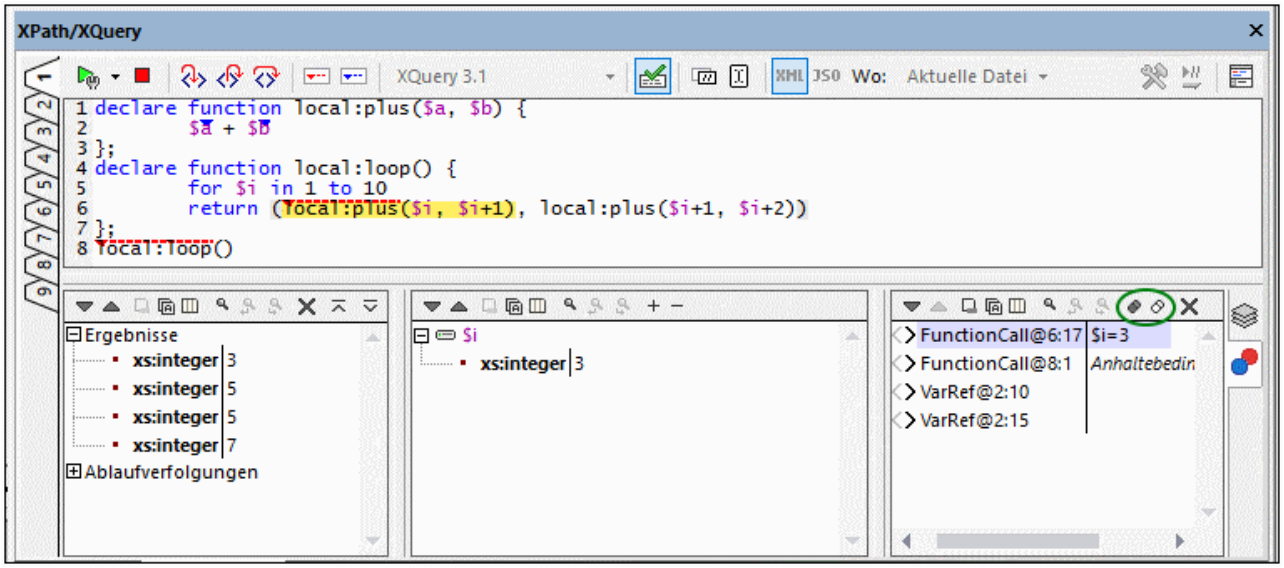
Um einen Überwachungsausdruck einzugeben, klicken Sie in der Symbolleiste des Fensters auf **Überwachung hinzufügen** (*in der Abbildung oben rot umrandet*), geben Sie anschließend den Ausdruck ein und klicken Sie anschließend auf **Eingabe**. Um einen Überwachungsausdruck zu entfernen, wählen Sie ihn aus und klicken Sie in der Symbolleiste auf **Ausgewählten Überwachungsausdruck entfernen**. Wenn der Ausdruck während des Debuggens aus irgendeinem Grund (z.B. wenn sich eine seiner Variablen nicht im Geltungsbereich befindet) nicht korrekt ausgewertet werden kann, wird der Überwachungsausdruck rot angezeigt.

Call Stack

Auf dem Register *Call Stack* des Call Stack- und Debug-Punkte-Bereichs (*rechter unterer Bereich in der Abbildung oben*) werden die Prozessoraufrufe bis zu diesem Punkt im Debugging angezeigt. Der aktuelle Prozessoraufruf erscheint gelb markiert. Beachten Sie, dass nur die Aufrufe, die direkt zum aktuellen Auswertungsschritt geführt haben, angezeigt werden. So wurde z.B. in der Abbildung oben der Aufruf der Funktion `local:plus` erreicht. Obwohl dies die zweite Iteration von `local:loop` ist, werden die Prozessoraufrufe der ersten Iteration **nicht** angezeigt, da sich diese Aufrufe auf einer Parallelebene zur aktuellen Funktion befinden und nicht zu dieser Funktion geführt haben.

Debug-Punkte

Auf dem Register Debug-Punkte des Call Stack- und Debug-Punkte-Bereichs (*rechter unterer Bereich in der Abbildung oben*) werden die im Ausdruck gesetzten Breakpoints (rote Kreise) und Tracepoints (blaue Kreise) angezeigt. Jeder Debug-Punkt wird mit der jeweiligen Zeilen- und Zeichennummer angezeigt. So bedeutet etwa `FunctionCall@6:17`, dass sich in Zeile 6, Zeichen 17 ein Debug-Punkt befindet.



Beachten Sie die folgenden Funktionalitäten:

- Durch Doppelklick auf *Break-Bedingung eingeben*, Eingabe des Ausdrucks für die Bedingung und Drücken der **Eingabetaste** können Sie eine **Break-Bedingung** für Breakpoints eingeben. Dieser Breakpoint ist nur dann aktiv, wenn das Resultat der Auswertung der Bedingung `true` ist. So aktiviert die Break-Bedingung `$i=3` in der Abbildung oben den Breakpoint in diesem Funktionsaufruf nur, wenn der Wert von `$i 3` ist. In der Abbildung sehen Sie, dass die Auswertung an diesem Breakpoint angehalten wurde.
- Durch Klick auf die entsprechende Symbolleisten-Schaltfläche **Alle Debug-Punkte aktivieren** und **Alle Debug-Punkte deaktivieren** *(grün umrandete Schaltflächen in der Abbildung oben)* können Sie alle Debug-Punkte aktivieren bzw. deaktivieren. Wenn ein Debug-Punkt deaktiviert ist, bleibt er bis zu seiner erneuten Aktivierung für alle Auswertungen deaktiviert.
- Einzelne Breakpoints können im dazugehörigen Kontextmenü aktiviert bzw. deaktiviert werden.

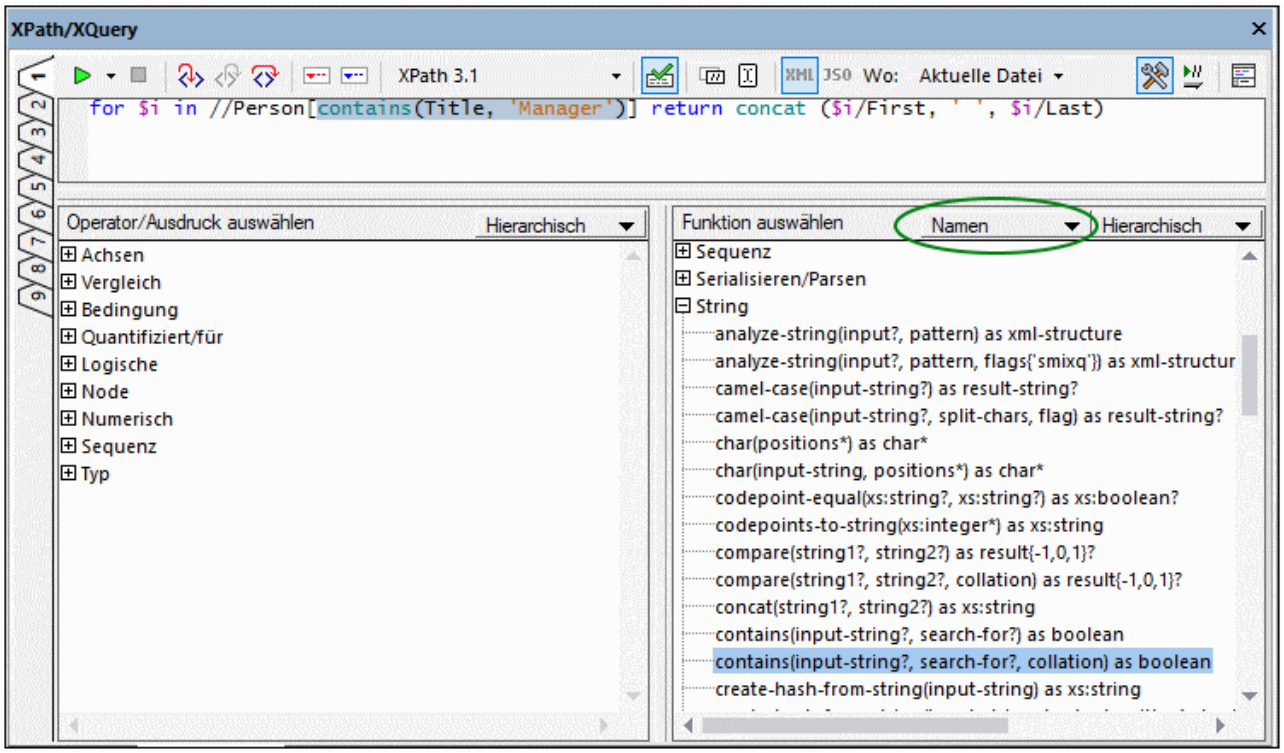
Symbolleisten-Schaltflächen in den Bereichen

Die Bereiche des "XPath/XQuery-Fensters im Debug-Modus (*siehe Abbildung oben*) enthalten Schaltflächen für die Navigation, Suche und Kopierfunktionen. Diese Schaltflächen werden in der Tabelle unten von links nach rechts beschrieben. Die dazugehörigen Befehle finden Sie auch im Kontextmenü der aufgelisteten Einträge.

Symbol	Funktion
Weiter, Zurück	Wählt jeweils den nächsten bzw. vorhergehenden Eintrag in der Ergebnisliste aus
Ausgewählte Textzeile in die Zwischenablage kopieren	Kopiert die Wertespalte des ausgewählten Ergebniseintrags in die Zwischenablage. Um alle Spalten zu kopieren, aktivieren Sie den Befehl *Beim Kopieren werden alle Spalten inkludiert (siehe unten)*
Alle Nachrichten in die Zwischenablage kopieren	Kopiert die Wertespalte aller Ergebniseinträge, darunter auch leere Werte, in die Zwischenablage. Jeder Eintrag wird als separate Zeile kopiert.
Beim Kopieren werden alle Spalten inkludiert	Wechselt zwischen der Funktion zum Kopieren (i) aller Spalten und (ii) nur der Wertespalte. Das Spaltentrennzeichen ist ein einziges Leerzeichen.
Suchen	Öffnet ein *Suchdialogfeld* zum Suchen nach einem beliebigen String - darunter auch von Sonderzeichen - in der Ergebnisliste.
Vorheriges suchen	Sucht nach der vorhergehenden Instanz des zuletzt im *Suchdialogfeld* gesuchten Eintrags.
Weitersuchen	Sucht nach der nächsten Instanz des zuletzt im *Suchdialogfeld* gesuchten Eintrags.
Zurücksetzen	Löscht die Ergebnisliste.
Mehrzeilige Ergebnisse auf eine einzige Zeile reduzieren	Wenn die Wertespalte eines Ergebniseintrags mehrzeiligen Text (Text der Zeilenvorschubzeichen beinhaltet) enthält, können Sie zwischen einer mehrzeiligen und einer einzeiligen Anzeige wechseln.
Vollständiges Ergebnis anzeigen	Zeigt den gesamten Inhalt von Nodes als Wert des Node an.
Kopfzeile in der Ausgabe anzeigen	Zeigt den Pfad der XML-Datei(en) und den XPath-Ausdruck an.

10.4 Ausdruckserstellung

Die Ausdruckserstellung (oder Erstellungsmodus) wird durch Klick auf das Symbol **Erstellungsmodus** der Hauptsymbolleiste  (*siehe Abbildung unten*) ein- und ausgeschaltet. Der Erstellungsmodus kann in beiden Modi (Auswertungsmodus und Debug-Modus 583) aktiviert werden. Er verfügt über zwei Eingabehilfefenster: (i) für Operatoren und Ausdrücke und (ii) für Funktionen (*siehe Abbildung unten*). Die Einträge in den beiden Bereichen können entweder hierarchisch gruppiert oder als flache Liste angezeigt werden. Wählen Sie die gewünschte Option in der Dropdown-Liste rechts oben im jeweiligen Bereich aus (*in der Abbildung unten*). In der Abbildung werden die Einträge in beiden Bereichen in hierarchischen Gruppen angezeigt.



Funktionen des Bereichs "Erstellung":

- Um eine Textbeschreibung eines Eintrags in einem der Eingabehilfebereiche zu sehen, platzieren Sie den Mauszeiger darüber.
- Jede Funktion wird mit ihrer Signatur angezeigt (d.h. mit ihren Argumenten, den Datentypen der Argumente und dem Datentyp der Funktionsausgabe).
- Wenn es zu einem einzigen Funktionsnamen mehrere Signaturen gibt, so wird jede Signatur als separate Funktion aufgelistet. (Diese Varianten werden als Überladungen dieses Funktionsnamens bezeichnet.) Z.B. wird die Funktion **`contains`** in der Abbildung oben zwei Mal angezeigt, für jede ihrer beiden Signaturen ein Mal.
- Die Argumente werden entweder nach dem Namen (falls vorhanden) oder dem Datentyp aufgelistet. Wählen Sie die gewünschte Option aus der Dropdown-Liste in der Titelleiste des Funktionsfensters (*in der Abbildung oben grün umrandet*) aus.

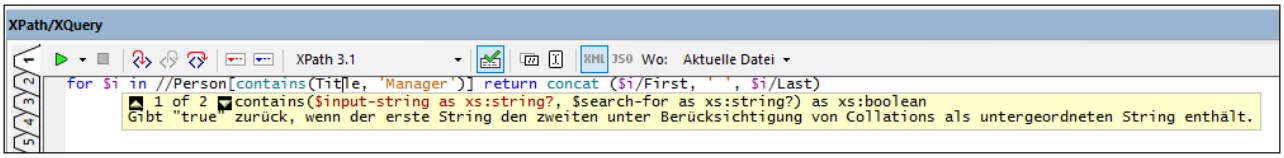
© 2016-2022 Altova GmbH

- Wenn Sie in einem der Bereiche auf einen Eintrag doppelklicken (Operator, Ausdruck oder Funktion), so wird dieser an der Cursorposition in den Ausdruck eingefügt. Funktionen werden mit ihren Argumenten in Form von Platzhaltern (#-Symbol) eingefügt.
- Wenn (i) im Bearbeitungsfeld des Ausdrucks Text ausgewählt ist und (ii) Sie auf einen Operator, einen Ausdruck oder eine Funktion mit einem Platzhalter doppelklicken, um sie einzufügen, so wird anstelle des Platzhalters der ausgewählte Text eingefügt. Dies ist eine Möglichkeit, langen Text (wie z.B. einen Pfadausdruck) in einen Operator, einen Ausdruck oder eine Funktion schnell einzufügen.
- Sie können den Pfad zu einem Node im aktiven Dokument einfügen, indem Sie den entsprechenden Node auswählen und anschließend in der Hauptsymbolleiste des Fensters auf die Schaltfläche **XPath von der aktuellen Auswahl in das Bearbeitungsfeld kopieren** klicken.

Symbolleisten-Schaltflächen für die Ausdruckserstellung

	**Horizontales/Vertikales Layout**	Wechselt zwischen dem horizontalen und dem vertikalen Layout.
	**Zum Erstellungsmodus wechseln**	Wechselt in den Ausdruckserstellungsmodus, in dem Ihnen bei der Erstellung von Ausdrücken kontextsensitive Eingabehilfen zur Verfügung stehen.
	**XPath der aktuellen Auswahl kopieren**	Kopiert den Locator-Pfad des Node im XML-Dokument im Ausdrucksfenster an die letzte Cursor-Position.
	**Aktuelle Auswahl als Kontext festlegen**	Wechselt beim Auswertungskontext zwischen dem Root-Node und der aktuellen Auswahl.
	**XPath/XQuery-Fragment laden/speichern**	Lädt bzw. speichert ein XPath/XQuery-Fragment aus bzw. in einer XQuery-Datei

Nachdem Sie eine Funktion in den Ausdruck eingefügt haben, werden bei Platzieren der Maus über den Funktionsnamen die Signatur der Funktion und eine Textbeschreibung der Funktion angezeigt. Wenn für eine Funktion mehrere Signaturen vorhanden sind, so werden diese am unteren Rand der Anzeige mit einem Überladungsfaktor angezeigt. Wenn Sie den Cursor innerhalb der Klammern der Funktion platzieren und **Strg+Umschalt+Leertaste** drücken, sehen Sie die Signaturen der verschiedenen Überladungen dieses Funktionsnamens (*siehe Abbildung unten*).



Öffnen und Speichern von XPath/XQuery-Fragmenten aus bzw. in einer Datei

Sie können einen XPath/XQuery-Ausdruck oder längere Fragmente, die Sie in das XPath/XQuery-Fenster eingegeben haben, zusammen mit den aktuellen Einstellungen des Fensters in einer XQuery-Datei speichern und XPath/XQuery-Fragmente aus einer XQuery-Datei laden. Um diese beiden Funktionen auszuführen, klicken Sie in der Symbolleiste des Fensters auf das entsprechende Symbol (Snippet laden/speichern) .



Nachdem ein XPath/XQuery-Fragment in einer Datei gespeichert wurde, kann es in das XPath/XQuery-Fenster jeder beliebigen XMLSpy-Instanz (ab Version 2022) geladen werden. Dies ist nützlich, wenn Sie das Fragment auf einem anderen Rechner verwenden, es an einen anderen Benutzer weitergeben oder es später selbst auf demselben Rechner wiederverwenden möchten. Wenn der Ausdruck aus einer XQuery-Datei in das XPath/XQuery-Fenster geladen wurde, wechseln die Einstellungen des Fensters automatisch zu den Fenstereinstellungen, die mit dem Fragment in der Datei gespeichert wurden.

Fragment in Datei speichern
Um einen XPath/XQuery-Ausdruck oder ein XPath/XQuery-Fragment in einer Datei zu speichern, gehen Sie folgendermaßen vor:

1. Definieren Sie die Fenstereinstellungen: (i) die Auswertungssprache; (ii) die *Wo*-Feld-Einstellung und (iii), falls für das Feld *Wo Projekt* ausgewählt wurde, ob externe Ordner übersprungen werden sollen oder nicht.
2. Geben Sie den zu speichernden XPath/XQuery-Ausdruck bzw. das entsprechende Fragment ein.
3. Klicken Sie auf **Snippet speichern**.
4. Wählen Sie im daraufhin angezeigten Dialogfeld *Speichern unter* die Datei aus, oder geben Sie den Namen einer neuen Datei, in der das Fragment gespeichert werden soll, ein und klicken Sie auf **Speichern**.

Das Fragment wird in der Datei gespeichert, wobei die Einstellungen des Fensters als Kommentar in der ersten Zeile der Datei gespeichert werden. Unten sehen Sie das Codefragment einer gespeicherten Fragment-Datei.

```
(: {"language":"XQuery3","where":"CurrentFile"} :)
for $i in //para[count(*)=0] return $i
```

Anmerkung: Sie können diese Datei einschließlich der Kommentarzeile, die die Fenstereinstellungen enthält, auf Wunsch bearbeiten.

Fragment aus Datei laden
Um ein XPath/XQuery-Fragment aus einer Datei zu laden, gehen Sie folgendermaßen vor:

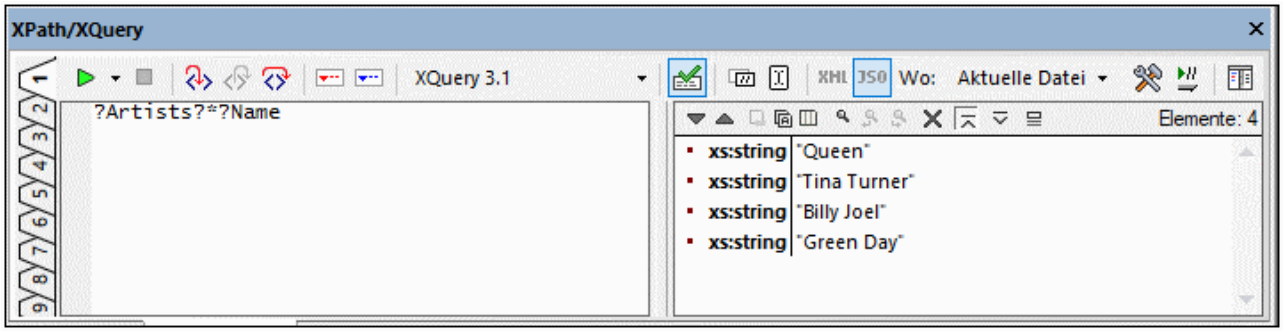
1. Klicken Sie auf dem Register, in das Sie den Ausdruck laden möchten, auf **Snippet laden**.
2. Navigieren Sie im daraufhin angezeigten *Öffnen*-Dialogfeld zur gewünschten XQuery-Datei und klicken Sie auf **Öffnen**.

Das Fragment wird daraufhin geladen und die Einstellungen des XPath/XQuery-Fensters werden in die in der XQuery-Datei definierten geändert. Wenn in der Datei keine Einstellungen gespeichert wurden, bleiben die Einstellungen im XPath/XQuery-Fenster unverändert. Wenn die Wo-Einstellung falsch ist, wird die Fenstereinstellung auf die Standardeinstellung *Aktuelle Datei* gesetzt.

© 2016-2022 Altova GmbH

10.5 XQuery-Ausdrücke für JSON

JSON-Dokumente können durch Eingabe eines XPath/XQuery 3.1-Abfrageausdrucks im XPath/XQuery-Ausgabefenster[124] (*siehe Abbildung unten*) abgefragt werden.



Um einen Ausdruck anhand eines JSON-Dokuments auszuwerten, gehen Sie folgendermaßen vor:

1. Wählen Sie entweder das **XPath 3.1-** oder das **XQuery 3.1**-Symbol aus.
2. Stellen Sie sicher, dass sich das Fenster im JSON-Auswertungsmodus befindet.
3. Geben Sie den XPath 3.1- oder XQuery 3.1-Ausdruck ein.
4. Klicken Sie (links in der Symbolleiste) auf **Auswertung starten**.

XQuery 3.1-Ausdrücke für JSON

Da in JSON-Datenstrukturen oft Objekte und Arrays verwendet werden, wird zum Identifizieren von Nodes in JSON-Objekten (bei denen es sich aus Sicht von XQuery im Grunde um Zuordnungen handelt) und JSON-Arrays der XQuery 3.1 Lookup-Operator `?` verwendet. Diese Methode der Node-Suche unterscheidet sich von der Methode, mit der Pfadausdrücke zum Suchen von Nodes in XML-Dokumenten geschrieben werden. In diesen Dokumenten werden Schritte in einem Pfadausdruck (z.B: `items/*`) mit Hilfe des Schrägstrich-Operators `/` verbunden. In XQuery-Ausdrücken für JSON wird der Schrägstrich-Operator nicht zur Suche von Nodes verwendet.

Beispiele für XQuery-Ausdrücke für JSON

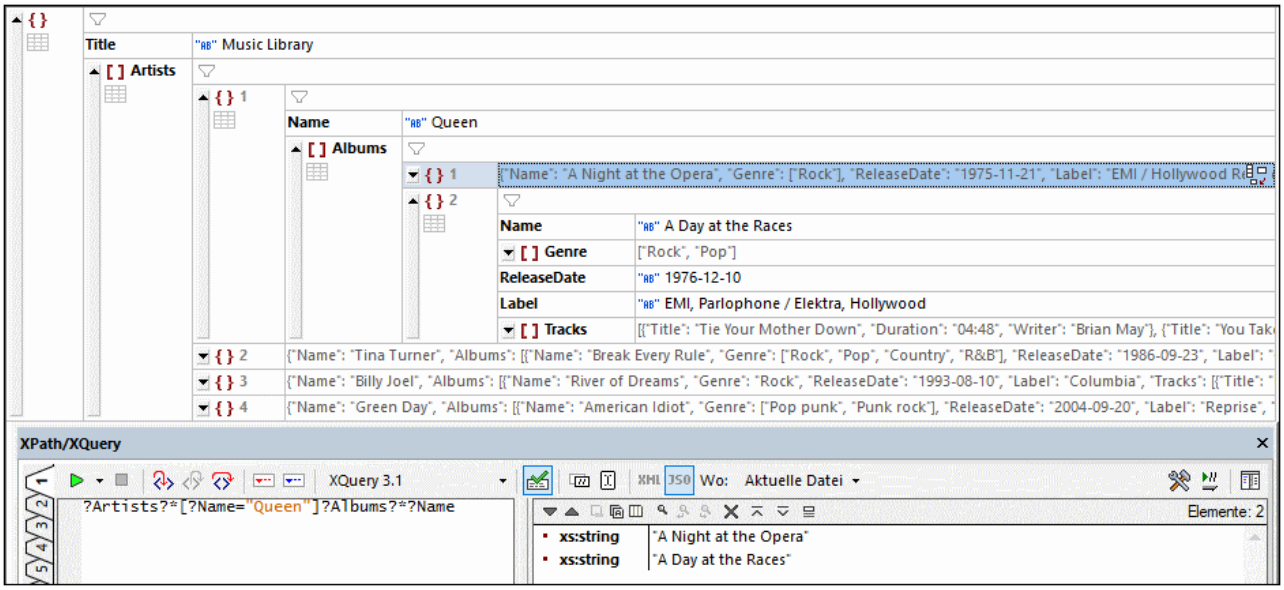
`?items?*`
Bedeutet: Suche den Child-Node `items` und anschließend alle diesem untergeordnete Nodes. Beachten Sie, dass erwartet wird, dass `items` ein Child-Node des Kontext-Node ist.

`?Artists?1?Albums?2?Name`
Bedeutet: Suche den Child-Node `Artists` und anschließend dessen ersten Child-Node. Suche innerhalb dieses Node den Child Node `Albums` und anschließend dessen zweiten Child Node. Gib nun den Node `Name` dieses zweiten Child Node zurück.

`?Tracks?*[contains(?Writer, 'Brian')]`
Bedeutet: Suche den Child Node `Tracks` und anschließend alle seine untergeordneten Nodes. Suche während der Suche der untergeordneten Nodes die einzelnen `Writer` Node Children und wähle nur diejenigen aus, die den String `'Brian'` enthalten. Beachte, dass dieser Ausdruck drei Lookup-Operatoren enthält. Jeder davon wird in einem neue Schritt, in dem ein Nodeset gesucht werden muss, verwendet.

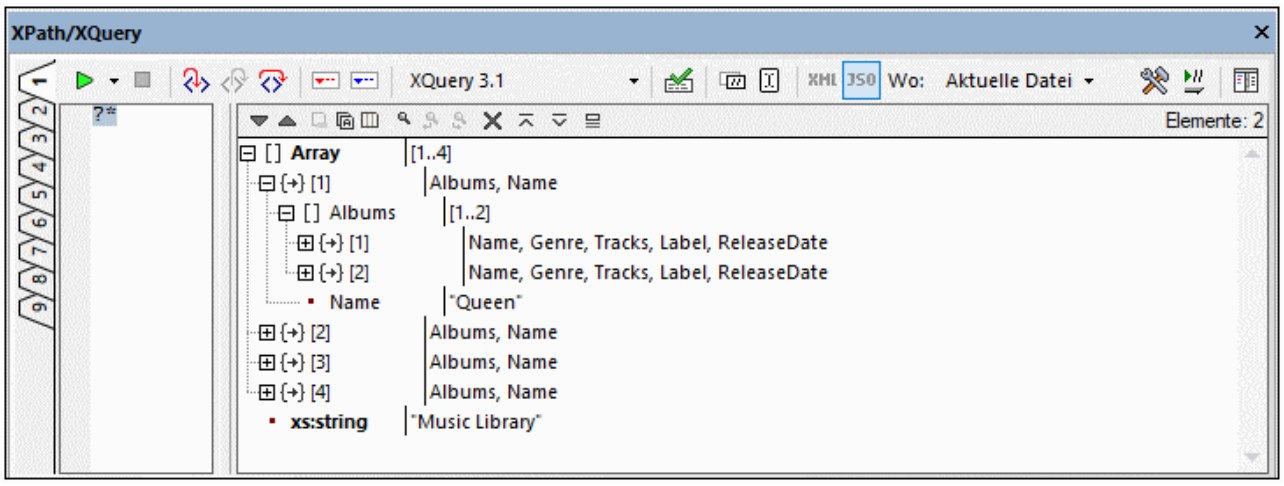
`?Artists?*[?Name="Queen"]?Albums?*?Name`

Bedeutet: Suche im Root-Objekt den Child Node **Artists** und anschließend alle dessen Children, die einen **Name** Node mit dem Wert **"Queen"** haben. Suche in diesen Nodes alle untergeordneten Album-Nodes und anschließend deren untergeordnete Nodes. Suche in diesen untergeordneten Nodes die jeweiligen **Name** Nodes und gib diese zurück. In der Abbildung unten sehen Sie diesen Ausdruck im XPath/XQuery-Fenster[582] zusammen mit dem JSON-Zieldokument in der JSON Grid-Ansicht[690].



Ergebnisbereich für die JSON-Auswertung

Im Ergebnisbereich werden die ausgewählten JSON-Komponenten auf der linken Seite des Ergebnisbereichs fett angezeigt und der Wert der Komponente wird auf der rechten Seite des Bereichs angezeigt. In der Abbildung unten werden die Ergebnisse fett angezeigt. Das Array wurde erweitert, sodass Sie dessen Mitglieder sehen.



Zuordnungen und Arrays werden je nachdem, ob die Symbolleisten-Schaltfläche *Vollständiges Ergebnis anzeigen* aktiviert oder deaktiviert ist, im Kurzformat oder im ausführlichen Format angezeigt. Die Zuordnungen

© 2016-2022 Altova GmbH

und Arrays auf der linken Seite des Bereichs können durch Klick auf das entsprechende Plus-Symbol erweitert werden.

10.6 Anmerkungen

XPath 1.0-Ausdrücke

- XPath 1.0-Funktionen müssen ohne Namespace-Präfix eingegeben werden.
- Es werden die vier Node-Tests nach Typ unterstützt: `node()`, `text()`, `comment()` und `processing-instruction()`.

XPath 2.0- und 3.1-Ausdrücke

- Es werden String- (z.B. 'Hello') und numerische Literale (z.B. 256) unterstützt. Zum Erstellen anderer Literale auf Basis vom XML-Schema-Typen wird ein Konstruktor verwenden, dem ein Namespace vorangestellt wird (z.B. `xs:date('2004-09-02')`). Das für XML-Schema-Typen verwendete Namespace-Präfix muss an den XML-Schema-Namespace gebunden werden: `http://www.w3.org/2001/XMLSchema` und dieser Namespace muss in Ihrer XML-Datei deklariert sein.
- Die vom XPath Evaluator verwendeten XPath 2.0- und 3.1-Funktionen gehören zum Namespace `http://www.w3.org/2005/xpath-functions`. Normalerweise ist das Präfix `fn:` an diesen Namespace gebunden. Da es sich bei diesem Namespace jedoch um den vom XPath Evaluator verwendeten Standard-Namespace für Funktionen handelt, müssen Sie bei Funktionen kein Präfix definieren. Wenn Sie ein Präfix verwenden, stellen Sie sicher, dass es an den Namespace für XPath-Funktionen gebunden ist, den Sie im XML-Dokument deklarieren müssen. Beispiele für die Verwendung einer Funktion: `current-date()` (wobei der Funktions-Namespace im XML-Dokument nicht deklariert ist); `fn:current-date()` (wobei der Funktions-Namespace im XML-Dokument nicht deklariert ist oder im XML-Dokument deklariert ist und an das Präfix `fn:` gebunden ist). Sie können das Namespace-Präfix sogar weglassen, wenn der Funktions-Namespace im XML-Dokument mit oder ohne Präfix deklariert ist; denn eine Funktion, die in einem XPath-Ausdruck so verwendet wird, befindet sich im Standard-Namespace. Dies ist der Standard-Namespace für Funktionen.
- Die Altova-XPath-Erweiterungen befinden sich im Namespace **`http://www.altova.com/xslt-extensions`**.

Anmerkung: Zusammenfassend ist zu Namespaces zu sagen: Bei Verwendung von Konstruktoren oder Typen aus dem XML-Schema-Namespace müssen Sie den XML-Schema-Namespace im XML-Dokument deklarieren und im XPath-Ausdruck die korrekten Namespace-Präfixes verwenden. für XPath-Funktionen müssen Sie kein Präfix verwenden.

Datentypen in XPath 2.0 und 3.1

Wenn Sie einen XPath 2.0- oder 3.1-Ausdruck für ein XML-Dokument, das ein XML-Schema referenziert und gemäß diesem Schema gültig ist, auswerten, müssen Sie Datentypen, die nicht durch eine Operation implizit in den benötigten Datentyp konvertiert werden, explizit konstruieren oder konvertieren. In dem vom integrierten XPath-Prozessor verwendeten XPath 2.0- und 3.1-Datenmodell werden alle **atomisierten** Node-Werte aus dem XML-Dokument dem Datentyp `xs:untypedAtomic` zugewiesen. Der Typ `xs:untypedAtomic` funktioniert gut bei impliziten Typkonvertierungen. So hat der Ausdruck `xs:untypedAtomic("1") + 1` als Ergebnis den Wert 2, da der `xs:untypedAtomic`-Wert implizit durch den Additions-Operator in `xs:double` umgewandelt wird. Arithmetische Operatoren wandeln Operanden implizit in `xs:double` um. Vergleichsoperatoren wandeln Operanden vor dem Vergleich in `xs:string` um.

© 2016-2022 Altova GmbH

In einigen Fällen müssen Datentypen jedoch explizit in den benötigten Datentyp konvertiert werden. Wenn Sie z.B. zwei Elemente haben, `startDate` und `endDate`, die im XML-Schema als Typ `xs:date` definiert sind, wird z.B. bei Verwendung des XPath 2.0 Ausdrucks `endDate - startDate` ein Fehler angezeigt. Wenn Sie dagegen `xs:date(endDate) - xs:date(startDate)` oder `(endDate cast as xs:date) - (startDate cast as xs:date)` verwenden, wird der Ausdruck korrekt zu einer einzigen Sequenz vom Typ `xs:dayTimeDuration` ausgewertet.

Anmerkung: Die XPath-Prozessoren, die vom XPath Evaluator verwendet werden, werden auch vom Altova XSLT-Prozessor verwendet. XPath 2.0- oder 3.1-Ausdrücke in XSLT Stylesheets, die nicht implizit in den erforderlichen Datentyp konvertiert werden, müssen explizit als der benötigte Datentyp konstruiert oder konvertiert werden.

Stringlänge der Zeichen- und Entity-Referenzen

Bei Verwendung von Zeichen- und Entity-Referenzen als Eingabestring für die Funktion `string-length()` können die Referenzen nicht aufgelöst werden und es wird die Länge des nicht aufgelösten Textstring zurückgegeben. In einer XSLT-Umgebung hätten diese Referenzen allerdings eine Bedeutung und es wird die Länge des aufgelösten String zurückgegeben.

Unterstützung von XPath 2.0- und 3.1-Funktionen

Siehe Anhänge 2143.

11 Authentic

Die Authentic-Ansicht (siehe Abbildung unten) ist eine grafische Darstellung Ihres XML-Dokuments. Sie können damit XML-Dokumente **ohne** Markup-Code und **mit** den entsprechenden Formatierungs- und Dateneingabefunktionen wie z.B. Eingabefeldern, Listenfeldern und Optionsfeldern darstellen. Die Daten, die der Benutzer in der Authentic-Ansicht eingibt, werden in die XML-Datei eingegeben.



In der Authentic-Ansicht lassen sich XML-Dokumente und Datenbanken (DB) über eine einfache Benutzeroberfläche problemlos editieren.

Um ein XML-Dokument in der Authentic-Ansicht anzeigen und bearbeiten zu können, muss es mit einem **StyleVision Power Stylesheet** verknüpft werden, das in Altova <%SPY-STYLE%> erstellt wird. Ein SPS (`.sps`-Datei) ist im Grunde ein XSLT Stylesheet. Es definiert eine Ausgabedarstellung für ein XML-Dokument, das auch Dateneingabemechanismen enthält. Benutzer der Authentic-Ansicht können daher Daten zurück in die XML-Datei oder die DB schreiben. Ein StyleVision Power Stylesheet basiert auf einem Schema und ist schemaspezifisch. Wenn Sie ein SPS verwenden möchten, um eine XML-Datei in der Authentic-Ansicht zu editieren, muss es sich dabei um ein SPS handeln, das auf demselben Schema basiert, wie dasjenige, auf dem die XML-Datei basiert.

Verwendung der Authentic-Ansicht

- Wenn eine XML-Datei geöffnet ist, können Sie durch Klicken auf die Schaltfläche **Authentic** am unteren Rand des Dokumentenfensters in die Authentic-Ansicht wechseln. Wenn der XML-Datei noch kein SPS

© 2016-2022 Altova GmbH

zugewiesen wurde, werden Sie aufgefordert, dies jetzt zu tun. Es muss ein SPS verwendet werden, das auf demselben Schema basiert, wie die XML-Datei.

- Wenn Sie den Befehl **Datei | Neu** wählen und anschließend auf die Schaltfläche "StyleVision Stylesheet auswählen..." klicken, wird in der Authentic-Ansicht eine neue XML-Datei erstellt und angezeigt. Diese neue Datei ist eine Vorlagendatei, die mit dem SPS verknüpft ist, das Sie öffnen. Sie kann bereits verschiedene Ausgangsdaten enthalten. Diese Ausgangsdaten befinden sich in einer XML-Datei (einer XML-Vorlagendatei), die optional mit dem SPS verknüpft werden kann. Nachdem die Authentic-Ansicht einer XML-Datei nun angezeigt wird, können Sie Daten darin eingeben und die Datei speichern.
- Sie können ein SPS auch mit dem Befehl **Authentic | Neues Dokument** öffnen. Wenn dem SPS eine Vorlagendatei zugewiesen wurde, so werden die Daten in der XML-Vorlagendatei als Ausgangsdaten für die in der Authentic-Ansicht erstellte XML-Dokumentvorlage verwendet.

In diesem Abschnitt

Dieser Abschnitt enthält ein Tutorial zur Authentic-Ansicht, in dem beschrieben wird, wie Sie die Authentic-Ansicht verwenden. Es folgt der Abschnitt "Bearbeiten in der Authentic-Ansicht", in dem die einzelnen Bearbeitungsfunktionen näher erläutert werden.

Weitere Informationen zur Authentic-Ansicht

Nähere Informationen zur Authentic-Ansicht finden Sie (i) in dieser Dokumentation im Abschnitt Bearbeitungsansichten | Authentic-Ansicht 622, in dem das Bearbeitungsfenster der Authentic-Ansicht beschrieben wird, und (ii) im Abschnitt Menü "Authentic" 1428 des Abschnitts Menübefehle.

11.1 Tutorial Authentic-Ansicht

In der Authentic-Ansicht können Sie XML-Dokumente über eine grafische WYSIWYG-Oberfläche (*Abbildung unten*) wie in einer Textverarbeitungsanwendung, z.B. Microsoft Word, bearbeiten. Sie müssen dabei nur die Daten eingeben und sich nicht um die Formatierung des Dokuments kümmern, da die Formatierung bereits im Stylesheet definiert ist, das festlegt, wie das XML-Dokument in der Authentic-Ansicht dargestellt wird. Das Stylesheet (StyleVision Power Stylesheet, in diesem Tutorial als SPS bezeichnet) wird im Produkt Altova StyleVision erstellt:



Ein XML-Dokument kann in der Authentic in zweierlei Hinsicht bearbeitet werden: (i) Sie können die Struktur des Dokuments bearbeiten (z.B. Dokumentteile wie z.B. Absätze oder Überschriften hinzufügen oder löschen), und (ii) Sie können Daten eingeben (den Inhalt der Dokumentteile).

Dieses Tutorial enthält eine schrittweise Anleitung zu Folgendem:

- Öffnen einer bestehenden XML-Datei in der Authentic-Ansicht. Damit ein Dokument in der Authentic -Ansicht bearbeitet werden kann, mit es mit einer SPS-Datei verknüpft sein.
- Aussehen der Benutzeroberfläche der Authentic-Ansicht und eine allgemeine Beschreibung der wichtigsten Bearbeitungsfunktionen.
- Bearbeiten der Dokumentstruktur durch Einfügen und Löschen von Nodes.
- Eingabe von Daten in das XML-Dokument.
- Eingabe von (i) Attributwerten über die Eingabehilfe "Attribute" und (ii) Entity-Werten
- Drucken des Dokuments

© 2016-2022 Altova GmbH

Denken Sie daran, dass dieses Tutorial als Einführung gedacht ist und daher nicht auf komplizierte Aspekte eingeht. Zusätzliches Referenzmaterial und nähere Informationen zu einzelnen Funktionen finden Sie in den Abschnitten zur Oberfläche der Authentic-Ansicht 622.

Voraussetzungen für das Tutorial

Sie finden alle für das Tutorial benötigten **Dateien** im Ordner `Examples` des Altova-Applikationsordners. Es sind die Dateien:

- `NanonullOrg.xml` (das XML-Dokument, das Sie öffnen)
- `NanonullOrg.sps` (das StyleVision Power Stylesheet, mit dem das XML-Dokument verknüpft ist)
- `NanonullOrg.xsd` (das XML-Schema, auf dem das XML-Dokument und das StyleVision Power Stylesheet basieren und mit dem diese verknüpft sind)
- `nanonull.gif` und `Altova_right_300.gif` (zwei Bilddateien, die in dem Tutorial verwendet werden)

Anmerkung: An einigen Stellen des Tutorials werden Sie gebeten, sich den XML-Text des XML-Dokuments anzusehen (also nicht die Authentic-Ansicht des Dokuments). Wenn die von Ihnen verwendete Altova Produktversion keine Textansicht enthält (dies ist bei der kostenlosen Authentic Desktop und der Authentic Browser der Fall), verwenden Sie einen einfachen **Texteditor**, wie Wordpad oder Notepad, um den Text des XML-Dokuments anzuzeigen.

Vorsicht: Verwenden Sie für das Tutorial am besten eine Kopie der Datei `NanonullOrg.xml`, damit Sie das Original im gegebenen Fall jederzeit wieder aufrufen können.

11.1.1 Öffnen eines XML-Dokuments in der Authentic-Ansicht

Sie können in der Authentic-Ansicht ein vorhandenes XML-Dokument bearbeiten oder ein neues XML-Dokument erstellen. In diesem Tutorial werden Sie ein vorhandenes XML-Dokument in der Authentic-Ansicht öffnen (wie in diesem Abschnitt beschrieben) und erfahren, wie Sie es bearbeiten können (Beschreibung siehe nächste Abschnitte). Außerdem erfahren Sie in diesem Abschnitt, wie ein neues XML-Dokument erstellt werden kann, um in der Authentic-Ansicht bearbeitet zu werden.

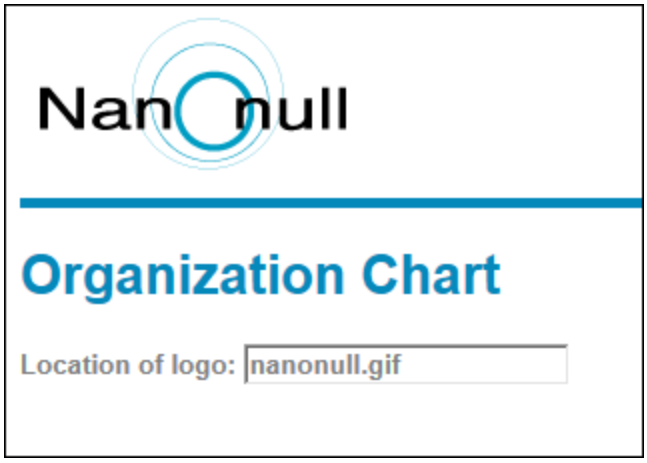
Öffnen eines vorhandenen XML-Dokuments

Die Datei, die Sie öffnen werden, heißt `NanonullOrg.xml`. Sie befindet sich im Ordner `Examples` Ihrer Altova-Applikation. Es gibt zwei Möglichkeiten, um `NanonullOrg.xml` zu öffnen:

- Klicken Sie in Ihrem Altova-Produkt auf **Datei | Öffnen**, suchen Sie anschließend die Datei `NanonullOrg.xml` im Dialogfeld, das nun angezeigt wird und klicken Sie auf **Öffnen**.
- Suchen Sie die Datei im Windows Explorer, rechtsklicken Sie darauf und wählen Sie Ihr Altova-Produkt als die Applikation aus, mit der die Datei geöffnet werden soll.

Die Datei **NanonullOrg.xml** wird direkt in der Authentic-Ansicht (*Abbildung unten*) geöffnet. Dies hat folgende Gründe:

- Der Datei wurde bereits ein StyleVision Power Stylesheet (SPS) zugewiesen und
- im Dialogfeld "Optionen" (Extras | Optionen) auf dem Register "Ansicht" ist die Option, XML-Dateien in der Authentic-Ansicht zu öffnen, wenn eine SPS-Datei zugewiesen wurde, aktiviert. (Andernfalls würde die Datei in der Textansicht geöffnet).



Beachten Sie: Wie ein XML-Dokument in der Authentic-Ansicht angezeigt wird, wird im SPS definiert. Ohne ein SPS kann das Dokument nicht in der Authentic-Ansicht dargestellt werden.

Erstellen eines neuen XML-Dokuments auf Basis eines SPS

Sie können auch ein neues XML-Dokument erstellen, das auf einem SPS basiert. Dabei gibt es zwei Möglichkeiten: über den Menübefehl **Datei | Neu** und über den Menübefehl **Authentic | Neues Dokument**. In beiden Fällen wird ein SPS ausgewählt.

Über Datei | Neu

1. Klicken Sie auf **Datei | Neu**.
2. Wählen Sie im Dialogfeld "Neues Dokument erstellen" das gewünschte SPS aus.

Wenn dem StyleVision Power Stylesheet eine XML-Vorlagendatei zugewiesen wurde, werden die Daten in der XML-Vorlagendatei als Ausgangsdaten der in der Authentic-Ansicht erstellten Vorlage verwendet.

Über Authentic | Neues Dokument

1. Klicken Sie auf **Authentic | Neues Dokument**.
2. Wählen Sie im Dialogfeld "Neues Dokument erstellen" das gewünschte SPS aus.

Wenn dem StyleVision Power Stylesheet eine XML-Vorlagendatei zugewiesen wurde, werden die Daten in der XML-Vorlagendatei als Ausgangsdaten der in der Authentic-Ansicht erstellten Vorlage verwendet.

11.1.2 Die Benutzeroberfläche der Authentic-Ansicht

Die Benutzeroberfläche der Authentic-Ansicht besteht aus einem Hauptfenster, in dem Sie die Dokumentdaten eingeben und bearbeiten, und drei Eingabehilfen. Die Bearbeitung eines Dokuments geht ganz einfach. Wenn Sie die Markup-Tags des Dokuments sehen möchten, blenden Sie die Markup-Tags ein. Geben Sie anschließend den Inhalt Ihres Dokuments ein. Um die Dokumentstruktur zu ändern, verwenden Sie entweder das Kontextmenü oder die Eingabehilfe "Elemente".

© 2016-2022 Altova GmbH

Anzeige der XML-Node Tags (Dokument-Markup-Tags)

Ein XML-Dokument ist im Grunde eine Hierarchie von Nodes. Z.B.:

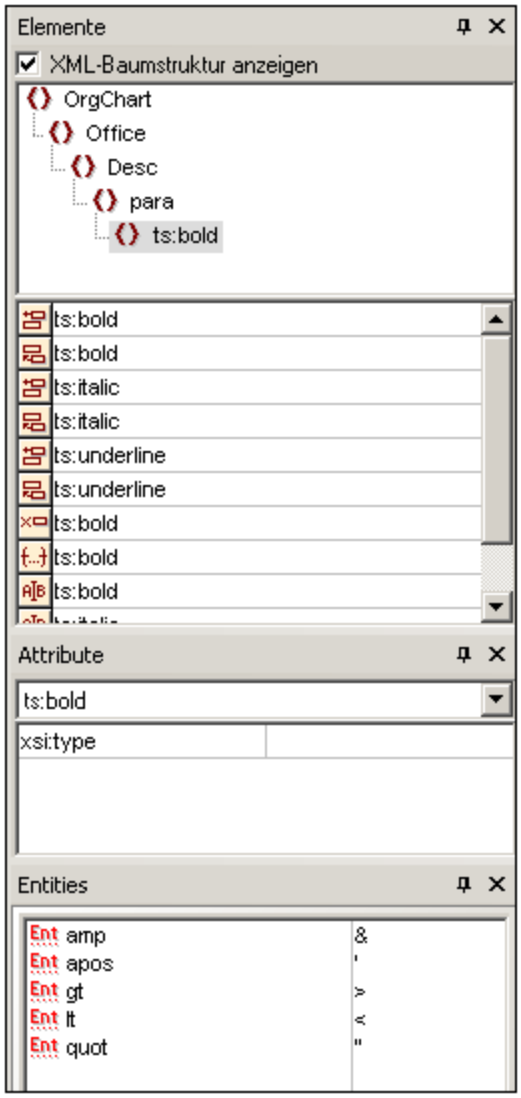
```
<DocumentRoot>
    <Person id="ABC001">
        <Name>Alpha Beta</Name>
        <Address>Some Address</Address>
        <Tel>1234567</Tel>
    </Person>
</DocumentRoot>
```

Standardmäßig werden die Node-Tags in der Authentic-Ansicht nicht angezeigt. Sie können sie durch Auswahl des Menübefehls **Authentic | Große Markup-Symbole einblenden** (oder der Symbolleisten-Schaltfläche ) einblenden. Große Markup-Tags enthalten den Namen des jeweiligen Node. Alternativ dazu können Sie kleine Markup-Symbole (keine Node-Namen in den Tags) und gemischte Markup-Tags (eine Mischung aus großen, kleinen und keinen Markup-Tags, welche durch den Designer des Stylesheets definiert ist; die Standardeinstellung für gemischte Markup-Symbole ist kein Markup) auswählen.

Sie können den Text des XML-Dokuments in der Text-Ansicht Ihres Altova-Produkts oder in einem Text Editor ansehen.

Eingabehilfen

Die Benutzeroberfläche bietet drei Eingabehilfen (*Abbildung unten*), die standardmäßig am rechten Rand des Applikationsfensters angeordnet sind. Es sind dies die Eingabehilfen "Element", "Attribute" und "Entities".



Element-Eingabehilfe

In der Element-Eingabehilfe werden Elemente angezeigt, die an der aktuellen Cursorposition oder der im Hauptfenster ausgewählten Stelle eingefügt und entfernt werden können. Beachten Sie, dass die Eingabehilfe kontextsensitiv ist; ihr Inhalt ändert sich, je nachdem, wo sich der Cursor gerade befindet, bzw. welcher Text ausgewählt ist. Der Inhalt der Eingabehilfe kann auch auf eine andere Art geändert werden: Wenn in der XML-Struktur ein anderer Node ausgewählt wird, werden in der Eingabehilfe die für diesen Node relevanten Elemente angezeigt. Sie können die Element-Eingabehilfe erweitern, um die XML-Struktur anzuzeigen, indem Sie das Kontrollkästchen "XML-Baumstruktur anzeigen" am oberen Rand der Eingabehilfe aktivieren (*siehe Abbildung oben*). In der XML-Baumstruktur wird die Hierarchie der Nodes vom obersten Element-Node bis hinunter zu dem im Hauptfenster ausgewählten Node angezeigt.

Attribute-Eingabehilfe

In der Attribute-Eingabehilfe werden die Attribute des im Hauptfenster ausgewählten Elements sowie deren Werte angezeigt. Attributwerte können in der Attribute-Eingabehilfe eingegeben oder bearbeitet werden. In der Auswahlliste der Attribute-Eingabehilfe können Element-Nodes von der obersten Elementebene bis hinunter zum ausgewählten Element ausgewählt werden. Bei Auswahl eines Elements aus dieser Dropdown-Liste

© 2016-2022 Altova GmbH

werden die Attribute dieses Elements in der Eingabehilfe angezeigt, wo sie anschließend bearbeitet werden können.

Entities-Eingabehilfe
Die Entities-Eingabehilfe ist nicht kontextsensitiv. Hier werden alle im Dokument deklarierten Entities angezeigt. Wenn Sie auf eine Entity doppelklicken, wird sie an der Cursorposition eingefügt. Eine Beschreibung, wie man Entities für ein Dokument hinzufügt, finden Sie im Abschnitt Benutzeroberfläche der Authentic-Ansicht 622.

Kontextmenü

Wenn Sie mit der rechten Maustaste auf eine Stelle in der Authentic-Ansicht doppelklicken, erscheint ein Kontextmenü zum entsprechenden Node. Im Kontextmenü finden Sie Befehle, die Ihnen Folgendes ermöglichen:

- Einfügen von Nodes an dieser Stelle oder vor oder nach dem ausgewählten Node. Das Untermenü enthält eine Liste der Nodes, die an der jeweiligen Stelle eingefügt werden können.
- Entfernen des ausgewählten Node (falls dies im Schema zulässig ist) oder aller entfernbarer übergeordneter Elemente. Die Nodes, die (laut Schema) entfernt werden dürfen, werden im Untermenü aufgelistet.
- Einfügen von Entities und CDATA-Abschnitten. Die für das Dokument deklarierten Entities werden in einem Untermenü aufgelistet. CDATA-Abschnitte können nur innerhalb von Text eingefügt werden.
- Ausschneiden, Kopieren, Einfügen (einschließlich Einfügen als XML oder Text) und Löschen von Dokumentinhalt.

Anmerkung: Nähere Informationen über die Benutzeroberfläche finden Sie unter Benutzeroberfläche der Authentic-Ansicht 622.

11.1.3 Node-Operationen

Es gibt in der Authentic-Ansicht von XML-Dokumenten zwei grundlegende Node-Typen: **Element-Nodes** und **Attribut-Nodes**. Diese Nodes sind alle mit Tags versehen, die Sie einblenden können 610. Es gibt auch andere Nodes im Dokument, wie z.B. Text-Nodes (die mit keinen Markup-Tags versehen sind) und CDATA-Abschnitt-Nodes (welche mit Markup versehen sind, um sie vom umgebenden Text abzugrenzen).

Die in diesem Abschnitt beschriebenen Node-Operationen beziehen sich nur auf Element-Nodes und Attribut-Nodes. Beim Ausprobieren der hier beschriebenen Operationen sollten Sie am besten die großen Markup-Symbole einblenden 610

Anmerkung:
Beachten Sie bitte, dass Sie **nur Elemente derselben oder einer übergeordneten Ebene** vor oder nach dem aktuellen Element einfügen oder anhängen können. Elemente derselben Ebene sind **gleichrangige** Elemente. Bei **gleichrangigen** Elementen eines Absatzelements handelt es sich entweder um andere Absatzelemente oder um Listen, eine Tabelle, eine Grafik usw. handeln. Gleichrangige Elemente können vor oder nach dem Element vorkommen. Elemente übergeordneter Ebenen sind **Vorfahren**-Elemente und Geschwisterelemente von Vorfahren. Bei Vorfahren eines Absatzelements könnte es sich um einen Abschnitt, ein Kapitel, einen Artikel usw. handeln. Ein Absatz in einem gültigen XML-Dokument hat bereits Vorfahren. Wenn Sie daher in der Authentic-Ansicht ein Element einer übergeordneten Ebene erstellen, wird dieses als Geschwisterelement des jeweiligen Vorfahrenelements erstellt. Wenn z.B. ein Abschnittselement an einen

Absatz angehängt wird, wird es als Geschwisterelement des Abschnitts erstellt, der das aktuelle Absatzelement enthält.

Durchführen von Node-Operationen

Node-Operationen können entweder durch Auswahl eines Befehls im Kontextmenü[613] oder durch Klicken auf den Eintrag für die Node-Operation in der Element-Eingabehilfe[611] durchgeführt werden. In einige Fällen kann ein Element oder Attribut hinzugefügt werden, indem Sie in der Authentic-Ansicht des Dokuments auf den Link Node hinzufügen[614] klicken. In speziellen Fällen, in denen Elemente als Absätze oder Listeneinträge definiert wurden, können Sie durch Drücken der Eingabetaste[615] ein neues gleichrangiges Element derselben Art erstellen, wenn sich der Cursor in einem derartigen Element befindet. In diesem Abschnitt erfahren Sie außerdem, wie Nodes über die Funktionen Element anwenden[615], Node entfernen[615] und Element löschen[615] erstellt und gelöscht werden können.

Einfügen von Elementen

Elemente können an den folgenden Stellen eingefügt werden:

- an der Cursorposition innerhalb eines Element-Node. Die an dieser Stelle verfügbaren Elemente werden im Untermenü des Kontextmenüs **Einfügen** aufgelistet. In der Element-Eingabehilfe werden Elemente, die an dieser Stelle eingefügt werden können, durch ein Symbol markiert. Platzieren Sie im Dokument `NanonullOrg.xml` den Cursor in das Element `para` und erstellen Sie über das Kontextmenü und die Element-Eingabehilfe die Elemente `bold` und `italic`.
- vor oder nach dem ausgewählten Element oder einem seiner Vorfahren, wenn dies laut Schema gestattet ist. Wählen Sie das erforderliche Element aus dem bzw. den angezeigten Untermenü(s) aus. In der Element-Eingabehilfe werden Elemente, die vor oder nach dem ausgewählten Element eingefügt werden können, durch die Symbole bzw. gekennzeichnet. Beachten Sie, dass Sie in der Element-Eingabehilfe Elemente nur vor/nach dem ausgewählten Element einfügen können, nicht aber vor/nach dem Vorfahrenelement. Probieren Sie diesen Befehl aus, indem Sie den Cursor zuerst in das Element `para` und anschließend in die Tabelle mit der Liste der Angestellten platzieren.

Link "Node hinzufügen"

Wenn ein Element oder Attribut im Dokument-Design nicht aber im XML-Dokument selbst vorhanden ist, wird an der Stelle im Dokument, an der dieser Node definiert ist, ein `Node hinzufügen` Link angezeigt. Um diesen Link in der Zeile mit dem Text *Location of logo* zu sehen, wählen Sie den Node `@href` im Element `CompanyLogo` aus und löschen Sie ihn (durch Drücken der **Entf**-Taste). Der Link `@href hinzufügen` wird im Element `CompanyLogo`, das Sie bearbeitet haben, angezeigt (*Abbildung unten*). Wenn Sie auf den Link klicken, wird der Node `@href` zum XML-Dokument hinzugefügt. Das Textfeld innerhalb des Tags `@href` wird angezeigt, weil im Design festgelegt ist, dass der Node `@href` auf diese Art hinzuzufügen ist. Sie müssen nun noch den Wert (oder Inhalt) des Node `@href` eingeben. Geben Sie den Text `nanonull.gif` ein.



 © 2016-2022 Altova GmbH

Wenn das Content Model eines Elements nicht eindeutig ist, wenn z.B. festgelegt ist, dass eine Sequenz von Child-Elementen in jeder beliebigen Reihenfolge angezeigt werden kann, so wird der Link `...hinzufügen` angezeigt. Beachten Sie, dass kein Node-Name definiert ist. Wenn Sie auf den Link klicken, erscheint eine Liste von Elementen, die an dieser Stelle eingefügt werden dürfen.

Anmerkung: Der Link "Node hinzufügen" erscheint direkt in der Dokumentvorlage; es gibt keinen entsprechenden Eintrag dazu im Kontextmenü oder in der Element-Eingabehilfe.

Erstellen neuer Elemente mit der Eingabetaste

In Fällen, in denen ein Element (vom Stylesheet Designer) als Absatz oder Listeneintrag formatiert werden muss, wird bei Drücken der Eingabetaste ein neuer Node dieser Art nach dem aktuellen Node eingefügt, wenn sich der Cursor innerhalb eines solchen Node befindet. Sie können diese Funktion im Dokument `NanonullOrg.xml` ausprobieren. Gehen Sie zum Ende des Node `para` (genau vor den End-Tag) und drücken Sie die **Eingabetaste.**

Anwenden von Elementen

In Elementen mit gemischtem Inhalt (denjenigen, die sowohl Text als auch Child-Elemente enthalten) können Sie Textinhalt markieren und darauf eines der zulässigen Child-Elemente anwenden. Der ausgewählte Text wird zum Inhalt des angewendeten Elements. Um Elemente anzuwenden, wählen Sie im Kontextmenü den Befehl "Anwenden" und wählen Sie anschließend eines der vorgeschlagenen Elemente aus. (Wenn auf den ausgewählten Text keine Elemente angewendet werden können, wird der Befehl "Anwenden" im Kontextmenü nicht angezeigt.) In der Element-Eingabehilfe werden Elemente, die für eine Auswahl angewendet werden können, durch das Symbol  gekennzeichnet. Wählen Sie im Dokument `NanonullOrg.xml` Text im Element gemischten Inhalts `para` aus und probieren Sie aus, wie sich die Elemente `bold` und `italic` bei Anwendung auswirken.

Der Stylesheet Designer hätte auch eine Symbolleisten-Schaltfläche für die Anwendung eines Elements erstellen können. Im Dokument `NanonullOrg.xml` können die Elemente `bold` und `italic` durch Klicken auf die Symbole "bold" bzw. "italic" in der Authentic-Symbolleiste der Applikation angewendet werden.

Entfernen von Nodes

Ein Node kann entfernt, werden, wenn das Dokument dadurch nicht ungültig wird. Bei Entfernung eines Node werden der Node und sein gesamter Inhalt gelöscht. Sie können einen Node mit dem Kontextmenübefehl **entfernen** entfernen. Wenn der Befehl "Entfernen" markiert wird, erscheint ein Untermenü, das alle Nodes enthält, die entfernt werden dürfen, beginnend beim ausgewählten Node bis zum obersten Node des Dokuments. Um einen zu löschenden Node auszuwählen, platzieren Sie den Cursor in den Node oder markieren Sie den Node (oder einen Teil davon). In der Element-Eingabehilfe werden Nodes, die entfernt werden dürfen, durch das Symbol  gekennzeichnet. Sie können einen solchen Node auch entfernen, indem Sie ihn auswählen und die **Entf-**Taste drücken. Probieren Sie im Dokument `NanonullOrg.xml` die oben beschriebenen Methoden aus, indem Sie einige Nodes entfernen. Sie können Ihre Änderungen mit **Strg+Z** rückgängig machen.

Löschen von Elementen

Element-Nodes, bei denen es sich um Child-Elemente mit gemischtem Inhalt (sowohl Text- als auch Element-Children) handelt, können gelöscht werden. Sie können das gesamte Element löschen, wenn Sie den Node auswählen oder den Cursor als Einfügepunkt in den Node setzen. Sie können den Element-Markup-Code eines Textfragments innerhalb des Elements löschen, indem Sie das Textfragment markieren. Klicken Sie nach dem

Markieren des Fragments im Kontextmenü auf den Befehl **Löschen** und wählen Sie anschließend das zu löschende Element aus. In der Element-Eingabehilfe werden Elemente, die aus einer bestimmten Auswahl gelöscht werden können, durch das Symbol (Einfügepunktauswahl) und (Bereichsauswahl) gekennzeichnet. Probieren Sie das Löschen von Elementen im Dokument `NanonullOrg.xml` an den Child-Elementen `bold` und `italic` des Elements `para` (welches gemischten Inhalt hat) aus.

Tabellen und Tabellenstruktur

Es gibt zwei Arten von Tabellen in der Authentic-Ansicht:

- *SPS-Tabellen (statische und dynamische)*. Die allgemeine Struktur von SPS-Tabellen wird durch den Stylesheet Designer festgelegt. Innerhalb dieser grundlegenden Struktur sind die einzigen zulässigen Änderungen an der Struktur inhaltsbezogene. So können Sie z. B. neue Zeilen zu einer dynamischen SPS-Tabelle hinzufügen.
- *XML-Tabellen*, in welchen Sie entscheiden, dass der Inhalt eines bestimmten Node (z.B. eines Node für personenspezifische Daten) als Tabelle dargestellt werden soll. Wenn der Stylesheet Designer die Erstellung dieses Node als XML-Tabelle ermöglicht hat, können Sie die Struktur der Tabelle festlegen sowie ihren Inhalt bearbeiten. XML-Tabellen werden im Abschnitt Verwendung von Tabellen in der Authentic-Ansicht 642 näher beschrieben.

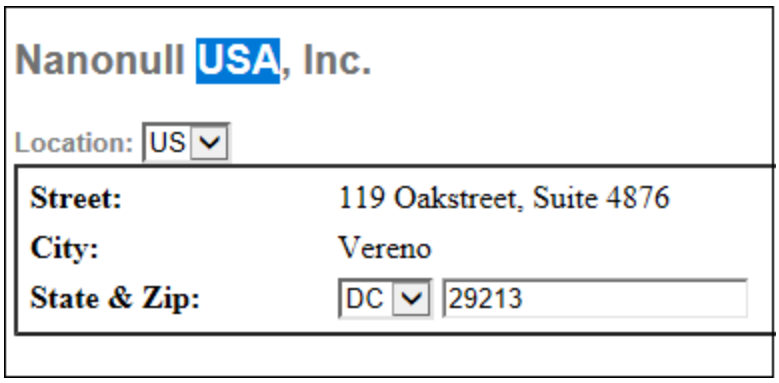
11.1.4 Eingabe von Daten in der Authentic-Ansicht

Daten werden im XML-Dokument direkt im Hauptfenster der Authentic-Ansicht eingegeben. Zusätzlich dazu können Daten für Attribute (also der Attributwert) über die Attribute-Eingabehilfe eingegeben werden 618.Daten werden entweder (i) direkt als Text oder (ii) durch Auswahl einer Option in einem Dateneingabeelement, eingegeben, das anschließend auf einen vordefinierten Texteintrag gemappt wird.

Hinzufügen von Textinhalt

Sie können Elementinhalt und Attributwerte im Hauptfenster der Authentic-Ansicht direkt als Text eingeben. Platzieren Sie dazu den Cursor an die gewünschte Stelle und geben Sie den Text ein. Sie können Text auch aus der Zwischenablage in das Dokument kopieren. Inhalt kann auch mit den normalen Inhaltsbearbeitungsmechanismen wie **Feststelltaste** und **Entfernen** bearbeitet werden. Zum Beispiel können Sie den zu bearbeitenden Text markieren und den Ersetzungstext mit gedrückter **Feststelltaste** eingeben.

Um z.B. den Namen der Firma im Feld `Name` des Elements `Office` zu ändern, platzieren Sie den Cursor hinter Nanonull und geben Sie "USA" ein, um den Namen von "Nanonull, Inc." in "Nanonull USA, Inc." zu ändern.



 © 2016-2022 Altova GmbH

Wenn Text editierbar ist, können Sie den Cursor hineinsetzen und den Text markieren, andernfalls ist das nicht der Fall. Versuchen Sie, einen der **Feldnamen** (nicht den Wert im Feld) "Street", "City", "State/Zip" im Adressblock zu ändern. Sie können den Cursor nicht in diesen Text platzieren, da dies kein XML-Inhalt ist, sondern aus dem StyleVision Power Stylesheet stammt.

Einfügen von Sonderzeichen und Entities

Bei der Eingabe von Daten werden folgende Inhaltsarten speziell behandelt:

- *Sonderzeichen, die als XML-Markup-Symbole verwendet werden* (kaufmännisches Und, Apostroph, Größer als, Kleiner als und Anführungszeichen). Diese Zeichen stehen als vordefinierte Entities[619] zur Verfügung und können durch Doppelklick auf die entsprechende Entity in der Entities-Eingabehilfe im Dokument eingefügt werden. Wenn diese Zeichen häufig vorkommen (z.B. in Programmcodelisten), können Sie innerhalb von CDATA-Abschnitten eingegeben werden. Klicken Sie zum Einfügen eines CDATA-Abschnitts mit der rechten Maustaste an die gewünschte Stelle und wählen Sie im Kontextmenü den Befehl **CDATA-Abschnitt einfügen**. Der XML-Prozessor ignoriert alle Markup-Zeichen innerhalb von CDATA-Abschnitten. Dies bedeutet auch, dass Sonderzeichen innerhalb von CDATA-Abschnitten als das Zeichen selbst und nicht die Entity-Referenz eingegeben werden müssen.
- *Sonderzeichen, die nicht über die Tastatur eingegeben werden können,* sollten durch Kopieren des Zeichens aus der Zeichentabelle Ihres Systems an die gewünschte Stelle im Dokument eingegeben werden.
- *Ein häufig verwendeter Textstring* kann als Entity definiert werden[659], die in der Entities-Eingabehilfe angezeigt wird. Die Entity wird an der gewünschten Stelle eingefügt[619]. Platzieren Sie dazu den Cursor an die gewünschte Stelle und doppelklicken Sie in der Entities-Eingabehilfe auf die Entity. Diese Funktion eignet sich dazu, um Dokumente auf aktuellem Stand zu halten, da der Wert des Textstrings nur einmal gespeichert ist. Wenn der Wert geändert werden muss, müssen Sie nur die Entity-Definition ändern.

Anmerkung: Wenn Markup-Tags in der Authentic -Ansicht ausgeblendet werden, werden leere Elemente leicht übersehen. Um sicherzustellen, dass Sie kein leeres Element übersehen haben aktivieren Sie ein der Optionen große Markup-Symbole (oder kleine Markup-Symbole) anzeigen.[610]

Probieren Sie die Verwendung der einzelnen oben beschriebenen Arten von Text aus.

Hinzufügen von Inhalt über ein Dateneingabe-Steuerelement

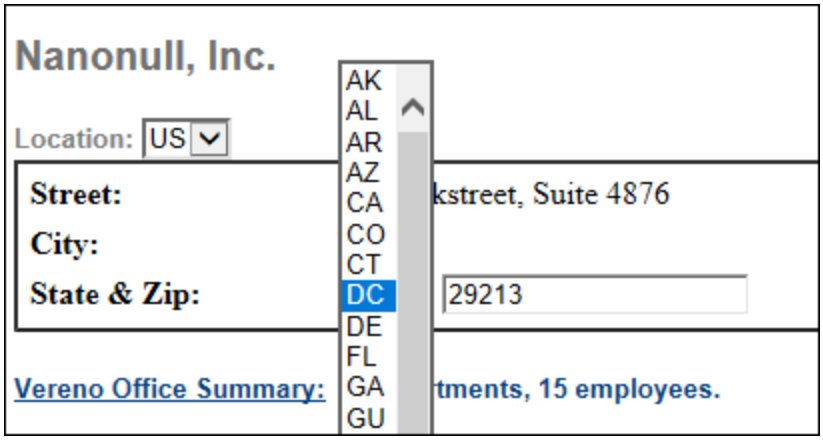
Beim Editieren von Inhalt wird dieser direkt durch Eingabe des Texts als Elementinhalt eingegeben. Es gibt aber noch eine andere Methode, um **Elementinhalt** (oder Attributwerte) in der Authentic-Ansicht einzugeben: mit Hilfe von Dateneingabe-Steuerelemente.

Nachfolgend sehen Sie eine Liste von Dateneingabe-Steuerelementen in der Authentic-Ansicht sowie eine Erklärung dazu, wie Daten bei jedem dieser Elemente in die XML-Datei eingegeben werden.

Dateneingabe-Steuerelement	Daten in der XML-Datei
Eingabefeld (Textfeld)	Text wird vom Benutzer eingegeben
Eingabefeld mit mehreren Zeilen	Text wird vom Benutzer eingegeben

Dropdown-Liste	Auswahl des Benutzers wird auf einen Wert gemappt
Kontrollkästchen	Auswahl des Benutzers wird auf einen Wert gemappt
Optionsfeld	Auswahl des Benutzers wird auf einen Wert gemappt
Schaltfläche	Auswahl des Benutzers wird auf einen Wert gemappt

In der statischen Tabelle mit den Adressfeldern (siehe unten) stehen zwei Dateneingabe-Steuerelemente zur Verfügung: Ein Eingabefeld für das Feld "Zip" (PLZ) und eine Dropdown-Liste für das Feld "State" (Bundesstaat). Die Werte, die Sie in die Textfelder eingeben, werden direkt als XML-Inhalt der entsprechenden Elemente eingegeben. Bei anderen Dateneingabe-Steuerelementen wird Ihre Auswahl auf einen Wert gemappt.



Hier sehen Sie den XML-Text für die oben gezeigte Authentic-Ansicht:

```
<Address>
   <ipo:street>119 Oakstreet, Suite 4876</ipo:street>
   <ipo:city>Vereno</ipo:city>
   <ipo:state>DC</ipo:state>
   <ipo:zip>29213</ipo:zip>
</Address>
```

Beachten Sie, dass die Dropdown-Listen-Auswahl "DC" auf einen Wert "DC" gemappt ist. Der Wert im Feld "zip" wird direkt als Inhalt des Elements `ipo:zip` eingegeben.

11.1.5 Eingabe von Attributwerten

Ein Attribut ist eine Eigenschaft eines Elements. Ein Element kann beliebig viele Attribute haben. Attribute haben Werte. Manchmal müssen XML-Daten als Attributwert eingegeben werden. In der Authentic-Ansicht werden Attributwerte auf zwei Arten eingegeben:

© 2016-2022 Altova GmbH

- Als Inhalt im Hauptfenster, wenn das Attribut so definiert wurde, dass es die Eingabe des Werts auf diese Art zulässt.
- In der Attribut-Eingabehilfe

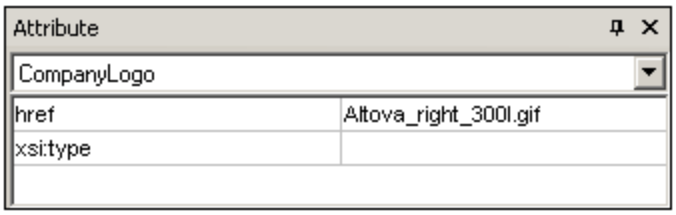
Attributwerte im Hauptfenster

Attributwerte können als normaler Text oder als Text in einem Eingabefeld eingegeben werden oder als vom Benutzer aus einer Auswahl gewählter Eintrag, der auf einen XML-Wert gemappt wird. Attributwerte werden auf dieselbe Art eingegeben, wie Element-Inhalt: siehe Eingabe von Daten in der Authentic-Ansicht 616. In solchen Fällen erfolgt die Unterscheidung zwischen Element-Inhalt und Attributwerten durch das StyleVision Power Stylesheet und die Daten werden entsprechend behandelt.

Attributwerte in der Attribut-Eingabehilfe

Wenn Sie einen Attributwert eingeben oder ändern wollen, können Sie dies in der Attribut-Eingabehilfe tun. Zuerst wird der Attribut-Node in der Authentic-Ansicht ausgewählt, dann wird der Wert des Attributs in der Attribute-Eingabehilfe eingegeben oder bearbeitet. Der Pfad des in `NanonullOrg.xml` verwendeten Logos wird als der Wert des `href` Attributs des Elements `CompanyLogo` gespeichert. Um ein anderes Logo zu verwenden, gehen Sie vor wie folgt:

1. Wählen Sie das Element `CompanyLogo` aus, indem Sie auf den Tag CompanyLogo klicken. Die Attribute des Elements `CompanyLogo` werden in der Attribut-Eingabehilfe angezeigt.
2. Ändern Sie in der Attribut-Eingabehilfe den Wert des Attributs `href` von `nanonull.gif` in `Altova_right_300.gif` (ein Bild im Ordner `EXFOLDER%>`).

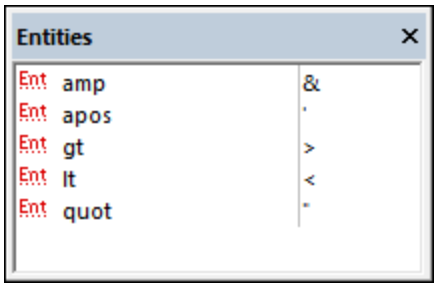


Daraufhin wird das Nanonull-Logo durch das Altova-Logo ersetzt.

Anmerkung: Entities können nicht in der Attribut-Eingabehilfe eingegeben werden.

11.1.6 Hinzufügen von Entities

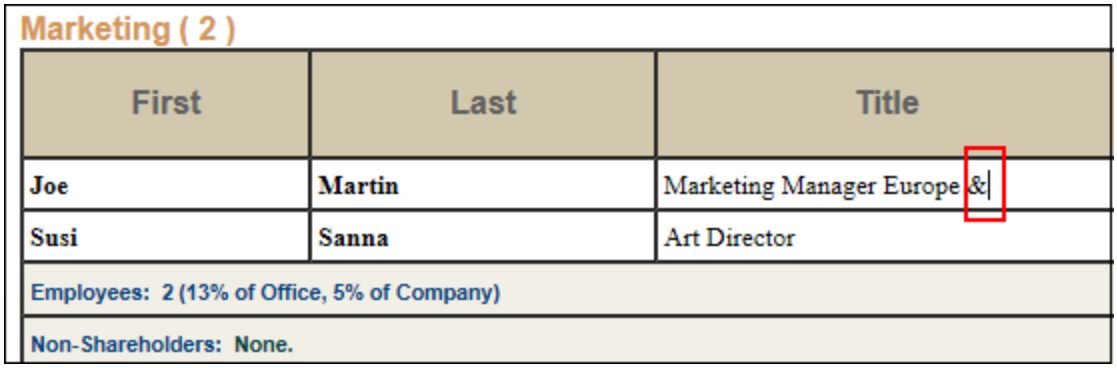
Bei einer Entity in der Authentic-Ansicht handelt es sich normalerweise (nicht notwendigerweise) um XML-Daten wie z.B. ein einzelnes Zeichen, einen Textstring oder auch ein Fragment eines XML-Dokuments. Auch eine Binärdatei wie z.B. eine Bilddatei kann eine Entity sein. Alle verfügbaren Entities werden in der Entities-Eingabehilfe (*Abbildung unten*) angezeigt. Um eine Entity einzufügen, platzieren Sie den Cursor an die gewünschte Stelle im Dokument und doppelklicken Sie anschließend in der Entities-Eingabehilfe auf die Entity. Beachten Sie, dass Sie Entities nicht in der Attribut-Eingabehilfe eingeben können.



Das kaufmännische Und (&) hat in XML eine spezielle Bedeutung (wie auch das Apostroph, das "Größer als"- und das "Kleiner als"-Zeichen und das doppelte Anführungszeichen). Um diese Zeichen einzufügen, werden Entities verwendet, damit sie nicht mit Zeichen, die eine XML-Bedeutung haben, verwechselt werden. Diese Zeichen sind in der Authentic-Ansicht als Entities verfügbar.

Ändern Sie in `NanonullOrg.xml` den Jobtitel von Joe Martin (in Marketing) in Marketing Manager Europe & Asia. Gehen Sie dazu folgendermaßen vor:

1. Platzieren Sie den Cursor dorthin, wo das kaufmännische Und eingefügt werden soll.
2. Doppelklicken Sie auf die Entity, die als "amp" aufgelistet ist. Daraufhin wird ein kaufmännisches Und eingefügt (*Abbildung unten*)



Anmerkung:
Die Entities-Eingabehilfe ist nicht kontextsensitiv. Es werden immer alle verfügbaren Entities angezeigt, egal wo sich der Cursor gerade befindet. Dies bedeutet nicht, dass eine Entity überall im Dokument eingefügt werden kann. Wenn Sie sich nicht sicher sind, validieren Sie das Dokument nach dem Einfügen der Entity mit: **XML | XML Validieren** (**F8**).

Definieren Ihrer eigenen Entities

Als Dokument-Bearbeiter können Sie auch Ihre eigenen Entities definieren. Eine Beschreibung dazu finden Sie im Abschnitt Definieren von Entities in der Authentic-Ansicht 659.

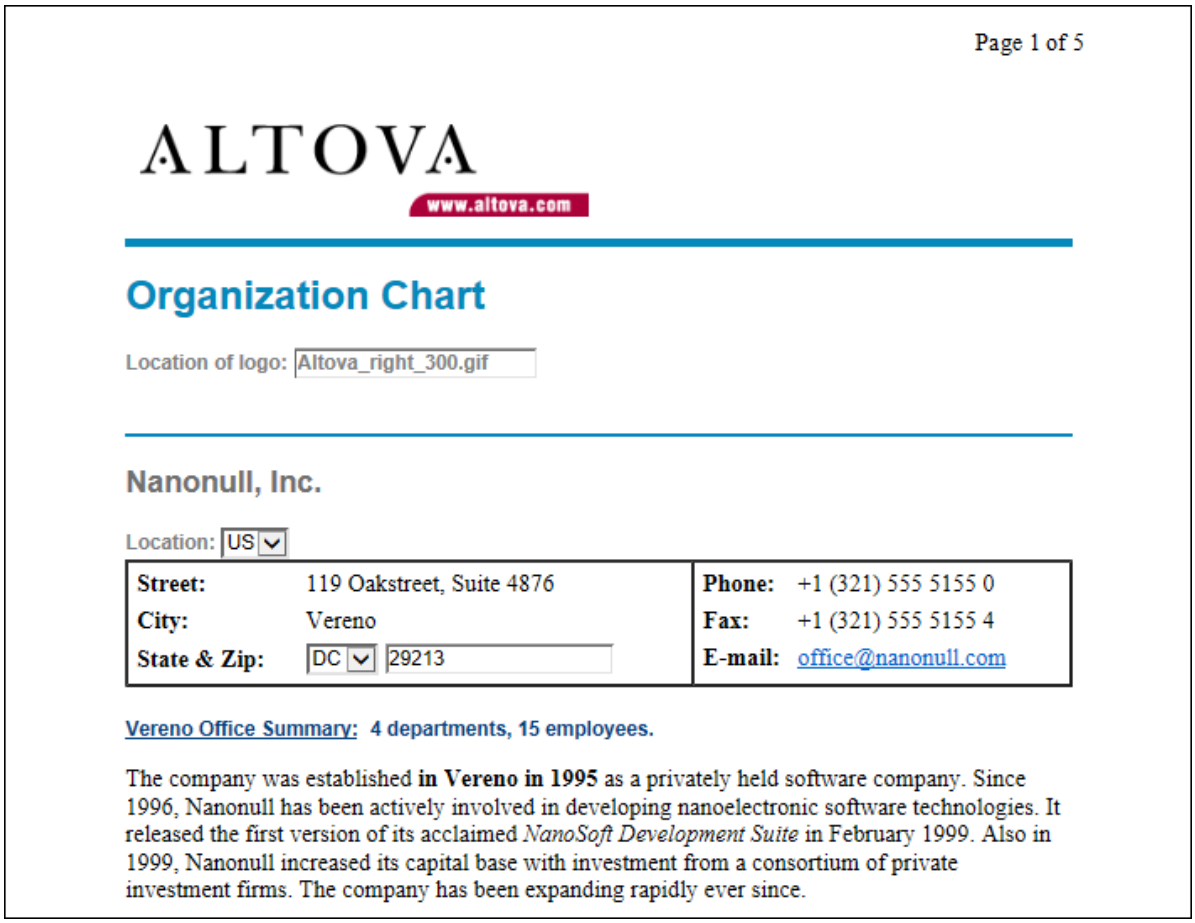
11.1.7 Drucken des Dokuments

Bei Ausdruck eines XML-Dokuments von der Authentic-Ansicht aus bleibt die in der Authentic-Ansicht sichtbare Formatierung erhalten.

© 2016-2022 Altova GmbH

So drucken Sie die Datei `NanonullOrg.xml`:

1. Wechseln Sie in den Modus "Markup-Symbole ausblenden", wenn Sie gerade einen anderen Modus verwenden, da sonst auch die Markup-Symbole gedruckt werden.
2. Wählen Sie **Datei | Druckvorschau**, um eine Vorschau aller Seiten anzuzeigen. In der Abbildung unten sehen Sie einen Ausschnitt einer um 50 % verkleinerten Druckvorschauseite. Beachten Sie, dass die Formatierung auf der Seite dieselbe ist, wie in der Authentic-Ansicht.



3. Um die Datei zu drucken, wählen Sie **Datei | Drucken**.

Beachten Sie, dass Sie auch eine Version des Dokuments drucken können, in der Markup-Symbole angezeigt werden. Wechseln Sie dazu in den Modus "Kleine Markup-Symbole einblenden" oder "Große Markup-Symbole einblenden" und wählen Sie den Befehl "Drucken".

11.2 Benutzeroberfläche der Authentic-Ansicht

Die Authentic-Ansicht steht zur Verfügung, wenn Sie auf das Register "Authentic" des aktiven Dokuments klicken. Wenn dem XML-Dokument kein SPS zugewiesen wurde, werden Sie aufgefordert, dies zu tun. Mit dem Befehl **Authentic | StyleVision Stylesheet zuweisen** können Sie jederzeit ein SPS zuweisen.

Dieser Abschnitt enthält:

- eine Übersicht über die Benutzeroberfläche
- eine Beschreibung der Symbolleistensymbole der Authentic-Ansicht
- eine Beschreibung der Ansichtsmodi im FensterAuthentic-Ansicht-
- eine Beschreibung der Eingabehilfen und deren Verwendung
- eine Beschreibung der verschiedenen in der Authentic-Ansicht verfügbaren Kontextmenüs

Weitere Informationen zur Authentic-Ansicht finden Sie:

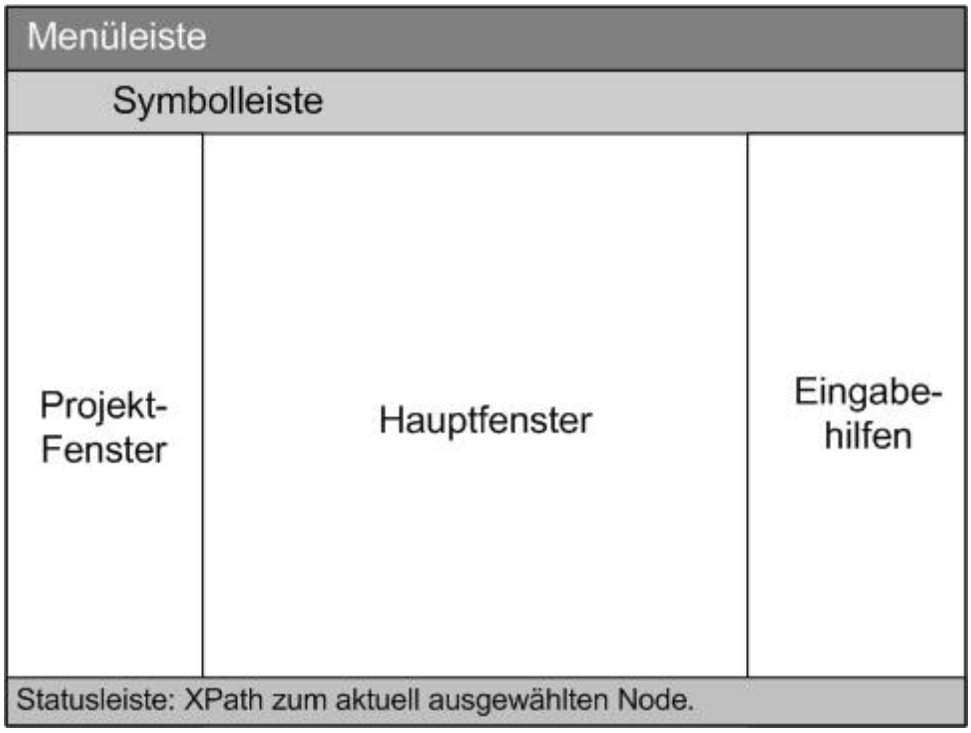
- im Tutorial zur Authentic-Ansicht, in dem beschrieben wird, wie Sie mit der Authentic-Ansicht arbeiten. Sie finden das Tutorial in der Dokumentation zu Altova XMLSpy und zur Altova Authentic Desktop (siehe Abschnitt Tutorials) sowie online.
- Eine ausführliche Beschreibung der Menübefehle der Authentic-Ansicht finden Sie im Abschnitt "Benutzerreferenz" Ihrer Produktdokumentation.

Altova Website: XML-Content-Bearbeitung, Erstellung von XML-Dokumenten

11.2.1 Übersicht über die Benutzeroberfläche

Die Authentic-Ansicht besteht aus einer Menüleiste am oberen Rand des Fensters und drei Bereichen: dem Projekt-Fenster, dem Hauptfenster und den Eingabehilfen-Fenstern (siehe Abbildung unten).

© 2016-2022 Altova GmbH



Menüleiste

Die Menüs der Menüleiste werden in der Produktdokumentation im Abschnitt "Benutzerreferenz" näher beschrieben.

Symbolleiste

Eine Beschreibung der Schaltflächen der Symbolleiste finden Sie im Abschnitt Symbolleistenschaltflächen der Authentic-Ansicht 624.

Projekt-Fenster

XML-, XSL-, XML-Schema und Entity-Dateien können in einem Projekt zusammen gruppiert werden. Verwenden Sie zum Erstellen und Bearbeiten der Liste der Projektdateien die Befehle im Menü "Projekt" (Beschreibung siehe Abschnitt "Benutzerreferenz" der Produktdokumentation). Die Liste der Projektdateien wird im Projekt-Fenster angezeigt. Um eine Datei im Projekt-Fenster aufzurufen, doppelklicken Sie darauf.

Info-Fenster

Dieses Fenster enthält Informationen über den gerade in der Authentic-Ansicht ausgewählten Node.

Hauptfenster

Das Hauptfenster ist der Bereich, in dem das XML-Dokument angezeigt und bearbeitet wird. Eine Beschreibung dazu finden Sie im Abschnitt Authentic-Ansicht Hauptfenster 627.

Eingabehilfen

Dieser Bereich enthält drei Eingabehilfen-Fenster: Elemente, Attribute und Entities. Welche Einträge in diesen Fenstern (Element- und Attribute-Eingabehilfen) zu sehen sind, ist vom Kontext abhängig, d.h. davon, an welcher Stelle im Dokument sich der Cursor gerade befindet. Um ein Element oder eine Entity in das Dokument einzugeben, doppelklicken Sie in der Eingabehilfe darauf. Der Wert eines Attributs wird in der Attribute-Eingabehilfe im Feld "Wert" des entsprechenden Attributs eingegeben. Nähere Informationen dazu finden Sie im Abschnitt Authentic-Ansicht Eingabehilfen 630.

Statusleiste

In der Statusleiste wird der XPath zum aktuell ausgewählten Node angezeigt.

Kontextmenüs

Dies sind Menüs, die angezeigt werden, wenn Sie im Hauptfenster auf die rechte Maustaste klicken. Es stehen Ihnen kontextsensitive Editierbefehle zur Verfügung, mit Hilfe derer Sie die Struktur und den Inhalt des ausgewählten Nodes bearbeiten können. Diese Befehle gestatten das Einfügen, Anhängen oder Löschen eines Nodes, das Hinzufügen von Entities und das Ausschneiden und Einfügen von Inhalt.

11.2.2 Authentic-Ansicht Symbolleistenschaltflächen

Die Schaltflächen der Symbolleiste der Authentic-Ansicht sind Befehls-Shortcuts. Einige Symbole kennen Sie sicher bereits von anderen Windows-Anwendungen oder anderen Altova-Produkten her, andere sind vielleicht neu für Sie. In diesem Abschnitt werden die für die Authentic-Ansicht spezifischen Befehle beschrieben. In der nachfolgenden Beschreibung sind verwandte Befehle gemeinsam gruppiert.

Ein- und Ausblenden von XML Markup

In der Authentic-Ansicht können die Tags für alle, einige oder keines der XML-Elemente oder Attribute eingeblendet werden - entweder mit Namen (große Markup-Symbole) oder ohne Namen (kleine Markup-Symbole). Die vier Markup-Symbole werden in der Symbolleiste angezeigt, die entsprechenden Befehle finden Sie im Menü **Authentic.**



	Markup-Symbole ausblenden. Alle XML-Tags mit Ausnahme der reduzierten Tags werden ausgeblendet. Wenn Sie im Modus "Markup-Symbole ausblenden" auf einen Tag, der reduziert wurde, doppelklicken (wodurch er normalerweise erweitert wird), wird der Inhalt des Nodes angezeigt und die Tags ausgeblendet.
	Kleine Markup-Symbole einblenden. Die XML-Element/Attribut-Tags werden ohne Namen angezeigt.
	Große Markup-Symbole einblenden. Die XML-Elemente/Attribute werden mit ihren Namen angezeigt.

© 2016-2022 Altova GmbH

	Gemischte Markup-Symbole einblenden. Im StyleVision Power Stylesheet können Sie für jedes XML-Element oder Attribut einzeln festlegen, wie es im angezeigt werden soll (mit großen oder kleinen Markup-Symbolen oder gar nicht). Der Modus heißt "Gemischte Markup-Symbole", weil darin für einige Elemente mit Markup-Symbolen angezeigt werden, andere dagegen ohne. Im gemischten Markup-Modus der Authentic-Ansicht ist die Anzeige der Markup-Symbole also benutzerdefiniert. Beachten Sie allerdings, dass diese Einstellungen von der Person vorgenommen werden, die das StyleVision Power Stylesheet erstellt hat. Sie kann nicht vom Benutzer der Authentic-Ansicht vorgenommen werden.

Bearbeiten von dynamischen Tabellenstrukturen

Zeilen in einer **dynamischen SPS-Tabelle** sind Wiederholungen einer Datenstruktur. Jede Zeile steht für eine Instanz eines einzelnen Elements, daher hat jede Zeile dieselbe XML-Substruktur wie die darauf folgende.

Die Befehle zum Bearbeiten dynamischer Tabellen dienen zum Bearbeiten der Zeilen einer dynamischen SPS-Tabelle, d.h. Sie können die Anzahl und Reihenfolge der Element-Instanzen ändern. Sie können jedoch nicht die Spalten einer dynamischen SPS-Tabelle bearbeiten, da dadurch die Substruktur der einzelnen Elementinstanzen geändert würde.

Die Symbole zum Bearbeiten dynamischer Tabellen werden in der Symbolleiste angezeigt und können auch über das Menü **Authentic** aufgerufen werden.



	Zeile anhängen
	Zeile in Tabelle einfügen
	Aktuelle Tabellenzeile kopieren (d.h. der Inhalt einer Zelle wird dupliziert)
	Aktuellen Zeile um eine Zeile nach oben verschieben
	Aktuellen Zeile um eine Zeile nach unten verschieben
	Aktuelle Zeile löschen

Anmerkung: Diese Befehle sollten nur bei **dynamischen SPS-Tabellen** und nicht innerhalb von statischen SPS-Tabellen verwendet werden. Die verschiedenen Arten der in der Authentic-Ansicht verwendeten Tabellen werden weiter hinten im Abschnitt Tabellen in der Authentic-Ansicht 642 näher beschrieben.

Erstellen und Bearbeiten von XML-Tabellen

Sie können Ihre eigenen Tabellen einfügen, wenn Sie Daten in Tabellenform darstellen möchten. Solche Tabellen werden als XML-Tabellen eingefügt. Sie können die Struktur einer XML-Tabelle bearbeiten und die Tabelle formatieren. Die Symbole dazu stehen in der Symbolleiste zur Verfügung (siehe Abbildung unten). Im Abschnitt XML-Tabellenbearbeitungsbefehle 648 finden Sie nähere Informationen zu diesen Befehlen.



Die Befehle zu diesen Symbolen **stehen nicht als Menübefehle** zur Verfügung. Beachten Sie außerdem, dass die entsprechende Funktion erst im StyleVision Power Stylesheet aktiviert und konfiguriert werden muss, damit Sie XML-Tabellen verwenden können. Eine ausführliche Beschreibung der in der Authentic-Ansicht verwendeten Tabellen bzw. wie diese erstellt und bearbeitet werden, finden Sie weiter hinten in der Dokumentation im Abschnitt Tabellen in der Authentic-Ansicht 642.

Symbole zur Textformatierung

Text wird in der Authentic-Ansicht formatiert, indem man darauf ein XML-Element oder -Attribut anwendet, das die gewünschte Formatierung aufweist. Wurde eine solche Formatierung definiert, kann der Autor des StyleVision Power Stylesheet in der Symbolleiste der Authentic-Ansicht Schaltflächen zur Verfügung stellen, mit denen die Formatierung angewendet wird. Um eine Textformatierung mit Hilfe eines Textformatierungssymbols anzuwenden, markieren Sie den gewünschten Text und klicken Sie auf das entsprechende Symbol.

Symbole für die Navigation in DB-Zeilen



Die Pfeilsymbole von links nach rechts sind: Gehe zum ersten Datensatz; Gehe zum vorhergehenden Datensatz; Dialogfeld *"Gehe zu Datensatz #*; Gehe zum nächsten Datensatz; und Gehe zum letzten Datensatz.

	Mit diesem Symbol wird das Dialogfeld "Datenbankabfrage bearbeiten" aufgerufen. Hier können Sie eine Abfrage eingeben. In der Authentic-Ansicht werden die abgefragten Datensätze angezeigt.

XML-Datenbankbearbeitung

Mit dem Befehl **Neue Zeile mit XML-Daten für Bearbeitung auswählen** können Sie in einer XML-Datenbank wie z.B. IBM DB2 eine neue Zeile in der entsprechenden Tabelle auswählen. Diese Zeile wird in der Authentic-Ansicht angezeigt, kann dort bearbeitet werden und wird dann wieder in der Datenbank gespeichert.

Portable XML Form (PXF)-Symbolleisten-Schaltflächen

In der Authentic-Ansicht von XMLSpy und Authentic Desktop stehen die folgenden PXF-Schaltflächen zur Verfügung:



Durch Klicken auf die jeweilige Schaltfläche wird ein HTML-, RTF-, PDF- und/oder DocX-Dokument generiert.

© 2016-2022 Altova GmbH

Diese Schaltflächen sind aktiv, wenn eine PXF-Datei in der Authentic-Ansicht geöffnet ist. Die einzelnen Schaltflächen sind aktiv, wenn die PXF-Datei so konfiguriert wurde, dass sie das XSLT-Stylesheet für das jeweilige Ausgabeformat enthält. Wenn die PXF-Datei z.B. so konfiguriert wurde, dass sie XSLT Stylesheets für HTML und RTF enthält, so sind nur die Symbolleisten-Schaltflächen für die HTML- und RTF-Ausabe aktiv, während diejenigen für die PDF- und DocX (Word 2007+)-Ausgabe deaktiviert sind.

11.2.3 Authentic-Ansicht Hauptfenster

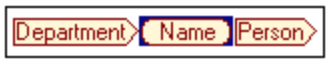
In der Authentic-Ansicht gibt es vier Ansichtsmodi: Große Markup-Symbole; Kleine Markup-Symbole; Gemischte Markup-Symbole; und Alle Markup-Symbole ausblenden. Damit können Sie die Markup-Informationen für Dokumente mit unterschiedlicher Genauigkeit anzeigen. Um zwischen diesen Ansichtsmodi zu wechseln, verwenden Sie die Befehle im Menü "Authentic" oder die Symbole der Symbolleiste (siehe vorhergehender Abschnitt, Authentic-Ansicht Symbolleistenschaltflächen 624).

Große Markup-Symbole

In diesem Modus werden die Start- und End-Tags von Elementen und Attributen mit dem Namen des Elements/Attributs im Tag angezeigt:



Das Element `Name` in der Abbildung oben ist **erweitert**, d.h. sein Start- und End-Tag sowie der Inhalt der Elements werden angezeigt. Durch Doppelklicken auf den Start- oder End-Tag eines Elements/Attributs können Sie dieses **reduzieren.** Um ein reduziert angezeigtes Element/Attribut wieder zu erweitern, doppelklicken Sie auf den reduzierten Tag.

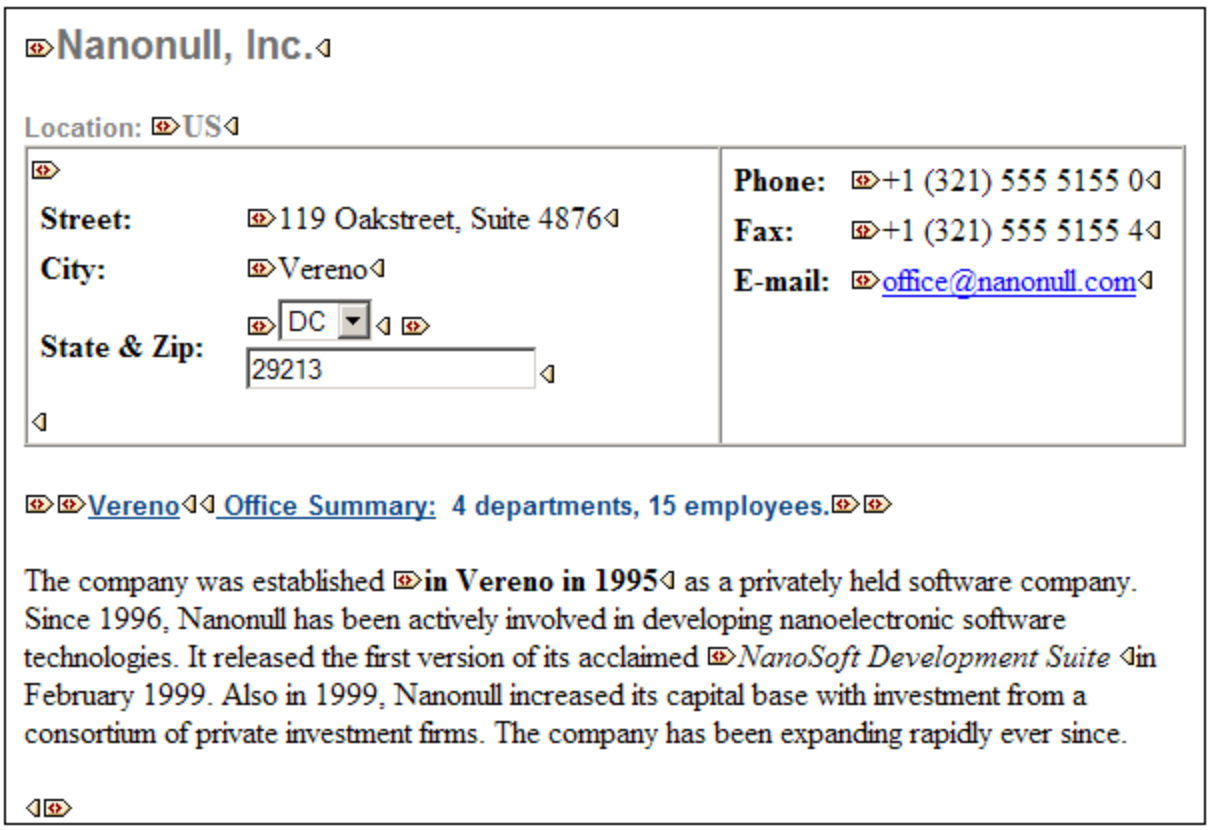


Im Modus für große Markup-Symbole werden Attribute durch das Ist-Gleich-Symbol im Start- und End-Tag gekennzeichnet:



Kleine Markup-Symbole

In diesem Modus werden die Start- und End-Tags von Elementen/Attributen ohne Namen angezeigt:



Beachten Sie, dass Start-Tags ein Symbol enthalten, während End-Tags leer sind. Außerdem haben Element-Tags eine eckige Klammer im Symbol, während Attribut-Tags als Symbol ein Ist-Gleich-Zeichen haben (*siehe Abbildung unten*).



Um ein Element/Attribut zu reduzieren bzw. zu erweitern, doppelklicken Sie auf den entsprechenden Tag. Im unten gezeigten Beispiel sehen Sie ein reduziertes Element (blau markiert). Beachten Sie die Form des Tags des reduzierten Elements und die des Start-Tags des erweiterten Elements links davon.



Gemischte Markup-Symbole

Im gemischten Markup-Modus wird eine benutzerdefinierte Markup-Ansicht angezeigt. Der Autor des StyleVision Power Stylesheet kann für einzelne Elemente/Attribute im Dokument große, kleine oder gar keine Markup-Symbole einstellen. Der Benutzer der Authentic-Ansicht sieht diese benutzerdefinierte Ansicht im "Gemischten Markup"-Ansichtsmodus.

© 2016-2022 Altova GmbH

Alle Markup-Symbole ausblenden

In diesem Modus werden alle XML-Markup-Symbole ausgeblendet. Da die Formatierung in der Authentic-Ansicht die Formatierung des gedruckten Dokuments ist, handelt es sich bei diesem Modus um eine WYSIWYG-Ansicht des Dokuments.

Anzeige von Inhalt

In der Authentic-Ansicht wird Inhalt auf zwei Arten angezeigt:

- Reiner Text. Sie geben den Text ein und dieser Text wird der Inhalt des Elements bzw. der Wert des Attributs.



- Dateneingabe-Hilfen. Dabei handelt es sich entweder um ein Eingabefeld (Textfeld), ein Eingabefeld mit mehreren Zeilen, eine Dropdown-Liste, ein Kontrollkästchen oder ein Optionsfeld. Text, den Sie in Eingabefelder und Eingabefelder mit mehreren Zeilen eingeben, wird der XML-Inhalt des Elements bzw. der Wert des Attributs.



  Bei den anderen Dateneingabe-Hilfen wird durch Ihre Auswahl der entsprechende XML-Wert erzeugt, der im StyleVision Power Stylesheet definiert wurde. In einer Auswahlliste könnte also beispielsweise "approved" (das in der Dropdown-Liste der Auswahlliste verfügbar wäre) ausgewählt werden, und dieser Eintrag könnte auf einen XML-Wert "1" oder auf "approved" oder einen beliebigen anderen Eintrag gemappt werden, während "not approved" dem Wert "0" oder "not approved" oder einem beliebigen anderen Wert zugeordnet würde.



Optionale Nodes

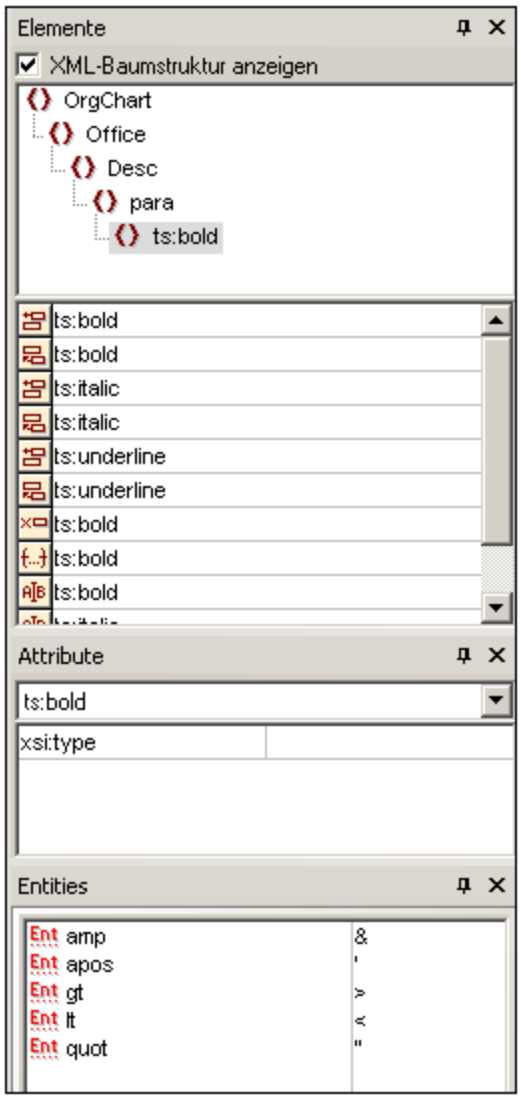
Wenn ein Element oder Attribut gemäß dem referenzierten Schema **optional** ist, wird eine Eingabeaufforderung vom Typ "add [*Element/Attribut*]" angezeigt.



Wenn Sie auf die Eingabeaufforderung klicken, wird das Element hinzugefügt und der Cursor so gesetzt, dass Daten eingegeben werden können. Wenn es mehrere optionale Nodes gibt, wird die Aufforderung "add..." angezeigt. Wenn Sie darauf klicken, wird ein Menü mit den optionalen Nodes angezeigt.

11.2.4 Authentic-Ansicht Eingabehilfen

Die Authentic-Ansicht verfügt über drei Eingabehilfen: für Elemente, Attribute und Entities. Sie werden als Fenster am rechten Rand der Authentic-Ansicht angezeigt. (*siehe Abbildung unten*).



Die Element- und Attribut-Eingabehilfen sind kontextsensitiv, d.h. was in der Eingabehilfe angezeigt wird, hängt davon ab, an welcher Stelle im Dokument der Cursor sich befindet. Die in der Entities-Eingabehilfe angezeigten Entities sind nicht kontextsensitiv; es werden - unabhängig von der Cursorposition - alle im Dokument zulässigen Entities angezeigt.

Im Folgenden werden die einzelnen Eingabehilfen beschrieben.

Element-Eingabehilfe

Die Element-Eingabehilfe besteht aus zwei Teilen:

© 2016-2022 Altova GmbH

- dem oberen Teil mit einer XML-Baumstruktur, der über das Kontrollkästchen **Show xml tree** ein- und ausgeblendet werden kann. In der XML-Baumstruktur sehen Sie die übergeordneten Nodes bis hinauf zum Root-Element des aktuellen Elements des Dokuments. Wenn Sie auf ein Element in der XML-Baumstruktur klicken, werden die dazugehörigen Elemente (wie in nächsten Punkt dieser Liste beschrieben) im unteren Teil der Element-Eingabehilfe angezeigt.
- Im unteren Teil der Element-Eingabehilfe werden die Elemente aufgelistet, die im, vor und nach dem ausgewählten Element/Textbereich in der Authentic-Ansicht eingefügt werden können, entfernt werden können oder darauf (d.h. durch Ersetzung) angewendet werden können. Was Sie mit einem in der Eingabehilfe aufgelisteten Element tun können, wird in der Eingabehilfe durch das Symbol auf der linken Seite des Elementnamens angezeigt. Im Folgenden finden Sie eine Liste der in der Element-Eingabehilfe vorkommenden Symbole mit einer Erklärung dazu.

Um einen Node aus der Eingabehilfe zu verwenden, klicken Sie auf sein Symbol.

 *Hinter Element einfügen*

Das Element in der Eingabehilfe wird hinter dem ausgewählten Element eingefügt. Beachten Sie, dass es auf der richtigen hierarchischen Ebene angehängt wird. Wenn sich Ihr Cursor z.B. innerhalb eines `//sect1/para`-Elements befindet und Sie ein `sect1`-Element anhängen, wird das neue `sect1`-Element nicht als Geschwisterelement von `//sect1/para` angehängt, sondern als `sect1`-Element, das ein Parent dieses `para`-Elements ist.

 *Vor Element einfügen*

Das Element in der Eingabehilfe wird vor dem ausgewählten Element eingefügt. Beachten Sie, dass das Element genau wie beim vorhergehenden Befehl auf der richtigen hierarchischen Ebene eingefügt wird.

 *Element entfernen*

Löscht des Element und dessen Inhalt.

 *Element einfügen*

Sie können ein Element aus der Eingabehilfe auch **innerhalb** eines Elements einfügen. Wenn Sie den Cursor in ein Element hineinsetzen, können die zulässigen Child-Elemente dieses Elements eingefügt werden. Beachten Sie, dass zulässige Child-Elemente sowohl Teil eines "elements-only Content Model" als auch Teil eines "Mixed Content Model" (Text plus Child-Elemente) sein können.

Ein zulässiges Child-Element kann entweder dann eingefügt werden, wenn ein Textbereich markiert wurde oder wenn der Cursor als Einfügepunkt in den Text platziert wurde.

- Wenn ein Textbereich ausgewählt und ein Element eingefügt wurde, wird der Textbereich der Inhalt des eingefügten Elements.
- Bei Auswahl einer Cursorposition wird das Element an dieser Stelle eingefügt

Nachdem das Element eingefügt wurde, kann es durch Klicken auf eines der beiden Symbole zum Löschen eines Elements (in der Element-Eingabehilfe) gelöscht werden. Welches der beiden Symbole angezeigt wird, ist abhängig davon, ob Sie einen Textbereich auswählen oder ob Sie den Cursor als Einfügepunkt in den Text setzen (siehe unten).

 *Element anwenden*

Wenn Sie in Ihrem Dokument ein Element auswählen (indem Sie in der "Große Markup-Symbol anzeigen"-Ansicht entweder auf den Start- oder den End-Tag eines Elements klicken) und wenn dieses Element durch ein anderes Element ersetzt werden kann (z.B. kann in einem Element gemischten Inhalts wie `para`, ein `italic` Element durch das `bold` Element ersetzt werden), bedeutet dieses Symbol, dass das Element in der

Eingabehilfe auf das ausgewählte (Original-) Element angewendet werden kann. Der Befehl **Element anwenden** kann auch auf einen Textbereich innerhalb eines Elements gemischten Inhalts angewendet werden; der Textbereich wird als Inhalt des angewendeten Elements erstellt.

- Wenn das angewendete Element ein **Child-Element mit demselben Namen** wie das Child-Element des Originalelements hat und das Original-Element eine Instanz dieses Child-Elements enthält, dann wird das Child-Element des Originals im neuen Element-Inhalt beibehalten.
- Wenn das angewendete Element **kein Child-Element mit demselben Namen** wie das instantiierte Child-Element des Originalelements enthält, dann wird das instantiierte Child-Element des Originalelements als Geschwisterelement von etwaigen Child-Elementen oder Elementen des neuen Elements angehängt.
- Wenn das angewendete Element **ein Child-Element hat, für das es im Content Model des Originalelements kein Äquivalent gibt**, wird dieses Child-Element nicht direkt erstellt, doch bietet Ihnen die Authentic-Ansicht die Möglichkeit, dieses einzufügen.

Wenn Sie anstatt eines Elements einen Textbereich auswählen, wird bei Anwendung eines Elements auf diesen Bereich an dieser Stelle das angewendete Element erstellt, wobei der markierte Text dessen Inhalt wird. Wenn der Cursor sich an einem Einfügepunkt befindet, ist die Anwendung eines Elements nicht zulässig.

 *Element löschen*

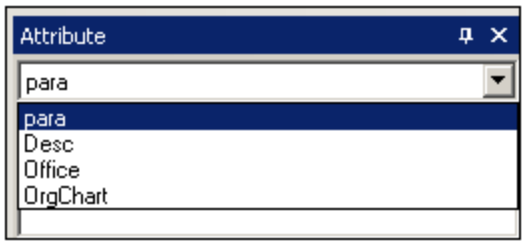
Dieses Symbol erscheint, wenn Text innerhalb eines Elements gemischten Inhalts ausgewählt wird. Wenn Sie auf dieses Symbol klicken, wird das Element, das den ausgewählten Bereich einschließt, gelöscht.

 *Element löschen (wenn ein Einfügepunkt ausgewählt ist)*

Dieses Symbol erscheint, wenn der Cursor in ein Element platziert wird, das ein Child eines Elements gemischten Inhalts ist. Wenn Sie auf das Symbol klicken, wird dieses zeileninterne Element gelöscht.

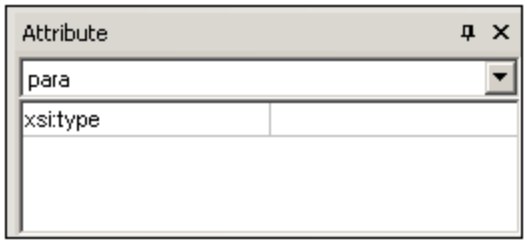
Attribut-Eingabehilfe

Die Attribut-Eingabehilfe besteht aus einem Listenfeld mit einer Dropdown-Liste von Attributen. Das ausgewählte Element (Sie können auf den Start- oder End-Tag klicken oder den Cursor irgendwo im Element-Inhalt platzieren) wird in der Dropdown-Liste angezeigt. Die in den nachfolgenden Abbildungen gezeigte Attribut-Eingabehilfe hat ein `para`-Element in der Dropdown-Liste. Wenn Sie auf den Abwärtspfeil in der Liste klicken, erscheint eine Liste aller **übergeordneten Elemente** des `para`-Elements hinauf bis zum Root-Element, in diesem Fall OrgChart.



Unterhalb der Dropdown-Liste wird eine Liste gültiger Attribute für dieses Element - in diesem Fall für das `para`-Element angezeigt. Wenn ein Attribut für ein bestimmtes Element obligatorisch ist, wird es fett angezeigt. (Im Beispiel unten gibt es keine obligatorischen Attribute mit Ausnahme des vordefinierten Attributs `xsi:type`)

© 2016-2022 Altova GmbH



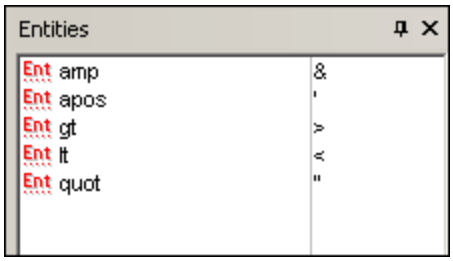
Um den Wert für ein Attribut einzugeben, klicken Sie in das Wertefeld des Attributs und geben Sie den Wert ein. Daraufhin wird das Attribut mit dem entsprechenden Wert im XML-Dokument erstellt.

Beachten Sie die folgenden Punkte:

- Im Falle des Attributs `xsi:nil`, welches in der Eingabehilfe "Attribut" erscheint, wenn ein nillable Element ausgewählt wurde, kann der Wert des Attributs `xsi:nil` nur durch Auswahl eines der zulässigen Werte (`true` oder `false`) aus der Dropdown-Liste der Werte des Attributs eingegeben werden.
- Das `xsi:type` Attribut kann durch Klicken in das Wertefeld des Attributs und entweder (i) Auswahl eines der aufgelisteten Werte aus der Dropdown-Liste oder (ii) durch Eingabe eines Werts geändert werden. Bei den in der Dropdown-Liste aufgelisteten Werten handelt es sich um die verfügbaren abstract types, die in dem XML-Schema, auf dem das Dokument in der Authentic-Ansicht basiert, definiert sind.

Entities-Eingabehilfe

Über die Entities-Eingabehilfe können Sie eine Entity in Ihr Dokument einfügen. Entities dienen zum Einfügen von Sonderzeichen oder Textfragmenten, die in einem Dokument öfter vorkommen (z.B. der Name einer Firma). Um eine Entity einzufügen, setzen Sie den Cursor an die gewünschte Stelle im Text und doppelklicken Sie in der Entities-Eingabehilfe auf die Entity.



Anmerkung: Eine interne Entity ist eine Entity, deren Wert in der DTD definiert wurde. Eine externe Entity ist eine Entity, deren Wert in einer externen Quelle z.B. einer anderen XML-Datei enthalten ist. Sowohl interne als auch externe Entities werden in der Entities-Eingabehilfe aufgelistet. Beim Einfügen einer Entity - egal ob intern oder extern - wird die Entity und nicht ihr Wert in den XML-Text eingefügt. Wenn es sich um eine interne Entity handelt, wird in der Authentic-Ansicht **der Wert der Entity** angezeigt. Dies bedeutet, dass eine XML-Datei, die eine externe Entity ist, in der Authentic-Ansicht als Entity angezeigt wird; die Entity wird in der Authentic-Ansicht nicht durch ihren Inhalt ersetzt.

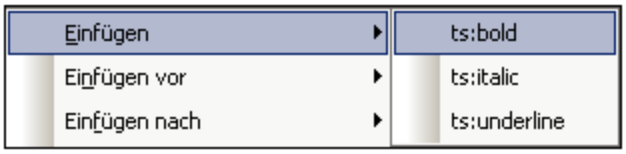
Sie können in der Authentic-Ansicht auch Ihre **eigenen Entities definieren,** die dann in der Eingabehilfe angezeigt werden. Nähere Informationen dazu finden Sie im Abschnitt "Bearbeitung in der Authentic -Ansicht" unter Definieren von Entities 659.

11.2.5 Authentic-Ansicht Kontextmenüs

Wenn Sie mit der rechten Maustaste auf einen ausgewählten Dokumenteninhalt oder Node klicken, wird ein Kontextmenü mit Befehlen, die für die Auswahl oder Cursorposition relevant sind, angezeigt.

Einfügen von Elementen

In der Abbildung unten sehen Sie das Kontextmenü mit dem Untermenü **Einfügen**, das eine Liste aller Elemente enthält, die an der aktuellen Cursorposition eingefügt werden können. Im Untermenü **Einfügen vor** werden alle Elemente aufgelistet, die vor dem aktuellen Element eingefügt werden können. Im Untermenü **Einfügen nach** werden alle Elemente aufgelistet, die nach dem aktuellen Element eingefügt werden können. In der Abbildung unten ist das aktuelle Element das `para`-Element. Die Elemente `italic` und `bold` können innerhalb des aktuellen `para`-Elements eingefügt werden.



Die Elemente `para` und `Office` können vor dem `para`-Element eingefügt werden.



Die Befehle zum Einfügen eines Node, zum Ersetzen eines Node (**Anwenden**) und zum Entfernen von Markup-Code (**Löschen**), die im Kontextmenü zur Verfügung stehen, stehen auch in den Eingabehilfen der Authentic-Ansicht[630] zur Verfügung und werden im entsprechenden Abschnitt ausführlich beschrieben.

Einfügen einer Entity

Wenn Sie den Cursor über den Befehl "Entity einfügen" positionieren, erscheint ein Untermenü mit einer Liste aller deklarierten Entities. Wenn Sie auf eine Entity klicken, wird sie am ausgewählten Node eingefügt. Eine Beschreibung, wie man Entities für das Dokument definiert, finden Sie unter Definieren von Entities[659].

Einfügen eines CDATA-Abschnitts

Dieser Befehl ist aktiviert, wenn Sie den Cursor innerhalb von Text setzen. Wenn Sie auf den Befehl klicken, wird am Cursor-Punkt ein CDATA-Abschnitt eingefügt. Der CDATA-Abschnitt steht innerhalb eines Start- und End-Tags; um diese Tags anzuzeigen, aktivieren Sie die großen oder kleinen Markup-Symbole. Innerhalb von CDATA-Abschnitten werden XML-Markup-Symbole und Parsing ignoriert. XML-Markup-Zeichen (Zeichen wie das kaufmännische Und, Apostroph, größer als, kleiner als und Anführungszeichen) werden nicht als Markup behandelt sondern als Literalzeichen. Daher eignen sich CDATA-Abschnitte für Text wie z.B. Auflistungen von Programmcode, welche XML-Markup-Zeichen enthalten.

Entfernen eines Node

Wenn Sie den Mauszeiger über den Befehl **Entfernen** positionieren, erscheint eine Menüliste bestehend aus dem ausgewählten Node und allen seinen übergeordneten Nodes bis hinauf zum Dokument-Element, die

© 2016-2022 Altova GmbH

entfernt werden können (ohne dass das Dokument ungültig wird). Klicken Sie auf das Element, das Sie entfernen möchten. Auf diese Art lässt sich ein Element oder ein beliebiges übergeordnetes Element einfach entfernen. Beachten Sie, dass bei Entfernung eines übergeordneten Elements auch alle untergeordneten Elementen einschließlich des ausgewählten Elements entfernt werden.

Löschen

Mit dem Befehl "Löschen" wird der Element-Markup-Code rund um die Auswahl gelöscht. Wenn der gesamte Node ausgewählt ist, wird der Element-Markup-Code für den gesamten Node gelöscht. Wenn ein Textabschnitt ausgewählt ist, wird der Element-Markup-Code nur rund um diesen Textabschnitt gelöscht.

Übernehmen

Mit dem Befehl **Übernehmen** wird ein ausgewählten Element in Ihre Auswahl im Hauptfenster übernommen. Nähere Informationen dazu finden Sie unter Authentic-Ansicht Eingabehilfen 630.

Kopieren, Ausschneiden, Einfügen

Dies sind die Standard-Windows-Befehle. Beachten Sie allerdings, dass Text mit dem Befehl **Einfügen** entweder als XML oder als Text eingefügt wird, je nachdem, was der Autor des Stylesheet für das SPS als Ganzes definiert hat. Informationen, wie die Befehle **Als XML-Text kopieren** und **Als Text kopieren** funktionieren, finden Sie unmittelbar unterhalb dieses Abschnitts in der Beschreibung zu **Einfügen als**.

Einfügen als

Der Befehl **Einfügen als** bietet die Option, ein in die Zwischenablage kopiertes XML-Fragment in der Authentic-Ansicht entweder als XML oder als Text einzufügen. Wenn das kopierte Fragment als XML eingefügt wird, wird es zusammen mit dem XML-Markup-Code eingefügt. Wenn es als Text eingefügt wird, wird nur der Textinhalt des kopierten Fragments eingefügt (nicht aber der XML-Markup-Code). Möglich sind die folgenden Situationen:

- Ein ***ganzer Node*** wird in der Authentic-Ansicht ***zusammen mit seinen Markup-Tags*** markiert und in die Zwischenablage kopiert. (i) Der Node kann als XML an jeder Stelle, an der dieser Node gültig ist, eingefügt werden. Er wird nicht an einer ungültigen Stelle eingefügt. (ii) Wenn der Node als Text eingefügt wird, wird nur der *Textinhalt* des Node eingefügt (nicht die Markup-Codes); das Textinhalt kann an jeder Stelle im XML-Dokument eingefügt werden, an der Text eingefügt werden darf.
- Ein ***Textfragment*** wird in der Authentic-Ansicht markiert und in die Zwischenablage kopiert. (i) Wenn dieses Fragment als XML eingefügt wird, so werden die XML-Markup-Tags des Texts - obwohl diese mit dem Textfragment nicht explizit mitkopiert wurden - zusammen mit dem Text eingefügt, allerdings nur, wenn der XML-Node an der Stelle, an der das Fragment eingefügt wird, gültig ist. (ii) Wenn das Fragment als Text eingefügt wird, kann er an jeder Stelle im XML-Dokument eingefügt werden, an der Text eingefügt werden darf.

Anmerkung: Text wird in Nodes kopiert, in denen Text zulässig ist. Sie müssen daher sicher stellen, dass das Dokument durch den kopierten Text nicht ungültig wird. Der kopierte Text sollte daher:
(i) an der neuen Stelle lexikalisch gültig sein (so wären z.B. nicht numerische Zeichen in einem numerischen Node ungültig) und
(ii) den Node nicht auf andere Weise ungültig machen (so würde der Node z.B. bei Einfügen einer vierstelligen Zahl in einen Node, in dem nur dreistellige Zahlen zulässig sind, ungültig werden).

Anmerkung: Wenn das Dokument durch ein eingefügten Text auf irgendeine Weise ungültig wird, so wird der eingefügte Text rot markiert.

Löschen

Mit dem Befehl **Löschen** wird der ausgewählte Node und sein gesamter Inhalt gelöscht. Ein Node gilt in diesem Zusammenhang als ausgewählt, wenn der Cursor in den Node platziert wurde oder indem Sie entweder auf den Start- oder End-Tag des Node klicken.

© 2016-2022 Altova GmbH

11.3 Bearbeitung in der Authentic-Ansicht

In diesem Abschnitt werden wichtige Features der Authentic-Ansicht näher beschrieben. Bei den Features, die hier behandelt werden, handelt es sich entweder um häufig gebrauchte Funktionen oder solche, deren Funktionsweise und Konzepte näher erklärt werden müssen.

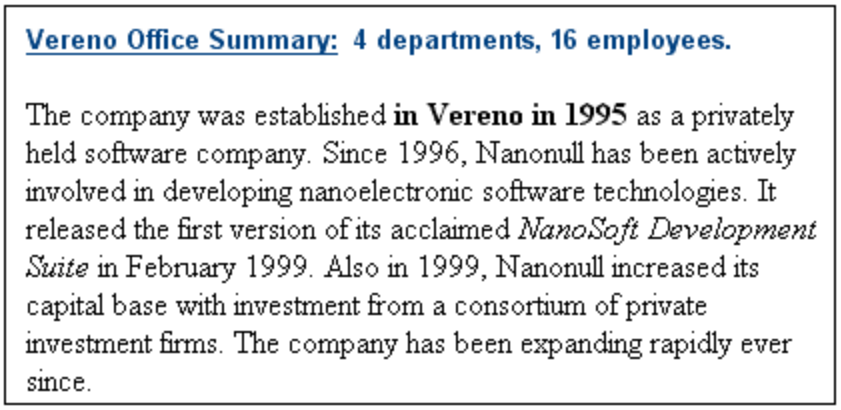
In diesem Abschnitt werden folgende Themen behandelt:

- Es gibt in der Authentic-Ansicht drei verschiedene Tabellenarten. Im Abschnitt Tabellen in der Authentic-Ansicht 642 werden diese drei Tabellenarten (statisches SPS, dynamisches SPS und XML) näher erläutert und deren Verwendung wird erklärt. Die Beschreibung beginnt mit einem allgemeinen Überblick. Anschließend wird näher auf die Einzelheiten eingegangen.
- Die Datumsauswahl ist ein grafischer Kalender, der Daten im korrekten XML-Format eingibt, wenn Sie auf ein Datum klicken. Siehe Datumsauswahl 657.
- Eine Entity ist ein Kürzel für ein Sonderzeichen oder einen Textstring. Sie können Ihre eigenen Entities definieren, um Sonderzeichen oder Textstrings durch Eingabe der Entity einfügen zu können. Nähere Informationen dazu finden Sie unter Definieren von Entities 659.
- In der Enterprise Edition von Altova-Produkten kann der Benutzer der Authentic-Ansicht XML-Dokumente mit digitalen XML-Signaturen signieren und diese Signaturen überprüfen.
- Die Bildformate 662, die in der Authentic-Ansicht angezeigt werden können.

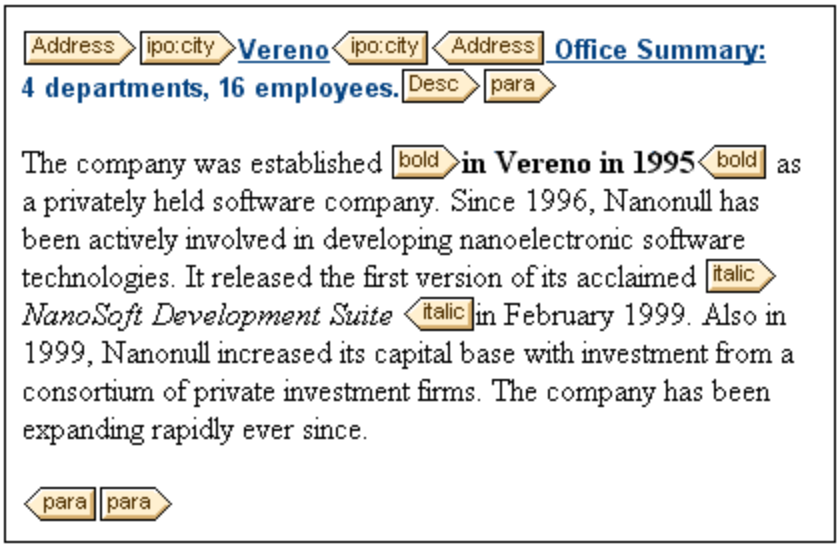
Altova Website: XML-Content-Bearbeitung, Erstellung von XML-Dokumenten

11.3.1 Grundlegendes zur Bearbeitung

Bei der Bearbeitung in der Authentic-Ansicht wird ein XML-Dokument bearbeitet. In der Authentic-Ansicht kann der Markup-Code, der die Struktur des XML-Dokuments definiert, jedoch ausgeblendet werden, sodass nur mehr der Inhalt des Dokuments angezeigt wird (*erste Abbildung unten*). Sie müssen sich daher nicht mit dem technischen Hintergrund von XML auseinandersetzen und können das Dokument wie ein normales Textdokument bearbeiten. Bei Bedarf können Sie den Markup-Code während der Bearbeitung jederzeit einblenden (*zweite Abbildung unten*).



Ein editierbares Dokument in der Authentic-Ansicht ohne XML-Markup-Tags



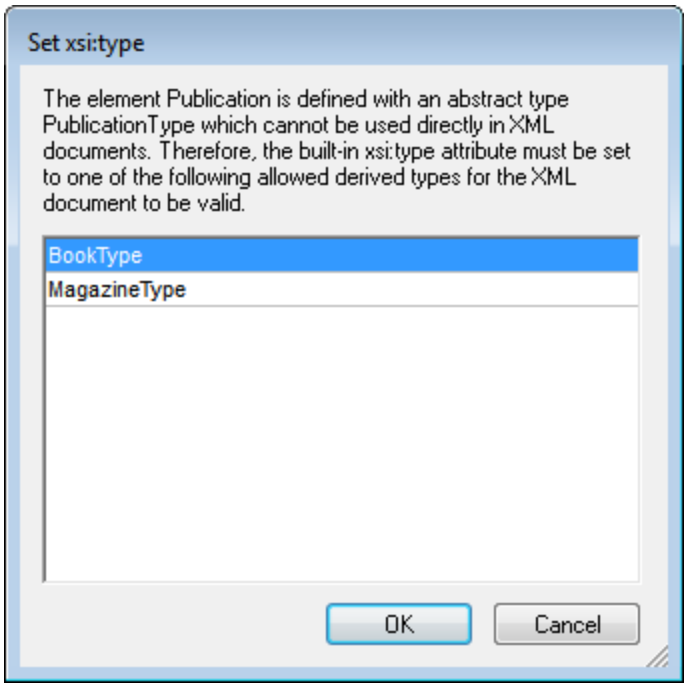
Ein editierbares Dokument in der Authentic-Ansicht mit XML-Markup-Tags

Einfügen von Nodes

Oft kommt es vor, dass ein neuer Node in das XML-Authentic-Dokument eingefügt werden muss. So muss z.B. möglicherweise ein neues `Person` Element zu einem Dokument vom Typ Adressbuch hinzugefügt werden. In solchen Fällen würde das XML-Schema das Hinzufügen des neuen Elements gestatten. Klicken Sie einfach mit der rechten Maustaste auf den Node im Authentic-Ansichtsdokument, vor bzw. hinter dem Sie den neuen Node hinzufügen möchten. Wählen Sie im Kontextmenü, das daraufhin angezeigt wird, je nach Bedarf den Befehl **Einfügen vor** oder **Einfügen nach** aus. Die Nodes, die an dieser Stelle in das Dokument eingefügt werden dürfen, werden in einem Untermenü aufgelistet. Klicken Sie auf den gewünschten Node, um ihn einzufügen. Der Node und alle obligatorischen untergeordneten Nodes werden eingefügt. Falls ein untergeordneter Node optional ist, wird ein Link `Add NodeName` angezeigt, auf den Sie klicken können, wenn der optionale Node eingefügt werden soll.

Wenn es sich beim hinzugefügten Node um ein Element mit einem abstract type handelt, so wird ein Dialogfeld (*wie in der Abbildung unten*) mit einer Liste von im XML-Schema verfügbaren derived types angezeigt.

© 2016-2022 Altova GmbH



Das oben gezeigte Dialogfeld wird angezeigt, wenn ein `Publication` Element hinzugefügt wird. Das `Publication` Element ist vom Typ `PublicationType`, einem abstract complexType. Die beiden complexTypes `BookType` und `MagazineType` werden vom abstract Type `PublicationType` abgeleitet. Wenn daher ein `Publication` Element zum XML-Dokument hinzugefügt wird, muss einer dieser beiden vom abstract Type `Publication` abgeleiteten concrete Types definiert werden. Das neue `Publication` Element wird mit einem `xsi:type` Attribut hinzugefügt.

```
<Publication xsi:type="BookType"> ... </Publication>
<Publication xsi:type="MagazineType"> ... </Publication>
 ...
<Publication xsi:type="MagazineType"> ... </Publication>
```

Wenn Sie einen der verfügbaren derived types auswählen und auf **OK** klicken, geschieht Folgendes:

- Der ausgewählte derived type wird als Wert des `xsi:type` Attributs des Elements ausgewählt
- Das Element wird zusammen mit seinen im Content Model des ausgewählten derived type definierten untergeordneten Nodes eingefügt.

Der ausgewählte derived type kann in der Folge durch Ändern des Werts des `xsi:type` Attributs des Elements in der Attribut-Eingabehilfe geändert werden. Wenn der Typ des Elements auf diese Art geändert wird, werden alle Nodes des früheren Content Model entfernt und die Nodes des Typs des neuen Content Model werden eingefügt.

Bearbeitung von Text

Ein Dokument in der Authentic-Ansicht besteht im Wesentlichen aus Text und Bildern. Um den Text im Dokument zu bearbeiten, platzieren Sie den Cursor an die Stelle im Text, an der Sie den Text einfügen möchten und geben Sie den Text ein. Sie können Text mit den gängigen Tastenkombinationen (wie z.B. der Entf-Taste) kopieren, verschieben und löschen und Drag-and-Drop-Mechanismen verwenden. Eine Ausnahme ist die **Eingabetaste**. Da das Dokument in der Authentic-Ansicht vorformatiert ist, werden und können

zwischen Einträgen keine Zeilen oder Leerzeichen eingefügt werden. Die **Eingabetaste** in der Authentic dient daher dazu, eine weitere Instanz des gerade bearbeiteten Elements einzufügen und sollte ausschließlich zu diesem Zweck verwendet werden.

Kopieren als XML oder als Text

Text kann als XML oder als Text kopiert und eingefügt werden.

- Wenn Text als XML eingefügt wird, so werden die XML Markup Tags zusammen mit dem Textinhalt von Nodes eingefügt. Die XML Markup Tags werden sogar dann eingefügt, wenn nur ein Teil des Node-Inhalts kopiert wurde. Damit der Markup-Code eingefügt werden kann, muss er dem Schema entsprechend an der Einfügestelle zulässig sein.
- Wenn Text als Text eingefügt wird, wird der XML Markup-Code nicht eingefügt.

Um Inhalt als XML oder als Text einzufügen, kopieren Sie zuerst den Text (**Strg+C)**, klicken Sie mit der rechten Maustaste an die Stelle, an der der Text eingefügt werden soll und wählen Sie im Kontextmenü den Befehl **Einfügen als | XML** oder **Einfügen als | Text**. Bei Verwendung des Tastaturkürzels **Strg+V** wird der Text im Standardeinfügemodus des SPS eingefügt. Der Standardeinfügemodus wird vom Ersteller des SPS definiert. Nähere Informationen dazu finden Sie im Abschnitt Kontextmenüs 634.

Alternativ dazu kann markierter Text auch mit der Maus an die gewünschte Einfügeposition gezogen werden. Wenn Sie die Maustaste loslassen, wird ein Popup-Fenster angezeigt, in dem Sie gefragt werden, ob der Text als Text oder als XML eingefügt werden soll. Wählen Sie die gewünschte Option aus.

Formatierung von Text

Eines der grundlegenden Prinzipien von XML-Dokumentsystemen ist, dass Inhalt und Darstellung separat gehalten werden. Das XML-Dokument enthält den Inhalt, während das Stylesheet die Darstellung (Formatierung) enthält. In der Authentic-Ansicht wird das XML-Dokument mit Hilfe eines Stylesheet dargestellt, d.h. die gesamte Formatierung, die Sie in der Authentic-Ansicht sehen, wird vom Stylesheet erzeugt. Wenn Sie Text fett angezeigt sehen, wurde die Fettformatierung vom Stylesheet vorgegeben. Wenn Sie eine Liste oder Tabelle sehen, wurde diese Darstellung vom Stylesheet vorgegeben. Das XML-Dokument, das Sie in der Authentic-Ansicht bearbeiten, enthält nur den Inhalt und keinerlei Formatierungen. Die Formatierung ist im Stylesheet definiert. Für Sie als Benutzer der Authentic-Ansicht bedeutet dies, dass Sie den von Ihnen bearbeiteten Text nicht formatieren müssen oder können. Sie bearbeiten nur den Inhalt. Die Formatierung, die automatisch auf den Inhalt angewendet wird, ist mit dem semantischen und/oder strukturellen Wert der Daten verknüpft, die Sie bearbeiten. So erhält beispielsweise eine E-Mail-Adresse (die als semantische Einheit betrachtet werden könnte) automatisch eine bestimmte Formatierung, weil es sich um eine E-Mail-Adresse handelt. Ebenso muss eine Überschrift (sowohl eine strukturelle als auch eine semantische Einheit) an einer bestimmten Stelle im Dokument vorkommen und wird automatisch so formatiert, wie dies der Designer des Stylesheet für Überschriften definiert hat. Sie können weder die Formatierung der E-Mail-Adresse noch die der Überschrift ändern. Sie ändern nur den Inhalt der E-Mail-Adresse oder der Überschrift.

In manchen Fällen muss Inhalt speziell formatiert werden; z.B. ein Textstring der fett gedruckt werden soll. In allen diesen Fällen muss die Darstellung mit einem Strukturelement des Dokuments verknüpft sein. So wird z.B. ein Textstring, der fett dargestellt werden soll, strukturell vom restlichen Inhalt durch Markup-Code getrennt, dem der Stylesheet Designer Fettformatierung zuweisen wird. Wenn Sie als Benutzer der Authentic-Ansicht einen solchen Textstring verwenden möchten, müssen Sie diesen in den entsprechenden Element Markup-Code einschließen. Nähere Informationen darüber finden Sie in dieser Dokumentation im Abschnitt Element-Eingabehilfe 630 unter dem Befehl "Element einfügen".

 © 2016-2022 Altova GmbH

Verwendung von RichEdit in der Authentic-Ansicht

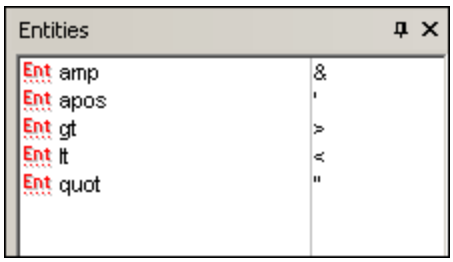
Wenn Sie den Cursor in der Authentic-Ansicht innerhalb von einem Element platzieren, das als RichEdit-Komponente erstellt wurde, werden die Schaltflächen und Steuerelemente in der RichEdit-Symbolleiste (*Abbildung unten*) aktiv. Anderfalls sind sie ausgegraut.



Wählen Sie den gewünschten Text aus und definieren Sie die Stile, die Sie darauf anwenden möchten, mit Hilfe der Schaltflächen und Steuerelemente der RichEdit-Symbolleiste. Mit Hilfe von RichEdit kann der Benutzer der Authentic-Ansicht die Schriftart (font), Schriftbreite (font-weight), Schriftverzierung (font-decoration), Schriftgröße (font-size), Hintergrundfarbe und Ausrichtung des Texts definieren. Der mit Stilen versehene Text wird in die Tags des Stilelements eingeschlossen.

Einfügen von Entities

Einige Zeichen sind in XML-Dokumenten für Markup-Tags reserviert und können in normalem Text nicht verwendet werden. Dies sind die Zeichen Kaufmännisches Und (`&`), Apostroph (`'`), kleiner als (`<`), größer als (`>`) und doppeltes Anführungszeichen (`"`). Wenn Sie diese Zeichen in Ihren Daten verwenden möchten, müssen Sie sie als Entity-Referenzen über die Eingabehilfe "Entities" 630 (*Abbildung unten*) einfügen.



XML bietet auch die Möglichkeit benutzerdefinierte Entities zu erstellen. Dabei kann es sich um Folgendes handeln: (i) Sonderzeichen, die auf Ihrer Tastatur nicht zur Verfügung stehen, (ii) Textstrings, die Sie in Ihrem Dokumentinhalt wiederverwenden möchten, (iii) XML-Datenfragmente oder (iv) andere Ressourcen, wie z.B. Bilder. Sie können in der Applikation Authentic-Ansicht Ihre eigenen Entities definieren 659. Nachdem Sie diese definiert haben, werden die Entities in der Eingabehilfe "Entities" 630 angezeigt und können dann in das Dokument eingefügt werden.

Einfügen von CDATA-Abschnitten

CDATA-Abschnitte sind Textabschnitte in einem XML-Dokument, die der XML-Parser nicht als XML-Daten verarbeitet. Mit Hilfe dieser Abschnitte können Sie große Textabschnitte auslassen, wenn die Ersetzung von Sonderzeichen durch Entity-Referenzen nicht wünschenswert ist; ein Beispiel wäre Programmcode oder ein XML-Fragment, das mit seinen Markup-Tags wiedergegeben werden soll. CDATA-Abschnitte können innerhalb von Elementinhalt vorkommen und werden am Beginn und Ende von `<![CDATA[` bzw. `]]>` eingeschlossen. Folglich sollte der Textstring `]]>` nicht in einem CDATA-Abschnitt vorkommen, da damit das Ende des Abschnitts gekennzeichnet würde. In diesem Fall kann das Zeichen "größer als" durch ihre Entity-Referenz (`>`) dargestellt werden. Um einen CDATA-Abschnitt in ein Element einzufügen, platzieren Sie den Cursor an der gewünschten Stelle, rechtsklicken Sie und wählen Sie im Kontextmenü den Befehl **CDATA-Abschnitt einfügen.** Um die CDATA-Abschnittstags in der Authentic-Ansicht anzuzeigen, wechseln Sie in die Markup-

Anzeige[624]. Alternativ dazu können Sie den Text markieren, der in einen DATA-Abschnitt eingeschlossen werden soll, und den Befehl **CDATA-Abschnitt einfügen** wählen.

Anmerkung: CDATA-Abschnitte können nicht in Eingabefelder (d.h. in ein- oder mehrzeilige Textfelder) eingefügt werden. CDATA-Abschnitte können nur in Elemente eingefügt werden, die in der Authentic-Ansicht als Komponenten des Textinhalts angezeigt werden

Bearbeiten von Links und Navigieren zum Ziel eines Links

Ein Hyperlink besteht aus zwei Teilen: dem Link-Text und dem Ziel des Links. Sie können den Link-Text bearbeiten, indem Sie auf den Text klicken und ihn ändern. Sie können jedoch das Ziel des Links nicht bearbeiten. (Das Ziel des Links wird vom Designer des Stylesheet definiert (entweder durch Eingabe einer statischen Zieladresse oder durch Abrufen der Zieladresse aus Daten im XML-Dokument). In der Authentic-Ansicht können Sie zum Ziel des Links navigieren, indem Sie die Strg-Taste drücken und auf den Link-Text klicken. (Beachten Sie: Wenn Sie nur auf den Link klicken, können Sie den Link-Text nur bearbeiten.)

11.3.2 Tabellen in der Authentic-Ansicht

Die drei Tabellentypen fallen in zwei Kategorien: SPS-Tabellen (statische und dynamische) und CALS/HTML-Tabellen.

SPS-Tabellen: Es gibt zwei Arten von SPS-Tabellen: statische und dynamische. SPS-Tabellen werden vom Autor des StyleVision Power Stylesheet, mit dem Ihr XML-Dokument verknüpft ist, erstellt. Sie können selbst keine SPS-Tabelle in das XML-Dokument einfügen, doch können Sie Daten in die Felder der SPS-Tabelle eingeben und Zeilen in dynamischen SPS-Tabellen hinzufügen und löschen. Im nachfolgenden Abschnitt zu SPS-Tabellen[642] werden die Funktionen dieser Tabellen näher beschrieben.

CALS/HTML-Tabellen werden vom Benutzer der Authentic-Ansicht eingefügt. Auf diese Art können Sie auf Wunsch an jeder Stelle im Dokument, an der dies zulässig ist, Tabellen einfügen. Die Editierfeatures von CALS/HTML-Tabellen[644] und die Bearbeitungssymbole für CALS/HTML-Tabellen[648] werden im Folgenden beschrieben.

11.3.2.1 SPS-Tabellen

In der Authentic-Ansicht werden zwei Arten von SPS-Tabellen verwendet: statische Tabellen und dynamische Tabellen.

Statische Tabellen

Statische Tabellen sind von der Struktur und der Art des Inhalts der Zellen unveränderbar. Als Benutzer der Authentic-Ansicht können Sie Daten in die Tabellenzellen eingeben, die Struktur der Tabellen (d.h. Hinzufügen von Zeilen oder Spalten usw.) oder den Inhaltstyp einer Zelle jedoch nicht verändern. Die Eingabe von Daten erfolgt entweder durch Eingabe von Text oder durch Auswahl von Optionen in Form von Kontrollkästchen, Optionsfeldern oder Dropdown-Listen. Nachdem Sie Daten eingegeben haben, können Sie diese bearbeiten.

© 2016-2022 Altova GmbH

Nanonull, Inc.

Street:	119 Oakstreet, Suite 4876	**Phone:**	+1 (321) 555 5155
City:	Vereno	**Fax:**	+1 (321) 555 5155 - 9
State & Zip:	DC 29213	**E-mail:**	office@nanonull.com

Anmerkung: Die Symbole oder Befehle zum Bearbeiten dynamischer Tabellen **dürfen nicht** zum Bearbeiten statischer Tabellen verwendet werden.

Dynamische Tabellen

Dynamische Tabellen enthalten Zeilen, die eine sich wiederholende Datenstruktur darstellen, d.h. jede Zeile weist dieselbe Datenstruktur auf (Dies ist bei statischen Tabellen nicht der Fall). Sie können daher Zeilen bearbeiten, also Zeilen anhängen, einfügen, nach oben oder nach unten verschieben und löschen. Diese Befehle stehen im Menü **Authentic** und als Symbole in der Symbolleiste (siehe Abb. unten) zur Verfügung.



Um diese Befehle zu verwenden, platzieren Sie den Cursor an eine beliebige Stelle in der gewünschten Zeile und wählen Sie den entsprechenden Befehl aus.

Administration

First	Last	Title	Ext	EMail	Shares	Leave		
						Total	Used	Left
Vernon	**Callaby**	Office Manager	581	v.callaby@nanonull.com	1500	25	4	21
Frank	Further	Accounts Receivable	471	f.further@nanonull.com	0	22	2	20
Loby	Matise	Accounting Manager	963	l.matise@nanonull.com	add Shares	25	7	18
Employees: 3 (20% of Office, 9% of Company)					**Shares: 1500 (13% of Office, 6% of Company)**			
Non-Shareholders: Frank Further, Loby Matise.								

Zum Navigieren innerhalb der Zellen einer Tabelle verwenden Sie die Pfeiltasten der Tastatur (nach oben, nach unten, nach links, nach rechts). Um von einer Zelle zur nächsten zu springen, drücken Sie die Tabulatortaste. Wenn Sie in der letzten Zelle der letzten Zeile die Tabulatortaste drücken, wird eine neue Zeile erstellt.

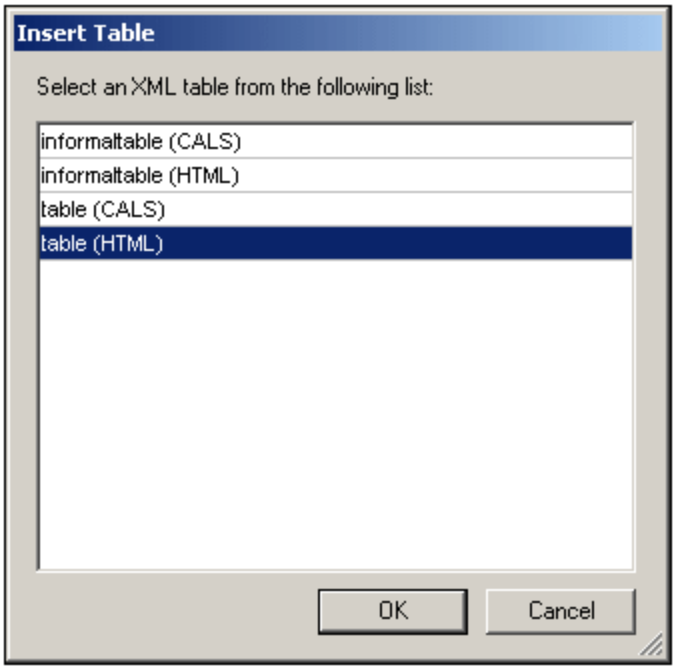
11.3.2.2 CALS/HTML-Tabellen

CALS/HTML-Tabellen können von Ihnen, dem Benutzer der Authentic-Ansicht für bestimmte XML-Datenstrukturen verwendet werden, die als Tabellenformat definiert wurden. Die Arbeit mit CALS/HTML-Tabellen erfordert drei Schritte: Einfügen der Tabelle, Formatierung der Tabelle und Eingabe von Daten. Die Befehle für die Arbeit mit CALS/HTML-Tabellen stehen in Form von Schaltflächen in der Symbolleiste zur Verfügung (siehe Symbole zur Bearbeitung von CALS/HTML-Tabellen 648).

Einfügen von Tabellen

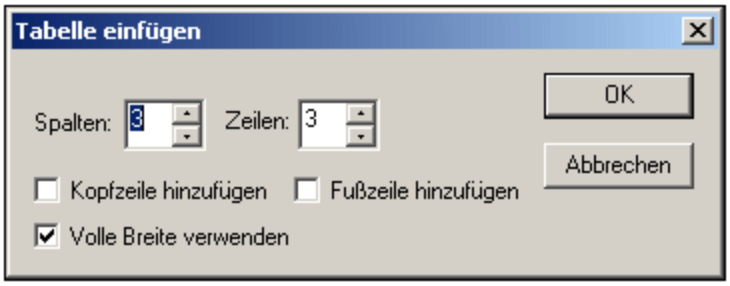
So fügen Sie eine CALS/HTML-Tabelle ein:

1. Setzen Sie Ihren Cursor an die Stelle, an der die Tabelle eingefügt werden soll und klicken Sie auf das Symbol . (Beachten Sie dass CALS/HTML-Tabellen nur an bestimmten im Schema definierten Stellen eingefügt werden können). Das Dialogfeld "Tabelle einfügen" (*Abbildung unten*) wird nun angezeigt. In diesem Dialogfeld werden alle XML-Element-Datenstrukturen, für die eine Tabellenstruktur definiert wurde, aufgelistet. So wurden z.B. in der Abbildung unten die Elemente `informaltable` und `table` sowohl als CALS-Tabelle als auch als HTML-Tabelle definiert.

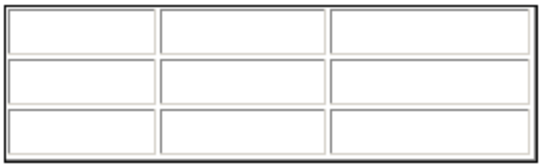


2. Wählen Sie den Eintrag mit dem gewünschten Element und Tabellenmodell aus und klicken Sie auf OK.
3. Wählen Sie im nächsten Dialogfeld (*Abbildung unten*) die Anzahl der Spalten und Zeilen aus und geben Sie an, ob eine Kopf- und/oder Fußzeile zur Tabelle hinzugefügt werden soll und ob die Tabelle sich über die gesamte Breite erstrecken soll. Klicken Sie anschließend auf **OK.**

© 2016-2022 Altova GmbH



Daraufhin wird anhand der Angaben oben die folgende Tabelle erstellt.



Über die Befehle im Menü **Tabelle** können Sie später Spalten hinzufügen und löschen, Zeilen und Spalten zusammenfügen oder trennen. Erstellen Sie daher zuerst die grundlegende Struktur.

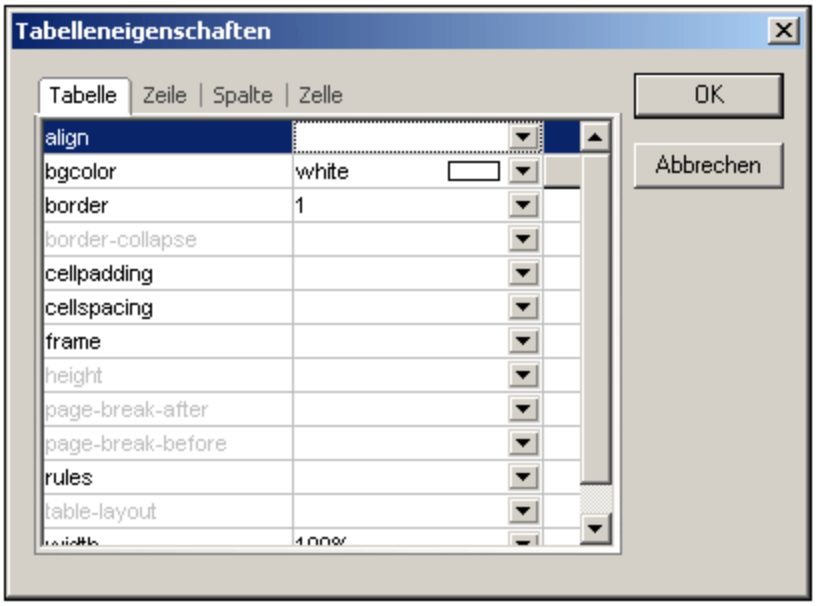
Formatieren von Tabellen und Eingabe von Daten

Die Tabellenformatierung wurde bereits im Dokumentdesign zugewiesen. Unter bestimmten Umständen können Sie die Tabellenformatierung jedoch ändern. Diese Umstände sind:

- Für die Elemente, die den verschiedenen Tabellenstrukturelementen entsprechen müssen (im zugrunde liegenden XML-Schema) die entsprechenden CALS- oder HTML-Tabelleneigenschaften als Attribute definiert sein. Nur die Attribute, die definiert sind, stehen für die Formatierung zur Verfügung. Wenn im Design Werte für diese Attribute definiert wurden, können Sie diese in der Authentic-Ansicht außer Kraft setzen.
- Im Design darf kein `style` Attribut, das CSS-Stile enthält, definiert worden sein. Wenn für ein Element ein Stilattribut mit CSS-Stilen definiert wurde, so hat das `style` Attribut Vorrang vor anderen für dieses Element gesetzten Formatierungsattributen. Folglich werden alle in der Authentic-Ansicht definierten Formatierungen außer Kraft gesetzt.

So formatieren Sie Ihre Tabelle, Zeile, Spalte oder Zelle:

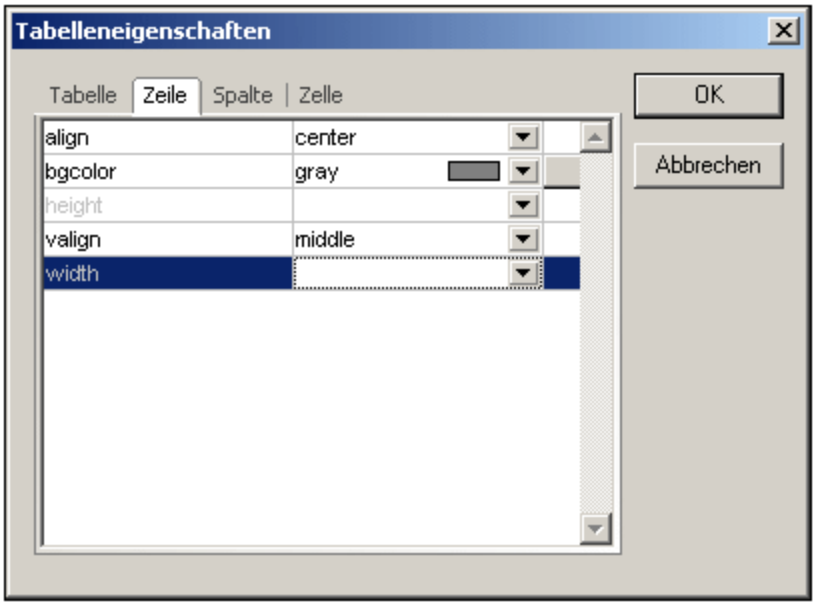
1. Platzieren Sie Ihren Cursor zum Formatieren der Tabelle an eine beliebige Stelle in der Tabelle und klicken Sie auf das Symbol  (Tabelleneigenschaften). Daraufhin erscheint das Dialogfeld "Tabelleneigenschaften" (siehe unten), in dem Sie die Formatierung für die Tabelle, oder für einzelne Zeilen, Spalten oder Zellen definieren können.



2. Setzen Sie die Eigenschaften "cellspacing" und "cellpadding" auf "0". Ihre Tabelle wird nun folgendermaßen aussehen:



3. Setzen Sie nun den Cursor in die erste Zeile, um diese zu formatieren und klicken Sie auf das Symbol  (Tabelleneigenschaften). Klicken Sie auf das Register "Zeile".



© 2016-2022 Altova GmbH

Da es sich bei der ersten Zeile um die Kopfzeile handelt, definieren Sie eine Hintergrundfarbe, um diese Zeile von den anderen Zeilen zu unterscheiden. Beachten Sie dass die Zeileneigenschaften in der Abbildung oben definiert wurden. Geben Sie anschließend den Text für die Spaltenüberschriften ein. Ihre Tabelle sieht nun folgendermaßen aus:

Name	Telephone	Email

Beachten Sie, dass die Ausrichtung wie angegeben mittig (centered) ist.

4. Angenommen, Sie möchten nun die Spalte "Telephone" in die Spalten "Office" und "Home" unterteilen. Um dies zu tun, müssen Sie die horizontale Breite der Spalte "Telephone" in zwei Spalten teilen. Zuerst werden wir allerdings die vertikale Größe der Kopfzeile aufteilen um eine Zeile für die Unter-Überschrift zu erstellen. Setzen Sie die Cursor in die Zelle "Telephone" und klicken Sie auf das Symbol  (Vertikal teilen). Ihre Tabelle sieht nun aus wie folgt:

Name	Telephone	Email

5. Setzen Sie nun den Cursor in die Zelle unterhalb der Zelle mit dem Eintrag "Telephone" und klicken Sie auf das Symbol  (Horizontal teilen). Geben Sie dann die Spaltenüberschriften "Office" und "Home" ein. Ihre Tabelle sieht nun folgendermaßen aus:

Name	Telephone		Email
	Office	Home	

Nun müssen Sie die horizontale Breite der einzelnen Zellen in der Spalte "Telephone" aufteilen.

Mit Hilfe der Schaltflächen zum Bearbeiten von Tabellen können Sie des Weiteren Spalten und Zeilen hinzufügen und löschen und Zellinhalte vertikal ausrichten. Die Schaltflächen zum Bearbeiten von CALS/HTML-Tabellen sind in der Benutzerreferenz im Abschnitt "Schaltflächen zum Bearbeiten von XML-Tabellen 648" näher beschrieben.

Navigieren innerhalb von Tabellenzellen

Um innerhalb der Zellen der CALS/HTML-Tabelle navigieren zu können, verwenden Sie die Pfeiltasten der Tastatur.

Eingabe von Daten in eine Zelle

Um Daten in eine Zelle einzugeben, setzen Sie den Cursor in eine Zelle und geben Sie die Daten ein.

Formatieren von Text

Text in einer CALS/HTML-Tabelle muss wie anderer Text im XML-Dokument mit Hilfe von XML-Elementen oder Attributen formatiert werden. Um ein Element hinzuzufügen, markieren Sie den Text und doppelklicken Sie auf das gewünschte Element in der Element-Eingabehilfe. Um einen Attributwert zu definieren, setzen Sie den Cursor in das Textfragment und geben Sie den erforderlichen Attributwert in der Attribut-Eingabehilfe ein. Nachdem Sie den Kopfzeilentext fett formatiert haben, sieht Ihre Tabelle folgendermaßen aus:

Name	Telephone		Email
	Office	Home	

Der Text in der Abbildung oben wurde durch Anklicken und Doppelklick auf das Element `strong` formatiert, für das eine globale Vorlage vorhanden ist, in der als Schriftbreite "Fett" definiert ist. Die Textformatierung wird sofort sichtbar.

Anmerkung: Damit die Textformatierung in der Authentic-Ansicht angezeigt wird, muss in StyleVision für das entsprechende Element eine globale Vorlage mit der erforderlichen Textformatierung erstellt worden sein.

11.3.2.3 Schaltflächen zum Bearbeiten von CALS/HTML-Tabellen

Die Befehle zum Bearbeiten von CALS/HTML-Tabellen stehen in Form von Symbolen in der Symbolleiste zur Verfügung. Beachten Sie, dass es zu diesen Symbolen keine Menübefehlentsprechungen gibt. Ausführliche Informationen zur Verwendung von CALS/HTML-Tabellen finden Sie unter CALS/HTML-Tabellen 644.

Tabelle einfügen



Der Befehl "Tabelle einfügen" fügt eine **CALS/HTML-Tabelle** an der aktuellen Cursorposition ein.

Tabelle löschen



Der Befehl "Tabelle löschen" löscht die derzeit aktive Tabelle.

Zeile anhängen



Der Befehl "Zeile anhängen" hängt eine Zeile an das Ende der derzeit aktiven Tabelle an.

Spalte anhängen



Der Befehl "Spalte anhängen" hängt eine Spalte an das Ende der derzeit aktiven Tabelle an.

Zeile einfügen



Der Befehl "Zeile einfügen" fügt eine Zeile oberhalb der aktuellen Cursorposition in der derzeit aktiven Tabelle ein.

© 2016-2022 Altova GmbH

Spalte einfügen



Der Befehl "Spalte einfügen" fügt eine Spalte links von der derzeit aktiven Cursorposition in der derzeit aktiven Tabelle ein.

Links zusammenfügen



Der Befehl "Links zusammenfügen" fügt die aktive Zelle (an der Cursorposition) mit der Zelle links davon zusammen. Die Tags von beiden Zellen bleiben in der neuen Zelle erhalten, die Spaltenüberschriften bleiben unverändert und werden zusammengeführt.

Rechts zusammenfügen



Der Befehl "Rechts zusammenfügen" fügt die aktive Zelle (an der Cursorposition) mit der Zelle rechts davon zusammen. Die Inhalte von beiden Zellen werden in der neuen Zelle zusammengeführt.

Unten zusammenfügen



Der Befehl "Unten zusammenfügen" fügt die aktive Zelle (an der Cursorposition) mit der Zelle unterhalb davon zusammen. Die Inhalte von beiden Zellen werden in der neuen Zelle zusammengeführt.

Oben zusammenfügen



Der Befehl "Oben zusammenfügen" fügt die aktive Zelle (an der Cursorposition) mit der Zelle oberhalb davon zusammen. Die Inhalte von beiden Zellen werden in der neuen Zelle zusammengeführt.

Horizontal teilen



Der Befehl "Horizontal teilen" erstellt eine neue Zelle rechts von der derzeit aktiven Zelle. Die Größe der beiden Zellen entspricht nun der Größe der Originalzelle

Vertikal teilen



Der Befehl "Vertikal teilen" erstellt eine neue Zelle unterhalb von der derzeit aktiven Zelle.

Bündig oben



Dieser Befehl richtet den Inhalt der Zellen bündig am oberen Zellenrand aus.

Vertikal zentrieren



Dieser Befehl zentriert den Inhalt der Zelle.

Bündig unten

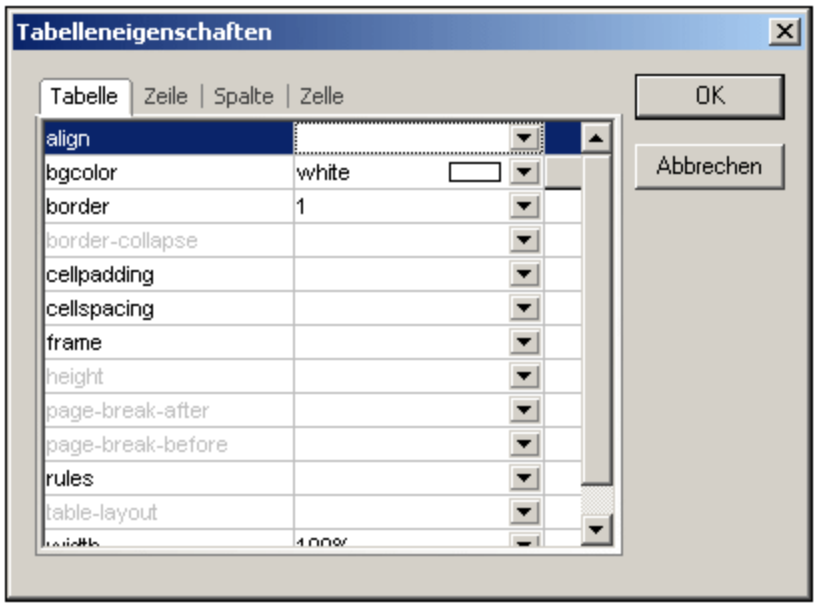


Dieser Befehl richtet den Inhalt der Zellen bündig am unteren Zellenrand aus.

Tabelleneigenschaften



Der Befehl "Tabelleneigenschaften" öffnet das Dialogfeld "Tabelleneigenschaften". Dieses Symbol wird nur bei HTML-Tabellen aktiv, bei CALS-Tabellen ist es deaktiviert.



11.3.3 Bearbeiten einer DB

In der Authentic-Ansicht können Sie Datenbank (DB)-Tabellen bearbeiten und die Daten in einer DB speichern. Dieser Abschnitt enthält eine ausführliche Beschreibung der Funktionen, die Ihnen zum Bearbeiten einer DB-Tabelle zur Verfügung stehen. Beachten Sie bitte die folgenden Punkte:

- Die Anzahl der Datensätze in einer DB-Tabelle, die in der Authentic-Ansicht angezeigt werden können, wurde unter Umständen vom Autor des StyleVision Power Stylesheet absichtlich eingeschränkt, um das Design kompakter zu gestalten. In diesem Fall wird anfangs nur diese beschränkte Anzahl an Datensätzen in der Authentic-Ansicht geladen. Sie können die restlichen Datensätze der DB-Tabelle mit Hilfe der Symbole zur Navigation in DB-Tabellenzeilen (*siehe* Navigieren in einer DB-Tabelle 650) laden und anzeigen.
- Sie können eine Datenbankabfrage 651 durchführen, um bestimmte Datensätze anzuzeigen.
- Sie können Datensätze hinzufügen, ändern und löschen und Ihre Änderungen in der DB speichern. Siehe Ändern einer DB-Tabelle 655.

Um ein DB-basiertes StyleVision Power Stylesheet in der Authentic Ansicht zu öffnen, klicken Sie auf **Authentic | Datenbank bearbeiten**, und suchen Sie nach dem gewünschten StyleVision Power Stylesheet.

Anmerkung: Daten aus einer SQLite-Datenbank können in der Authentic-Ansicht nicht bearbeitet werden. Wenn Sie versuchen, SQLite-Daten über die Authentic-Ansicht zu speichern, erscheint eine entsprechende Meldung.

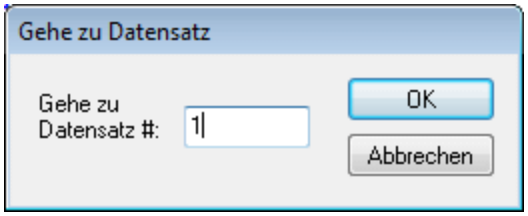
11.3.3.1 Navigieren in einer DB-Tabelle

Die Befehle zum Navigieren in den Zeilen einer DB-Tabelle stehen in Form von Schaltflächen im Authentic-Ansichtsdokument zur Verfügung. Normalerweise gibt es zu jeder DB-Tabelle einen Navigationsbereich mit vier oder fünf Schaltflächen.

© 2016-2022 Altova GmbH



Die Pfeilsymbole von links nach rechts sind: Gehe zum ersten Datensatz in der DB-Tabelle; Gehe zum vorhergehenden Datensatz; Dialogfeld "Gehe zu Datensatz #" öffnen (*siehe Abbildung unten*); Gehe zum nächsten Datensatz; und Gehe zum letzten Datensatz.



Um durch eine DB-Tabelle zu navigieren, klicken Sie auf die gewünschte Schaltfläche.

XML-Datenbanken

Im Fall von XML-Datenbanken wie z.B. IBM DB2 enthält eine Zelle (oder Zeile) ein einziges XML-Dokument. Daher wird immer nur eine einzige Zeile auf einmal in die Authentic-Ansicht geladen. Um ein XML-Dokument zu laden, das sich in einer anderen Zeile befindet, verwenden Sie die Menübefehl **Authentic | Neue Zeile mit XML-Daten für Bearbeitung auswählen**[1431].

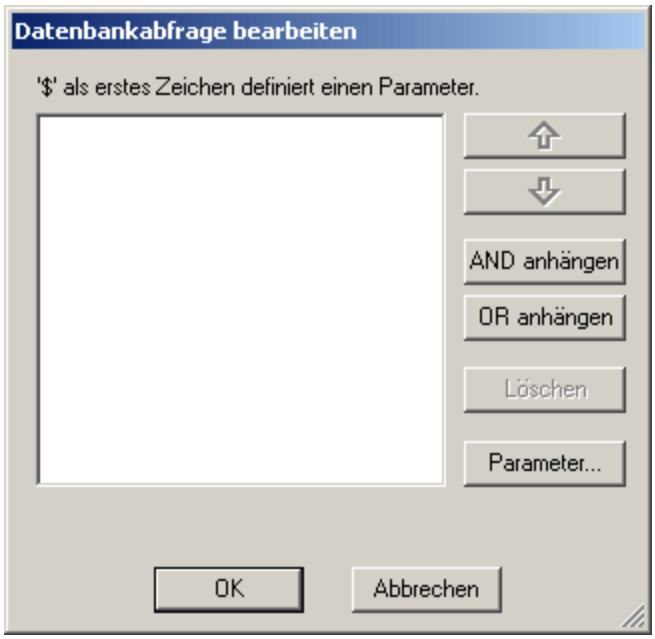
11.3.3.2 DB-Abfragen

Mit Hilfe einer DB-Abfrage können Sie die Datensätze einer in der Authentic-Ansicht angezeigten Tabelle abfragen. Eine Abfrage wird für eine einzelne Tabelle erstellt. Es kann für jede Tabelle nur eine Abfrage erstellt werden. Sie können während der Bearbeitung eines Dokuments jederzeit eine Abfrage erstellen. Wenn das in der Authentic-Ansicht angezeigte Dokument zu dem Zeitpunkt, zu dem Sie die Abfrage absenden, nicht gespeicherte Änderungen enthält, werden Sie gefragt, ob **alle** Änderungen im Dokument gespeichert oder verworfen werden sollen. Beachten Sie, dass auch an anderen Tabellen vorgenommene Änderungen gespeichert/verworfen werden. Nach Absenden der Abfrage wird die Tabelle mit den aktuellen Abfragekriterien erneut geladen.

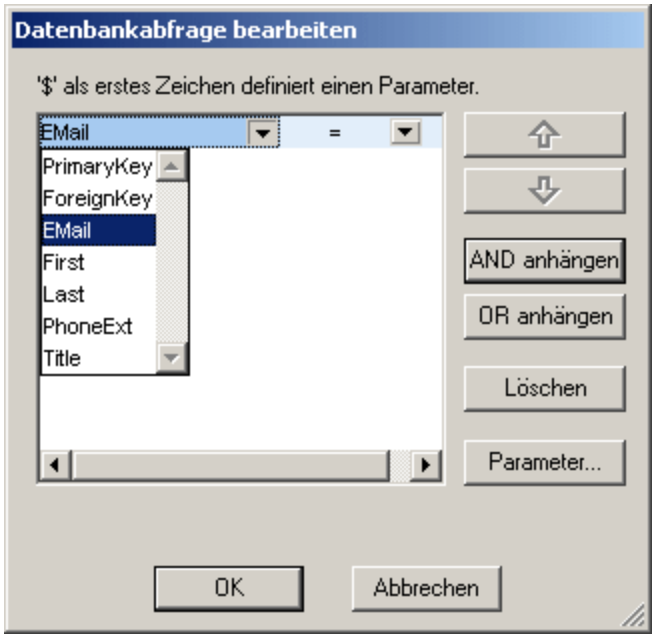
Anmerkung: Falls eine Meldung angezeigt wird, dass zu viele Tabellen geöffnet sind, können Sie die Anzahl der offenen Tabellen mit Hilfe einer Abfrage durch Filtern reduzieren.

So erstellen Sie eine Abfrage und senden diese ab:

1. Klicken Sie auf die Schaltfläche "Abfrage"  für die entsprechende Tabelle, um das Dialogfeld "Datenbankabfrage bearbeiten" aufzurufen (*siehe Abbildung unten).* Diese Schaltfläche erscheint normalerweise am oberen Rand einer DB-Tabelle oder unterhalb davon. Wenn keine Abfrage-Schaltfläche vorhanden ist, hat der Autor des StyleVision Power Stylesheet die DB-Abfragefunktion für diese Tabelle nicht aktiviert.



2. Klicken Sie auf die Schaltfläche **AND anhängen** oder **OR anhängen**. Daraufhin wird ein leeres Abfragekriterium (siehe unten) angehängt.



3. Geben Sie den Ausdruck für das Kriterium ein. Ein Ausdruck besteht aus: (i) einem Feldnamen (der in der dazugehörigen Dropdown-Liste zur Verfügung steht); (ii) einem Operator (der in der zugehörigen Dropdown-Liste zur Verfügung steht); und (iii) einem Wert (direkt einzugeben). Nähere Informationen zum Erstellen von Ausdrücken finden Sie im Abschnitt Ausdrücke in Kriterien[653] weiter unten.
4. Wenn Sie ein weiteres Kriterium hinzufügen möchten, klicken Sie - je nachdem, welchen logischen Operator (AND oder OR) Sie verwenden wollen, um Ihre Kriterien zu verknüpfen - auf die Schaltfläche **AND anhängen** bzw. **OR anhängen.** Fügen Sie nun das neue Kriterium hinzu. Nähere Informationen zu logischen Operatoren finden Sie im Abschnitt Neuordnen von Kriterien in DB-Abfragen[654].

 © 2016-2022 Altova GmbH

Ausdrücke in Kriterien

Ausdrücke in DB-Abfragekriterien bestehen aus einem Feldnamen, einem Operator und einem Wert. Als **Feldnamen** stehen die Child-Elemente der obersten Ebene der Datentabelle zur Verfügung; die Namen dieser Felder werden in der Dropdown-Liste aufgelistet (*siehe Abbildung oben)*. Unterhalb sehen Sie eine Liste der verfügbaren **Operatoren**:

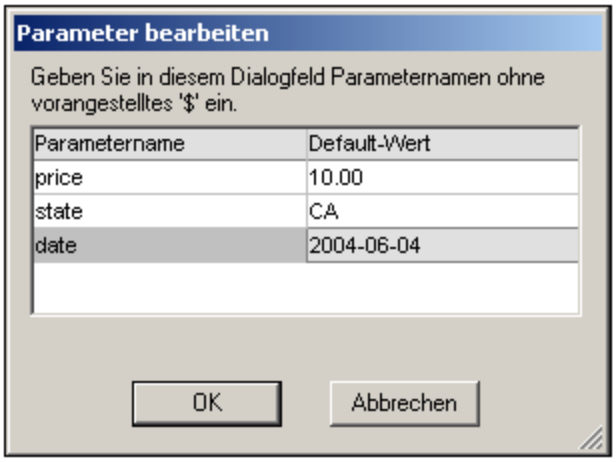
```
=             Ist gleich
<>            Ist nicht gleich
<             Kleiner als
<=            Kleiner oder gleich
>             Größer als
>=            Größer oder gleich
LIKE          Phonetisch ähnlich
NOT LIKE      Phonetisch nicht ähnlich
IS NULL       Ist leer
NOT NULL      Ist nicht leer
```

Wenn IS NULL oder NOT NULL ausgewählt ist, ist das Wertefeld deaktiviert. **Werte** müssen ohne Anführungszeichen (oder andere Begrenzungszeichen) eingegeben werden. Außerdem müssen Werte dieselbe Formatierung wie die des entsprechenden DB-Felds haben, sonst ist das Ergebnis des Ausdrucks `FALSE`. Wenn z.B. ein Kriterium für ein Feld des `date`-Datentyps in einer MS Access DB einen Ausdruck `StartDate=25/05/2004` enthält, ist das Ergebnis der Abfrage `FALSE`, da der `date`-Datentyp in einer MS Access DB das Format `JJJJ-MM-TT` hat.

Verwendung von Parametern mit DB-Abfragen

Sie können beim Erstellen von Abfragen den Namen eines **Parameters** als Wert eines Ausdrucks eingeben. Parameter sind Variablen, die anstelle von Literal-Werten in Abfragen verwendet werden können. Wenn Sie sie in einem Ausdruck eingeben, wird ihr Wert im Ausdruck verwendet. Parameter, die verfügbar sind, sind vom SPS-Designer im SPS definiert worden und können im Dialogfeld "Parameter anzeigen" (*siehe Abbildung unten*) angezeigt werden. Parametern wurde im SPS ein Standardwert zugewiesen, der überschrieben werden kann, indem dem Parameter über die Befehlszeile ein Wert zugewiesen wird (falls und während das Ausgabedokument über die Befehlszeile kompiliert wird).

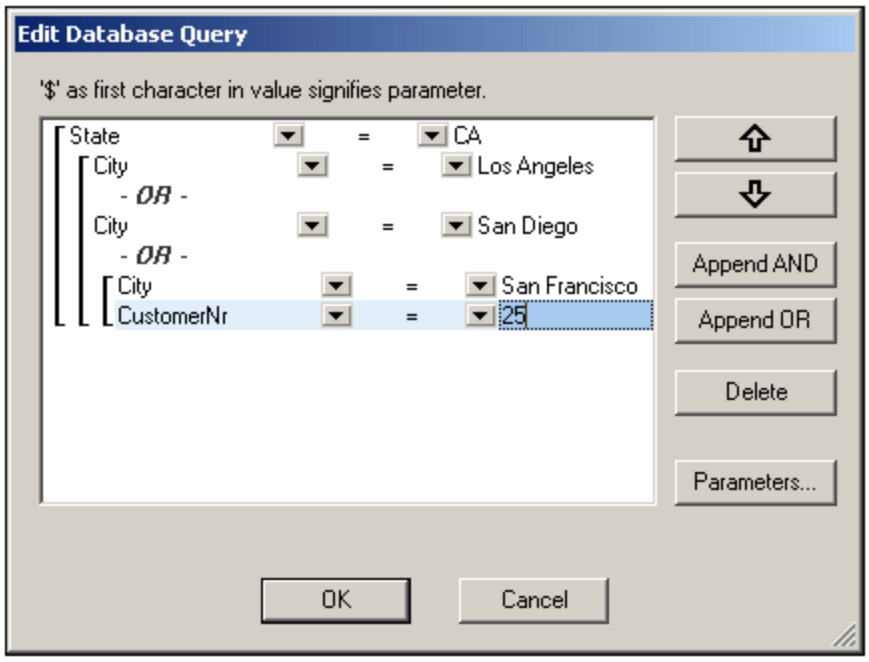
Um die Parameter, die für das SPS definiert wurden, anzuzeigen, klicken Sie im Dialogfeld **Datenbankabfrage bearbeiten** auf die Schaltfläche **Parameter**, um das unten gezeigte Dialogfeld **Parameter anzeigen** aufzurufen.



Das Dialogfeld "Parameter anzeigen" enthält **alle** Parameter, die für das Stylesheet definiert wurden und Parameter müssen im Stylesheet-Design bearbeitet werden.

Neuordnen von Kriterien in DB-Abfragen

Die logische Struktur der DB-Abfrage und die Beziehung zwischen zwei Kriterien oder Kriteriengruppen wird grafisch angezeigt. Jede Ebene der logischen Struktur wird durch eine eckige Klammer angezeigt. Zwei Kriterien oder Kriteriengruppen nebeneinander bedeuten, dass der AND-Operator verwendet wurde, während bei Trennung von zwei Kriterien durch das Wort OR der OR-Operator verwendet wurde. Die Kriterien dienen auch dazu, dem Benutzer einen klaren Überblick über die logische Struktur der DB-Abfrage zu verschaffen.



Die in der obigen Abbildung gezeigte DB-Abfrage würde in Textform folgendermaßen aussehen:

```
State=CA AND (City=Los Angeles OR City=San Diego OR (City=San Francisco AND
CustomerNr=25))
```


© 2016-2022 Altova GmbH

Sie können die DB-Abfrage umstrukturieren, indem Sie ein Kriterium oder eine Gruppe von Kriterien im Vergleich zu den anderen Kriterien in der DB-Abfrage nach oben oder unten verschieben. Um ein Kriterium oder eine Gruppe von Kriterien zu verschieben, gehen Sie vor wie folgt:

1. Wählen Sie das Kriterium durch Anklicken aus oder wählen Sie eine ganze Ebene aus, indem Sie auf die Klammer für diese Ebene klicken.
2. Klicken Sie im Dialogfeld auf den Nach oben- oder Nach unten-Pfeil.

Beachten Sie die folgenden Punkte:

- Wenn sich das in Richtung der Verschiebung benachbarte Kriterium auf derselben Ebene befindet, werden die beiden Kriterien vertauscht.
- Eine Gruppe von Kriterien (d.h. ein Kriterium mit einer Klammer) ändert seine Position innerhalb derselben Ebene; die Ebene ändert sich nicht.
- Ein einzelnes Kriterium ändert seine Position innerhalb derselben Ebene. Wenn sich das benachbarte Kriterium weiter außen/innen (d.h. nicht auf derselben Ebene) befindet, wird das ausgewählte Kriterium **immer um eine Ebene** nach außen/innen verschoben.

Um ein Kriterium in einer DB-Abfrage zu löschen, wählen Sie es aus und klicken Sie auf **Löschen**.

Ändern einer DB-Abfrage

So ändern Sie eine DB-Abfrage:

1. Klicken Sie auf die Schaltfläche "Abfrage" , um das Dialogfeld "Datenbankabfrage bearbeiten" aufzurufen. Sie können nun die Ausdrücke in jedem der aufgelisteten Kriterien ändern, neue Kriterien hinzufügen oder Kriterien aus der DB-Abfrage löschen.
2. Klicken Sie auf OK. Die Daten aus der DB werden automatisch in Authentic neu geladen, um die Änderungen an der DB-Abfrage anzuzeigen.

11.3.3.3 Ändern einer DB-Tabelle

Hinzufügen eines Datensatzes

So fügen Sie einen Datensatz hinzu:

1. Setzen Sie den Cursor in die gewünschte Zeile der DB-Tabelle und klicken Sie auf das Symbol , (um eine Zeile anzuhängen) oder  (um eine Zeile einzufügen). Daraufhin wird ein neuer Datensatz in der temporären XML-Datei erstellt.
2. Klicken Sie auf den Befehl **Datei | Speichern...**, um den neuen Datensatz zur Datenbank hinzuzufügen. In der Authentic-Ansicht wird in der Datenbank-Tabelle eine neue Zeile für den neuen Datensatz angehängt. Der `AltovaRowStatus` für diesen Datensatz wird auf `A` (für Added = Hinzugefügt) gesetzt.

Wenn Sie Daten für den neuen Datensatz eingeben, werden diese fett und unterstrichen angezeigt. Auf diese Art können Sie neu hinzugefügte Datensätze von vorhandenen unterscheiden - wenn vorhandene Datensätze nicht mit diesen Texteigenschaften formatiert wurden. Datentypfehler werden rot gekennzeichnet.

Der neue Datensatz wird zur DB hinzugefügt, wenn Sie **Datei | Speichern...** auswählen. Nachdem ein neuer Datensatz zur Datenbank hinzugefügt wurde, wird das `AltovaRowStatus`-Feld initialisiert (gekennzeichnet durch `---`) und der Datensatz wird in der Authentic-Ansicht als regulärer Datensatz angezeigt.

Ändern eines Datensatzes

Um einen Datensatz zu ändern, setzen Sie den Cursor an die gewünschte Stelle in der DB-Tabelle und nehmen Sie die gewünschten Änderungen vor. Wenn die Anzahl der Datensätze beschränkt ist, müssen Sie eventuell (siehe Navigieren in einer DB-Tabelle 650) zum gewünschten Datensatz navigieren.

Wenn Sie einen Datensatz bearbeiten, werden die Einträge in allen Feldern des Datensatzes unterstrichen und der `AltovaRowStatus` aller primären Instanzen dieses Datensatzes wird auf `U` (für Updated = Aktualisiert) gesetzt. Für alle sekundären Instanzen dieses Datensatzes wird der `AltovaRowStatus` auf `u` (klein geschrieben) gesetzt. Die primären und sekundären Instanzen eines Datensatzes werden durch die Struktur der DB definiert - und ebenfalls von dem davon generierten XML-Schema. Wenn z.B. eine Adresstabelle in einer Kundentabelle enthalten ist, kann die Adresstabelle im Design-Dokument in zwei Instantiierungstypen vorkommen: als Adresstabelle selbst und innerhalb von Instantiierungen der Kundentabelle. Derjenige der beiden Typen, der geändert wird, ist der primär geänderte Typ. Andere Typen - es kann mehr als einen anderen Typ geben - sind sekundäre Typen. Datentypfehler werden rot markiert.

Sie speichern die Änderungen in der DB durch Klicken auf **Datei | Speichern...**. Nachdem ein geänderter Datensatz in der DB gespeichert wurde, wird das `AltovaRowStatus` Feld initialisiert (gekennzeichnet durch `---`) und der Datensatz wird in der Authentic-Ansicht als regulärer Datensatz angezeigt.

Bitte beachten Sie die folgenden Punkte:

- Selbst wenn nur ein einziges Feld eines Datensatzes in der Authentic-Ansicht geändert wird, wird der gesamte Datensatz beim Speichern der Daten in der Datenbank aktualisiert.
- Der Datumswert `0001-01-01` ist bei einigen Datenbanken als `NULL`-Wert definiert und könnte daher eine Fehlermeldung verursachen.

Löschen eines Datensatzes

So löschen Sie einen Datensatz:

1. Setzen Sie den Cursor in die Zeile für den entsprechenden Datensatz und klicken Sie auf das Symbol . Der zu löschende Datensatz erscheint nun durchgestrichen. Der `AltovaRowStatus` erhält folgenden Wert: Primäre Instanzen des Datensatzes werden auf `D gesetzt;` sekundäre Instanzen auf `d`; indirekt gelöschte Datensätze werden mit `X` gekennzeichnet. Indirekt gelöschte Datensätze sind Felder im gelöschten Datensatz, die in einer separaten Tabelle enthalten sind. So könnte z.B. in einer Kundentabelle eine Adresstabelle enthalten sein. Wenn ein Kundendatensatz gelöscht würde, würde indirekt auch der entsprechende Adressdatensatz gelöscht. Wenn in der Kundentabelle ein Adressdatensatz gelöscht würde, würde der Adressdatensatz in der Kundentabelle primär gelöscht, jedoch würde derselbe Datensatz in einer unabhängigen Adresstabelle sekundär gelöscht, wenn diese instantiiert würde.
2. Sie speichern die Änderungen in der DB durch Klicken auf **Datei | Speichern**.

Anmerkung: Wenn Sie Daten in der DB speichern, wird der Rückgängig-Befehl zurückgesetzt. Sie können Operationen, die vor dem Speichern durchgeführt wurden, nicht mehr rückgängig machen.

© 2016-2022 Altova GmbH

11.3.4 Arbeiten mit Datumsangaben

Es gibt in der Authentic-Ansicht zwei Methoden, um Datumsangaben zu bearbeiten:

- Die Eingabe oder Änderung des Datums erfolgt über die Datumswahl[657].
- Das Datum wird durch Eingabe des Werts[658] eingegeben bzw. geändert.

Welche Methode der Benutzer der Authentic-Ansicht verwendet, ist im SPS definiert. Beide Methoden sind in den beiden Unterkapiteln dieses Abschnitts beschrieben.

Anmerkung zu Datumsformaten

Zur Speicherung des Datums im XML-Dokument stehen mehrere Datums-Datentypen zur Verfügung. Bei jedem dieser Datums-Datentypen muss das Datum in einem bestimmten lexikalischen Format gespeichert werden, damit das XML-Dokument gültig ist. So muss das Datum z.B. beim Datentyp `xs:date` das lexikalische Format `JJJJ-MM-TT` aufweisen. Wenn das Datum in einem `xs:date` Node anders als in diesem Format eingegeben wurde, ist das XML-Dokument ungültig.

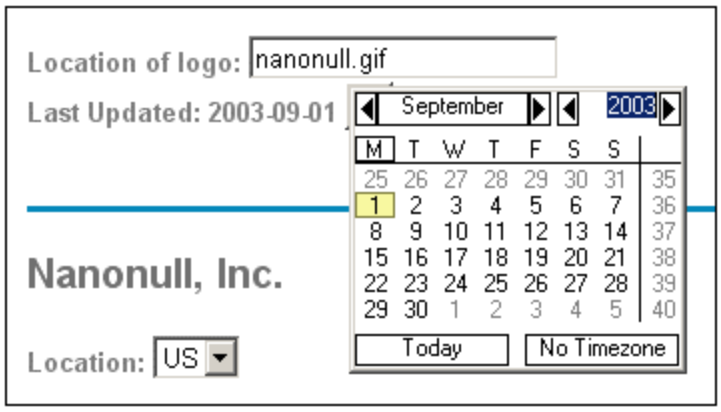
Um sicherzustellen, dass das Datum im richtigen Format eingegeben wird, kann der Designer des SPS die grafische Datumswahl in das Design integrieren. Dadurch wird sichergestellt, dass das mit der Datumswahl ausgewählte Datum im richtigen lexikalischen Format eingegeben wird. Falls keine Datumswahl vorhanden ist, sollte in der Authentic-Ansicht sichergestellt werden, dass das Datum im korrekten Format eingegeben wird. Bei der Validierung des XML-Dokuments erhalten Sie nützliche Tipps über das erforderliche lexikalische Format.

11.3.4.1 Datumswahl

Die Datumswahl ist ein grafischer Kalender zur Eingabe eines Datums in einem Standardformat in ein XML-Dokument. Es ist für die Verarbeitung des Dokuments wichtig, dass ein Standardformat verwendet wird. Das Datumswahlsymbol erscheint neben dem Datumsfeld, das dadurch geändert wird (siehe Abbildung unten).



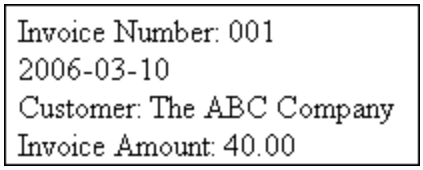
Um die Datumswahl (siehe unten) anzuzeigen, klicken Sie auf das Datumswahl-Symbol.



Um ein Datum auszuwählen, klicken Sie auf das gewünschte Datum, den gewünschten Monat bzw. das gewünschte Jahr. Das Datum wird in das XML-Dokument eingegeben und das angezeigte Datum wird entsprechend geändert. Falls nötig können Sie auch eine Zeitzone angeben.

11.3.4.2 Texteingabe

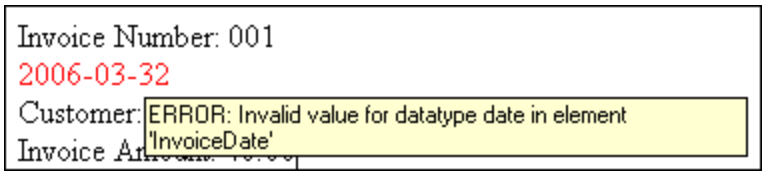
In Datumsfeldern, die keine Datumswahl haben (*siehe Abbildung*) können Sie das Datum direkt durch Eingabe des neuen Werts bearbeiten.



Fehler

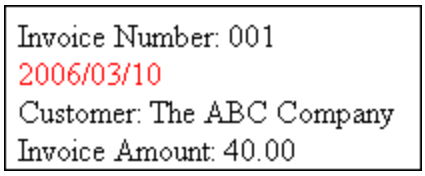
Die folgenden Arten von Fehlern werden gekennzeichnet:

- Wenn Sie ein Datum ändern und es sich daraufhin außerhalb des gültigen Bereichs für Datumswerte befindet, wird das Datum rot angezeigt, um den Fehler anzuzeigen. Wenn Sie den Mauszeiger über das ungültige Datum platzieren, erscheint eine Fehlermeldung (*siehe Abbildung*).

- Wenn Sie versuchen, das Datumsformat zu ändern, wird das Datum rot angezeigt. (So werden etwa in der Abbildung unten Schrägstriche anstelle von Bindestrichen verwendet).

© 2016-2022 Altova GmbH



11.3.5 Definieren von Entities

Entities

Unabhängig davon, ob Ihr Dokument auf einer DTD oder einem XML-Schema basiert, können Sie für die Verwendung in der Authentic-Ansicht Entities definieren. Diese Entities werden anschließend in der Entities-Eingabehilfe und im Kontextmenü-Untermenü von **Entity einfügen** angezeigt. Wenn Sie in der Entities-Eingabehilfe auf eine Entity doppelklicken, wird diese an der Cursorposition eingefügt.

Eine Entity eignet sich v.a. dann, wenn Sie einen Textstring, ein XML-Fragment oder einige andere externe Ressourcen an mehreren Stellen in Ihrem Dokument verwenden möchten. Die Entity, bei der es sich im Grunde um eine Kurzbezeichnung für die benötigten Daten handelt, wird im Dialogfeld "Entities definieren" definiert. Nachdem Sie die Entity definiert haben, können Sie diese an mehreren Stellen in Ihrem Dokument verwenden und dadurch Zeit sparen und effizienter arbeiten.

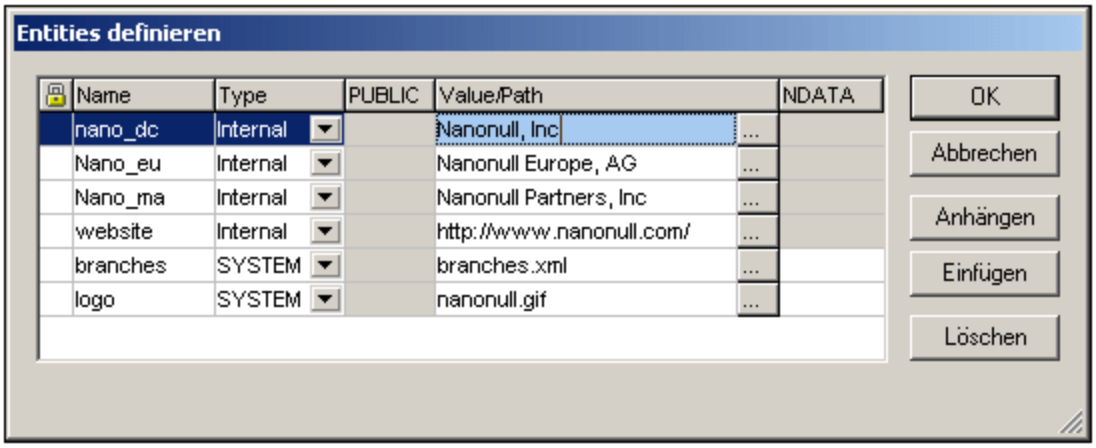
Entity-Arten

Sie können in Ihrem Dokument zwei grundlegende Arten von Entities verwenden: **geparste Entities**, wobei es sich um XML-Daten handelt (entweder einen Textstring oder ein Fragment aus einem XML-Dokument) oder **ungeparste Entities**, wobei es sich um nicht-XML-Daten handelt wie z.B. eine Binärdatei (normalerweise eine Grafik, Ton oder ein Multimedia-Objekt). Jede Entity hat einen Namen und einen Wert. Im Fall von geparsten Entities ist die Entity ein Platzhalter für die XML-Daten. Der Wert der Entity sind entweder die XML-Daten selbst oder eine URI, die auf eine `.xml`-Datei verweist, die die XML-Daten enthält. Bei ungeparsten Entities ist der Wert der Entity eine URI, die auf eine nicht-XML-Datei verweist.

Definieren von Entities

So definieren Sie eine Entity:

1. Klicken Sie auf **Authentic | XML-Entities definieren**. Daraufhin wird das Dialogfeld "Entities definieren" geöffnet. (*Abbildung unten*)



2. Geben Sie den Namen Ihrer Entity im Feld "Name" ein. Dies ist der Name, der in der Entities-Eingabehilfe angezeigt werden wird.
3. Wählen Sie den Typ der Entity aus der Dropdown-Liste im Feld "Type" aus. Es stehen die folgenden Typen zur Auswahl. Eine **interne** (internal) Entity ist eine Entity, für die der zu verwendende Text im XML-Dokument selbst gespeichert ist. Wenn Sie **PUBLIC** oder **SYSTEM** auswählen, wird angegeben, dass sich die Ressource außerhalb der XML-Datei befindet und mittels eines "Public Identifier" oder eines "System Identifier" gefunden wird. Ein System Identifier ist eine URI, die den Pfad der Ressource definiert. Ein Public Identifier ist ein pfadunabhängiger Identifier, der es manchen Prozessoren ermöglicht, die Ressource zu identifizieren. Wenn Sie sowohl einen Public als auch einen System Identifier angeben, wird der Public Identifier zum System Identifier aufgelöst und es wird der System Identifier verwendet.
4. Wenn Sie als Type PUBLIC ausgewählt haben, geben Sie den Public Identifier der Ressource im Feld PUBLIC ein. Wenn Sie Internal oder SYSTEM als Type ausgewählt haben, ist das Feld PUBLIC deaktiviert.
5. Im Feld "Value/Path" haben Sie folgende Eingabemöglichkeiten:

   - Wenn die Entity "Internal" ist, geben Sie den Textstring ein, der als Wert der Entity verwendet werden soll. Setzen Sie den Textstring nicht in Anführungszeichen. Alle Anführungszeichen werden als Teil des Textstrings behandelt.
   - Wenn der Entity-Typ SYSTEM ist, geben Sie die URI der Ressource ein oder wählen Sie mit Hilfe der Durchsuchen-Schaltfläche eine Ressource auf Ihrem lokalen Netzwerk aus. Wenn die Ressource geparste Daten enthält, muss es sich um eine XML-Datei handeln (d.h. die Datei muss die Erweiterung `.xml` haben). Als Alternative dazu kann die Ressource eine Binärdatei sein, z.B. eine GIF-Datei.
   - Wenn der Entity-Typ PUBLIC ist, müssen Sie zusätzlich einen System Identifier in dieses Feld eingeben.

6. Der Eintrag NDATA sagt dem Prozessor, dass diese Entity nicht geparst, sondern an den entsprechenden Prozessor gesendet werden soll. Das Feld "NDATA" muss daher einen Wert enthalten, der angibt, dass es sich um eine ungeparste Entity handelt.

Dialogfeldfunktionen

Im Dialogfeld "Entities definieren" können Sie Folgendes tun:

- Entities anhängen
- Entities einfügen
- Entities löschen

© 2016-2022 Altova GmbH

- Entities spaltenweise alphabetisch sortieren, indem Sie auf die entsprechende Spaltenüberschrift klicken; bei einem einmaligen Klick wird aufsteigend sortiert, bei zweimaligem Klick absteigend
- die Größe des Dialogfelds und die Spaltenbreite anpassen.
- Sperren. Sobald eine Entity im XML-Dokument verwendet wird, ist sie gesperrt und kann im Dialogfeld "Entities definieren" nicht bearbeitet werden. Gesperrte Entities werden durch das Schlosssymbol in der ersten Spalte gekennzeichnet. Durch das Sperren einer Entity wird sichergestellt, dass das Dokument in Bezug auf Entities gültig ist. (Das Dokument wäre ungültig, wenn eine Entity referenziert aber nicht definiert ist.)
- Doppelt vorhandene Entities werden markiert.

Einschränkungen für Entities

- Eine Entity innerhalb einer anderen Entity wird nicht aufgelöst - weder im Dialogfeld, noch in der Authentic-Ansicht, noch in der XSLT-Ausgabe, und das kaufmännische Und-Zeichen einer derartigen Entity wird in seiner Escape-Form, also `&` angezeigt.
- Externe ungeparste Entities, bei denen es sich nicht um Bilddateien handelt, werden in der Authentic-Ansicht nicht aufgelöst. Wenn für ein Bild im Design definiert ist, dass eine externe ungeparste Entity gelesen werden soll und wenn als URI ein Entity-Name (z.B. `'logo'`) definiert ist, so kann dieser Entity-Name im Dialogfeld "Entities definieren" (*siehe Abbildung oben*) als externe ungeparste Entity mit einem Wert, der zur URI der Bilddatei aufgelöst wird (wie diese für die Entity `logo` in der Abbildung oben geschehen ist), definiert werden.

11.3.6 XML-Signaturen

Bei der Erstellung eines SPS kann eine XML-Signatur für die Authentic-Ansicht konfiguriert werden. Wenn XML-Signaturen im SPS aktiviert sind, kann der Benutzer der Authentic-Ansicht die Authentic XML-Datei mit der aktivierten Signatur digital signieren. Wenn nach der Signierung des Dokuments Änderungen daran vorgenommen werden, schlägt die Überprüfung der Signatur fehl. Jedes Mal, wenn ein signiertes Authentic XML-Dokument in der Authentic-Ansicht eines beliebigen Altova-Produkts geöffnet wird, wird die Signatur des Dokuments überprüft und das Ergebnis der Überprüfung in einem Fenster angezeigt.

Anmerkung: XML-Signaturen können in der Authentic-Ansicht von Enterprise und Professional Editions der folgenden Altova-Produkte verwendet und überprüft werden: Authentic Desktop, Authentic Browser, XMLSpy und StyleVision

XML-Signaturaktionen

In der Authentic-Ansicht sind die folgenden Benutzeraktionen möglich:

- *Wählen des Zertifikats/Passworts:* Signaturen werden entweder anhand eines Zertifikats oder Passworts authentifiziert. Das Authentifikationsobjekt (Zertifikat oder Passwort) wird bei der Erstellung der Signatur und dann wieder bei der Überprüfung der Signatur benötigt. Wenn einem Authentic XML-Dokument ein SPS, für das Signaturen aktiviert wurden, zugewiesen wurde, so wurde dem SPS möglicherweise ein Standardzertifikat oder Passwort zugewiesen. Egal, ob ein Standardzertifikat bzw. ein Standardpasswort definiert wurde oder nicht, die Signatur kann so konfiguriert werden, dass der Benutzer der Authentic-Ansicht ein eigenes Zertifikat/Passwort auswählen kann. Er kann dies jederzeit im Dialogfeld "XML-Signatur" (*Abbildung unten*) tun und damit das Standardzertifikat/-passwort außer Kraft setzen, falls eines definiert wurde. Das eigene Zertifikat/Passwort des Benutzers wird im Arbeitsspeicher gespeichert und für die aktuelle Sitzung verwendet. Wenn der Benutzer der

Authentic-Ansicht ein eigenes Zertifikat/Passwort auswählt und die Datei oder die Applikation schließt, so wird das SPS wieder auf die Standardeinstellung für das Zertifikat/Passwort zurückgesetzt.



- *Signieren des Dokuments:* Das Authentic XML-Dokument kann entweder automatisch oder manuell signiert werden. Die automatische Signierung wird vom SPS-Designer in der Signaturkonfiguration definiert und bewirkt, dass das Authentic XML-Dokument beim Speichern automatisch signiert wird. Wenn die Option zur automatischen Signierung nicht aktiviert wurde, kann das Dokument manuell signiert werden. Klicken Sie dazu auf die Symbolleisten-Schaltfläche "XML-Signatur"  oder auf den Befehl **Authentic | XML-Signatur** und klicken Sie anschließend im Dialogfeld "XML-Signatur" (*Abbildung oben*) auf die Schaltfläche **Dokument signieren**. Beachten Sie Folgendes: Damit das Dokument mit einer eingebetteten Signatur signiert werden kann, muss laut Schema das Element `Signature` als das letzte Child-Element des Root (Dokument)-Elements zugelassen sein. Andernfalls ist das Dokument laut Schema ungültig. Das Authentifizierungsobjekt und die Platzierung der Signatur werden gemäß der Signaturkonfiguration gewählt. Sie müssen sicherstellen, dass Sie Zugriff auf die Authentifizierungsinformationen haben. Für nähere Informationen wenden Sie sich an Ihren SPS-Designer.

- *Überprüfen des Authentic-XML-Dokuments:* Wenn XML-Signaturen in einem SPS aktiviert sind, wird die Signatur jedes Mal, wenn das XML-Dokument in der Authentic-Ansicht geladen wird, überprüft. Wenn die Passwort- bzw. Zertifkat-Schlüsselinformationen nicht mit dem SPS und der Signatur gespeichert werden, wird der Benutzer der Authentic-Ansicht aufgefordert, das Passwort einzugeben oder ein Zertifikat für die Überprüfung auszuwählen. Beachten Sie, dass eine eingebettete Signatur beim Speichern mit der XML-Datei gespeichert wird.Die generierte Signatur muss explizit (über die Schaltfläche **Signatur entfernen** des das Dialogfelds "XML-Signatur") entfernt werden (*siehe Abbildung oben)*, wenn sie nicht mit der XML-Datei gespeichert werden soll. Ähnlich verhält es sich auch bei einer separat gespeicherten Signatur. Wenn Sie nicht benötigt wird, muss sie explizit entfernt werden.

11.3.7 Grafiken in der Authentic-Ansicht

Sie können in der Authentic-Ansicht Bilder definieren, die im Ausgabedokument (HTML, RTF, PDF und Word 2007) verwendet werden sollen. Beachten Sie jedoch, dass einige Bildformate in manchen Formaten oder von manchen Applikationen eventuell nicht unterstützt werden. So wird das SVG-Format z.B. in PDF unterstützt, nicht aber in RTF und für die Anzeige in HTML würden Sie ein Browser-Zusatzprogramm benötigen. Stellen Sie daher bei Auswahl eines Bildformats sicher, ein Format zu wählen, das in den Ausgabeformaten Ihres Dokuments auch unterstützt wird. Die meisten Bildformate werden in allen Ausgabeformaten unterstützt (*siehe Liste unten*).

Die Authentic-Ansicht basiert auf dem Internet Explorer und kann die meisten der Grafikformate, die von Ihrer Internet Explorer-Version unterstützt werden, anzeigen. Unterstützt werden die folgenden gebräuchlichen Grafikformate:

 © 2016-2022 Altova GmbH

- GIF
- JPG
- PNG
- BMP
- WMF (Microsoft Windows Metafile)
- EMF (Enhanced Metafile)
- SVG (nur für PDF-Ausgabe)

Relative Pfade

Relative Pfade werden relativ zur SPS-Datei aufgelöst.

11.3.8 Verwendung von Tasten in der Authentic-Ansicht

Eingabetaste
In der Authentic-Ansicht wird die **Eingabetaste** an bestimmten Cursorpositionen verwendet, um zusätzliche Elemente anzuhängen. Wenn das Kapitel eines Buchs z.B. (laut Schemadefinition) mehrere Absätze enthalten kann, wird bei Drücken der **Eingabetaste** innerhalb des Texts des Absatzes ein neuer Absatz direkt hinter dem aktuellen Absatz angehängt. Wenn ein Kapitel einen Titel und mehrere Absätze enthalten kann, wird bei Drücken der **Eingabetaste** innerhalb des Kapitels aber außerhalb eines Absatzelements (auch innerhalb des Titel-Elements) ein neues Kapitel hinter dem aktuellen Kapitel angehängt (vorausgesetzt es sind laut Schemadefinition mehrere Kapitel zulässig).

Anmerkung: Bei Drücken der **Eingabetaste** wird keine neue Zeile angelegt. Dies ist auch dann der Fall, wenn sich der Cursor innerhalb eines Textnode wie z.B. eines Absatzes befindet.

Verwendung der Tastatur

Verwenden Sie die Tastatur wie bei allen Anwendungen zum Eingeben von Text/Daten und zum Navigieren. Beachten Sie dabei die folgenden Punkte:

- Mit der **Tabulatortaste** springen Sie mit dem Cursor weiter. Der Cursor hält vor und hinter Nodes an und markiert den Inhalt von Nodes; statischer Inhalt wird übersprungen.
- Die Hyperlinks `hinzufügen...` und `Node hinzufügen` werden als Node-Inhalt betrachtet und markiert, wenn Sie mit der Tabulatortaste navigieren. Durch Drücken der Leertaste oder der **Eingabetaste** werden sie aktiviert.

11.4 Authentic Skripterstellung

Durch die Funktion **Authentic-Skripterstellung** können SPS-Designs flexibler und interaktiver gestaltet werden. Diese Designs können in der StyleVision Enterprise und der Professional Edition erstellt werden und in der Authentic-Ansicht der Enterprise und Professional Edition von Altova-Produkten angezeigt werden.

In der Tabelle unten sehen Sie eine komplette Liste aller Altova-Produkte, die diese Funktion unterstützen. Beachten Sie, dass die interne Skripterstellung in der trusted Version des Authentic Browser Plug-in aus Sicherheitsgründen deaktiviert ist.

Altova-Produkt	Erstellung von Authentic Skripts	Authentic Skripts aktiv
StyleVision Enterprise	Ja	Ja
StyleVision Professional	Ja	Ja
StyleVision Standard *	Nein	Nein
XMLSpy Enterprise	Nein	Ja
XMLSpy Professional	Nein	Ja
XMLSpy Standard	Nein	Nein
AuthenticDesktop Enterprise	Nein	Ja
AuthenticDesktop Community	Nein	Nein
Authentic Browser Plug-in Community **	Nein	Nein
Authentic Browser Plug-in Enterprise Trusted ***	Nein	Ja
Authentic Browser Plug-in Enterprise Untrusted	Nein	Ja

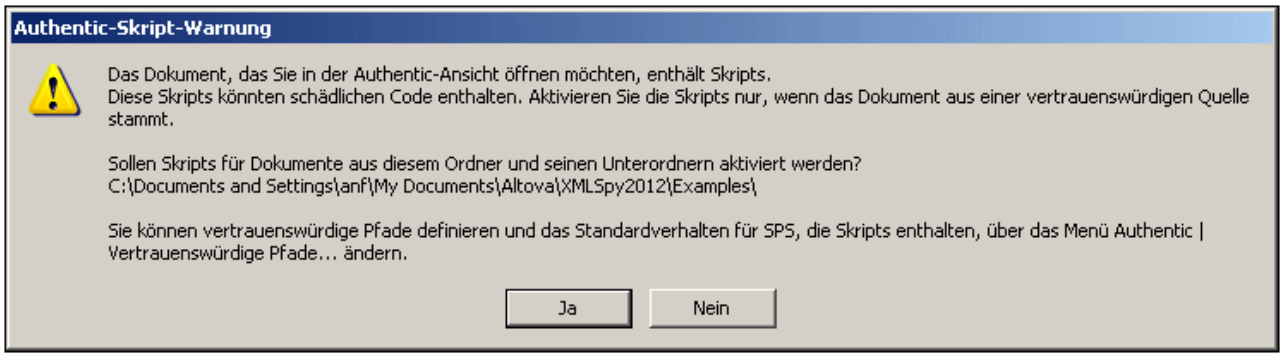
* *Keine Authentic-Ansicht*
** *Sowohl Trusted als auch Untrusted Version*
*** *Designs mit Skript werden angezeigt. Keine interne Makroausführung oder Event-Behandlung. Externe Events werden ausgelöst.*

Authentic-Skripts verhalten sich in allen Altova-Produkten gleich. Es ist kein produktspezifischer Code und keine produktspezifische Einstellung erforderlich.

Das Dialogfeld "Authentic-Skript-Warnung"

Wenn eine PXF-Datei oder eine mit einem SPS verknüpfte XML-Datei ein Skript enthält und die Datei geöffnet wird bzw. in die Authentic-Ansicht gewechselt wird, so wird ein Warndialogfeld (*Abbildung unten*) angezeigt.

© 2016-2022 Altova GmbH



Sie können eine der folgenden Optionen wählen:

- Klicken Sie auf **Ja,** um einen Ordner mit der Datei zur Liste der vertrauenswürdigen Pfade für Authentic-Skripts hinzuzufügen. In der Folge werden alle Dateien unter diesem Pfad in der Authentic-Ansicht geöffnet, ohne dass vorher dieses Warndialogfeld angezeigt wird. Sie können die Liste vertrauenswürdiger Pfade über den Menübefehl **Authentic | Vertrauenswürdige Pfade** 1438 aufrufen und ändern.
- Klicken Sie auf **Nein**, damit der Ordner mit der Datei nicht zur Liste der vertrauenswürdigen Pfade hinzugefügt wird. Die Datei wird in der Authentic-Ansicht angezeigt, wobei die Skripts deaktiviert sind. Das Dialogfeld "Authentic-Skript-Warnung" wird jedes Mal, wenn diese Datei in der Authentic-Ansicht geöffnet wird, angezeigt. Um den Ordner der Datei zu einem späteren Zeitpunkt zur Liste der vertrauenswürdigen Pfade hinzuzufügen, öffnen Sie das Dialogfeld "Vertrauenswürdige Pfade" über den Menübefehl **Authentic | Vertrauenswürdige Pfade** 1438 und fügen Sie den Ordner hinzu oder ändern Sie die Pfade nach Wunsch.

Eine Beschreibung des Dialogfelds "Vertrauenswürdige Pfade" finden Sie im Abschnitt Menübefehle unter der Beschreibung des Menübefehls **Authentic | Vertrauenswürdige Pfade** 1438.

Anmerkung: Bei Aufruf von XMLSpy über die COM-Schnittstelle (Beschreibung dazu siehe Referenz für Programmierer 1660) **wird die Sicherheitsüberprüfung nicht durchgeführt und das Dialogfeld "Authentic-Skript-Warnung" wird nicht angezeigt.**

Funktionsweise der Authentic Skripterstellung

Die Authentic-Skripterstellung kann beim Design von SPS-Dateien auf zwei Arten eingesetzt werden, um Authentic-Dokumente interaktiv zu machen:

- Durch Zuweisen von Skripts für benutzerdefinierte Aktionen (Makros) zu Design-Elementen, Symbolleisten-Schaltflächen und Kontextmenübefehlen.
- Durch Hinzufügen von Event Handlern zum Design, die auf Events der Authentic-Ansicht reagieren.

Alle Skripts, die erforderlich sind, um Authentic-Dokumente interaktiv zu machen, werden über die StyleVision-Benutzeroberfläche (Enterprise und Professional Edition) erstellt. Die Formulare, Makros und Event Handler werden mit dem Skript-Editor von StyleVision erstellt und diese Skripts werden mit dem SPS gespeichert. Anschließend werden die gespeicherten Skripts in der Design-Ansicht von StyleVision Design-Elementen, Symbolleisten-Schaltflächen und Kontextmenüs zugewiesen. Wenn ein XML-Dokument, das auf dem SPS basiert, in einem Altova-Produkt, das die Authentic-Skripterstellung unterstützt (*siehe Tabelle oben),* geöffnet wird, stehen diese zusätzlichen Funktionen im Dokument zur Verfügung.

Dokumentation zur Authentic Skripterstellung

Die Dokumentation zur Authentic Skripterstellung finden Sie in der Dokumentation zu StyleVision. Online finde Sie diese auf der Altova Website auf der Seite "Produktdokumentation".

© 2016-2022 Altova GmbH

12 HTML und CSS

XMLSpy bietet intelligente Bearbeitungsfunktionen für HTML- 668 und CSS 670 -Dokumente. Alle beiden Dokumenttypen werden in der Textansicht 144 bearbeitet und vom aktiven HTML-Dokument kann eine Vorschau in der Browser-Ansicht angezeigt werden.

Die intelligenten Bearbeitungshilfen werden separat in den Unterabschnitten dieses Abschnitts unter HTML 668 und CSS 670 beschrieben.

12.1 HTML

HTML-Dokumente können in der Textansicht bearbeitet werden. Anschließend kann die bearbeitete Seite sofort in der Browser-Ansicht angezeigt werden. Die Textansicht bietet eine Reihe nützlicher HTML-Bearbeitungsfunktionen. Diese sind in der Textansicht[144] genauer beschrieben, doch die wichtigsten Funktionen sowie die HTML-spezifischen Optionen werden unten aufgelistet.

Grad der Unterstützung

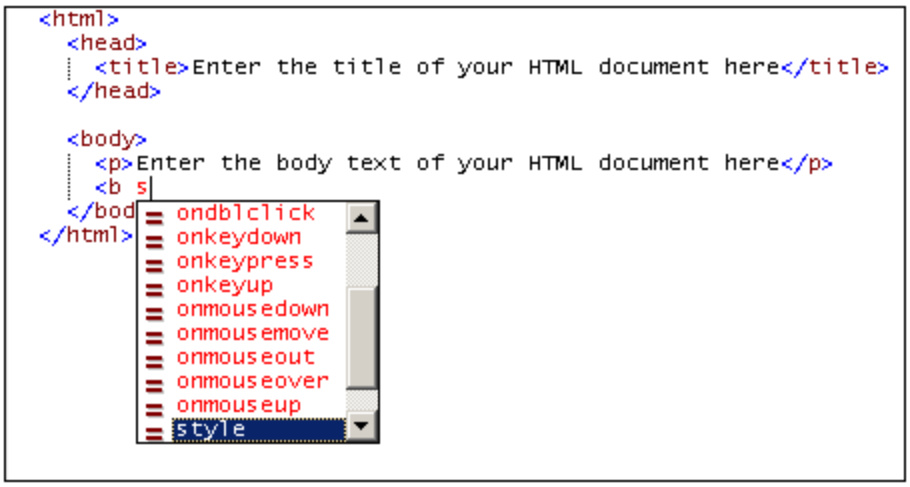
XMLSpy unterstützt HTML 4.0 und HTML 5.0. Es stehen Eingabehilfen und intelligente Bearbeitungsfunktionen für die entsprechende HTML-Version zur Verfügung. Eine Beschreibung zu diesen Funktionen finden Sie weiter unten.

Eingabehilfen

Elemente, Attribute und Entities stehen zur Verfügung, wenn ein HTML-Dokument aktiv ist. Die Eingabehilfen sind kontextsensitiv; die Einträge, die in den Eingabehilfen angezeigt werden, sind diejenigen, die an der aktuellen Cursorposition verfügbar sind. Verwenden Sie die HTML-Eingabehilfen wie in der Textansicht[165] beschrieben.

Autokomplettierung

Während der Eingabe von Markup-Text in Ihr HTML-Dokument stellt XMLSpy eine Autokomplettierungshilfe zur Verfügung. Es wird eine Popup-Liste aller Nodes angezeigt, die an der Cursorposition eingefügt werden können. Während der Eingabe wird in der Liste der passendste Vorschlag markiert (*siehe Abbildung unten*). Klicken Sie auf den ausgewählten Eintrag, um ihn an der Cursorposition einzufügen.



Die Autokomplettierung für Elemente wird angezeigt, wenn die linke Klammer des Node-Tag eingegeben wird. Bei Eingabe des Start-Tag eines Element-Node wird automatisch auch der End-Tag eingefügt. Dadurch wird die Wohlgeformtheit des Dokuments gewährleistet.

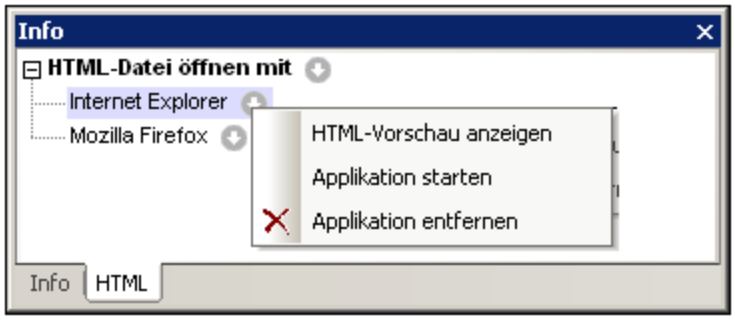
Die Autokomplettierung für Attribute erscheint, wenn nach dem Elementnamen in einem Start-Tag ein Leerzeichen eingegeben wird. Wenn Sie im Autokomplettierungs-Popup-Fenster auf einen Attributnamen klicken, wird das Attribut an der Cursorposition innerhalb von Anführungszeichen eingefügt, wobei der Cursor zwischen die Anführungszeichen gesetzt wird.

© 2016-2022 Altova GmbH

Die Eingabehilfe "Entities" enthält Zeichen-Entities aus den HTML 4.0 und HTML 5.0 Entity-Sätzen, Latin-1, Sonderzeichen und Symbole.

HTML-Info-Fenster

Im HTML-Info-Fenster (*Abbildung unten*) sind Applikationen aufgelistet, mit denen eine HTML-Datei schnell aufgerufen werden kann. Wenn eine HTML-Datei in XMLSpy aktiv ist, können Sie durch Doppelklick auf den Eintrag Mozilla Firefox eine Instanz von Mozilla Firefox starten und das aktive HTML-Dokument darin laden.



Beachten Sie die folgenden Punkte zur Verwendung:

- Über das Symbol rechts vom Eintrag *HTML-Datei öffnen mit* können Sie Applikationen zur Liste *HTML-Datei öffnen mit* hinzufügen. Alle auf dem System installierten Browser oder jede andere Applikation (wie z.B. ein Text-Editor) können über die Menübefehle, die über das Symbol *HTML-Datei öffnen mit* aufgerufen werden, hinzugefügt werden. Die damit verknüpften Applikationen sind meist Browser- oder Editor-Applikationen.
- Nachdem Sie eine Applikation zur Liste *HTML-Datei öffnen mit* hinzugefügt haben, können Sie ihren Namen in der Liste *HTML-Datei öffnen mit* ändern (außer Sie haben die Applikation mit dem Befehl *Installierte Browser hinzufügen*) hinzugefügt, indem Sie ihn auswählen, die Taste **F2** drücken und den Namen bearbeiten.
- Das Symbol rechts von den einzelnen Applikationen in der Liste *HTML-Datei öffnen mit* öffnet ein Menü mit Befehlen zum (i) Öffnen der Applikation; (ii) Öffnen der Applikation und Laden der verknüpften HTML-Datei; (iii) Entfernen der Applikation aus der Liste. Wenn Sie auf den Namen einer Applikation doppelklicken, wird die damit verknüpfte HTML-Datei in dieser Applikation geöffnet.
- Applikationen, die zur Liste *HTML-Datei öffnen mit* hinzugefügt wurden oder daraus entfernt wurden, werden auch zum CSS-Info-Fenster hinzugefügt bzw. daraus gelöscht.

Zuweisen einer DTD

Für ein XHTML-Dokument kann über das Menü DTD/Schema, in dem Sie nach der gewünschten DTD bzw. dem gewünschten XML-Schema suchen können, eine DTD oder ein XML-Schema zugewiesen werden. Ein XHTML-Dokument kann genau wie ein XML-Dokument bearbeitet werden 293.

Befehle der Browser-Ansicht

Die Befehle der Browser-Ansicht finden Sie im Menü **Browser.**

12.2 CSS

CSS-Dokumente können mit Hilfe der intelligenten Bearbeitungsfunktionen der Textansicht bearbeitet werden. Diese Funktionen werden im Zusammenhang mit der Bearbeitung von CSS-Dokumenten im Folgenden beschrieben.

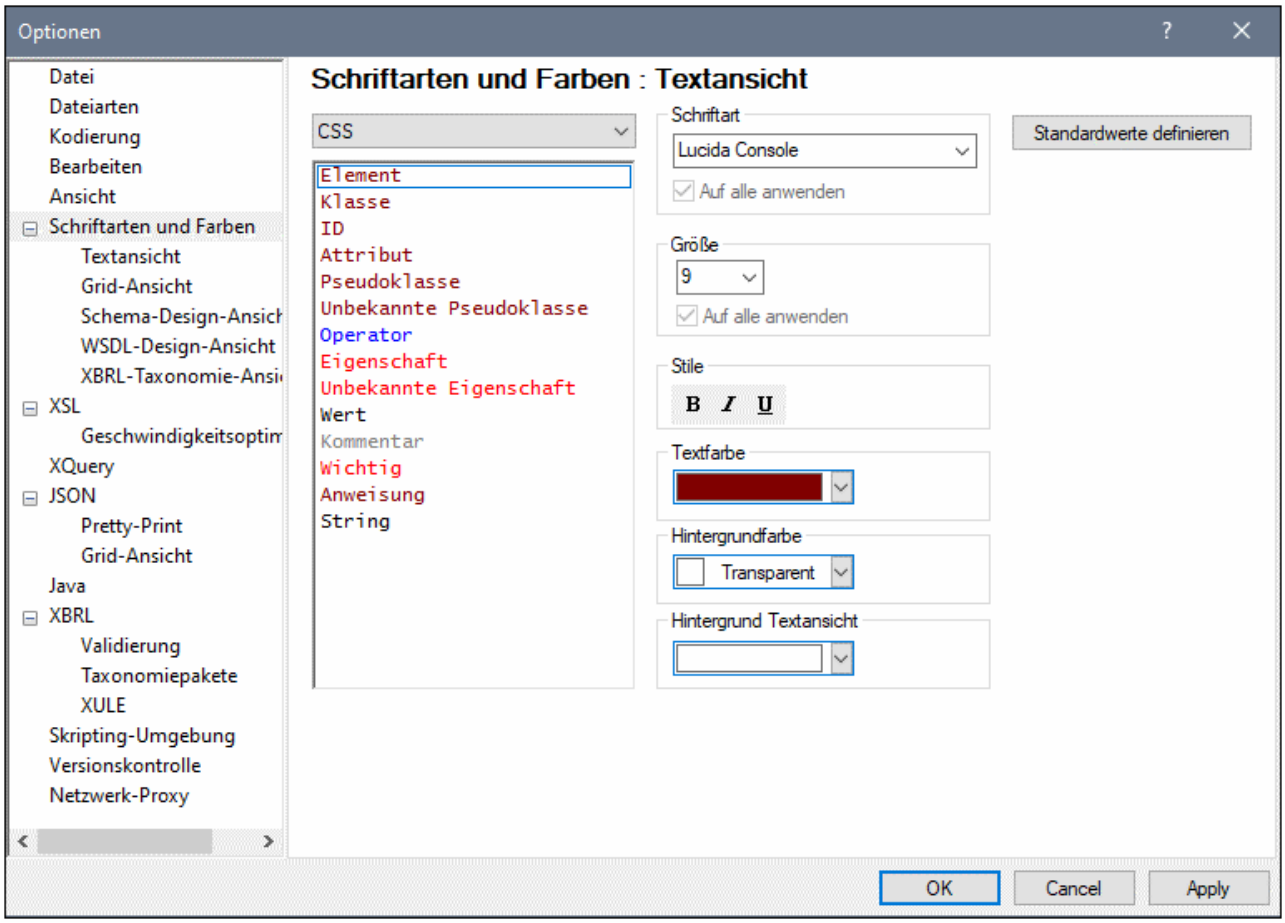
Syntaxfärbung

Eine CSS-Regel besteht aus einem Selektor, einer oder mehreren Eigenschaften und den Werten dieser Eigenschaften. Diese drei Komponenten können in noch spezifischere Kategorien unterteilt werden; so kann es sich bei einem Selektor z.B. um eine Klasse, eine Pseudo-Klasse, eine ID, ein Element oder ein Attribut handeln. Außerdem kann ein CSS-Dokument zusätzlich zu den Regeln noch weitere Elemente enthalten, wie z.B. Kommentare. In der Textansicht kann jede dieser Kategorien in einer anderen Farbe angezeigt werden (*Abbildung unten*), je nachdem, welche Einstellungen Sie im Dialogfeld "Optionen" (*siehe unten*) vorgenommen haben.

```
.header
{
    font-family: "Arial", sans-serif;
    font-weight: bold;
    color: red;
}
```

Sie können die Farben der verschiedenen CSS-Komponenten im Dialogfeld "Optionen" (*Abbildung unten*) im Abschnitt "Schriftarten" definieren. Wählen Sie in der Auswahlliste links oben die Option "CSS" aus und wählen Sie anschließend im Bereich "Stile" für die einzelnen CSS-Elemente die gewünschte Farbe aus.

© 2016-2022 Altova GmbH



Klappleisten

Mit Hilfe der Klappleiste können Sie jede CSS-Regel erweitert bzw. reduziert anzeigen. Dies wird in der Klappleiste durch ein +/−-Zeichen markiert. Die Klappleiste kann über das Dialogfeld "Einstellungen für Textansicht" 1506 ein- und ausgeblendet werden. Wenn eine Regel reduziert angezeigt wird, wird dies durch Auslassungspunkte gekennzeichnet. Wenn Sie den Mauszeiger über die Auslassungspunkte platzieren, wird der Inhalt der eingeklappten Regel in einem Popup-Fenster angezeigt. Falls der Inhalt für das Popup-Fenster zu groß ist, so wird dies durch Auslassungspunkte am unteren Rand des Popup-Fenster angezeigt.

Mit der Schaltfläche **Alle ein-/ausklappen** in der Symbolleiste "Text" werden **alle** Regeln in ihrer erweiterten Form angezeigt bzw. auf das Dokumentelement der obersten Ebene reduziert.

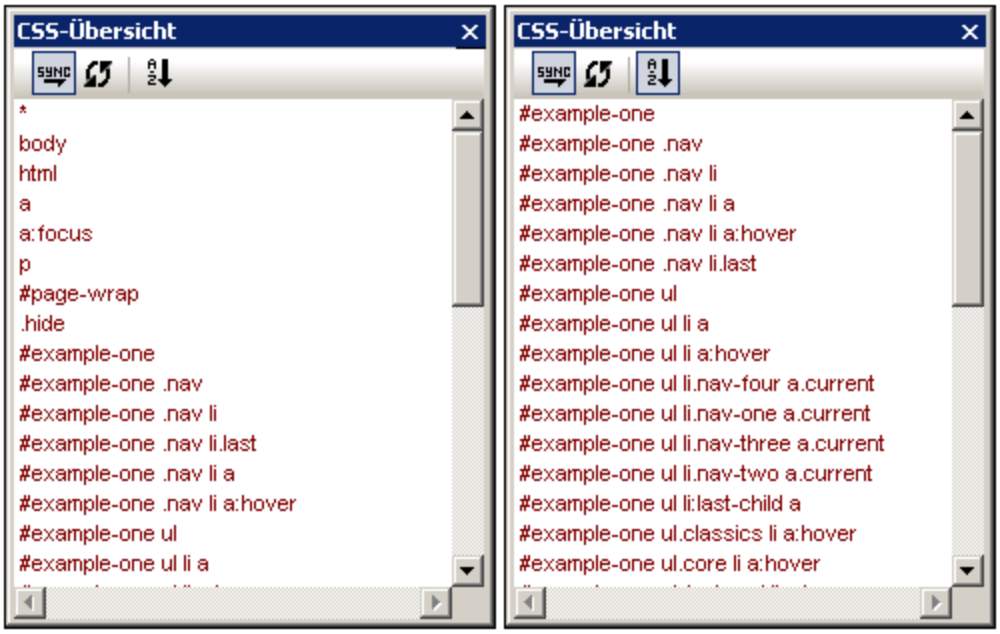
Beachten Sie außerdem, dass die geschweiften Klammern, innerhalb welcher eine Regel steht (Abbildung oben) fett angezeigt werden, wenn der Cursor entweder vor oder hinter eine der geschwungenen Klammern platziert wird. Auf diese Art sehen Sie genau, wo die Definition einer bestimmten Regel beginnt und wo sie endet.

CSS-Übersicht

Die Eingabehilfe "CSS-Übersicht" (*Abbildungen unten*) bietet einen Überblick über die Selektoren des Dokuments. Wenn Sie unter CSS-Übersicht auf einen Selektor klicken, wird er im Dokument markiert. In der linken unteren Abbildung sind die Selektoren nicht sortiert und werden in der Reihenfolge aufgelistet, in der sie

im Dokument vorkommen. In der Abbildung auf der rechten Seite wurde die Funktion zur alphabetischen Sortierung (über die Symbolleistenschaltfläche) aktiviert, und die Selektoren sind alphabetisch sortiert.

Beachten Sie bitte die folgenden Punkte: (i) Bei der Überprüfung der alphabetischen Reihenfolge der Selektoren werden alle Teile des Selektors berücksichtigt, einschließlich Punkt, Raute und Doppelpunkte; (ii) Wenn das CSS-Dokument mehrere gruppierte Selektoren enthält, die gemeinsam eine einzige Regel definieren (z.B.. `h4, h5, h6 {...}`), wird jeder Selektor in der Gruppe separat aufgelistet.



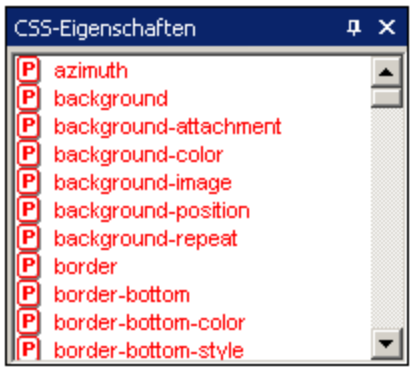
Die Symbolleisten-Schaltflächen in der Eingabehilfe "CSS-Übersicht" (von links nach rechts), haben die folgenden Funktionen:

	Schaltet die automatische Synchronisierung (mit dem Dokument) ein bzw. aus. Wenn die automatische Synchronisierung aktiviert ist, werden die Selektoren schon während der Eingabe ins Dokument in die Eingabehilfe eingegeben.
	Synchronisiert die Eingabehilfe mit dem aktuellen Zustand des Dokuments.
	Schaltet die alphabetische Sortierung ein bzw. aus. Wenn diese Sortierung deaktiviert ist, werden die Selektoren in der Reihenfolge aufgelistet, in der sie im Dokument aufscheinen. Bei alphabetischer Sortierung werden ID-Selektoren als erste angezeigt, da ihnen ein Rautensymbol vorangestellt ist (z.B. `#intro`).

Eingabehilfe "Eigenschaften"

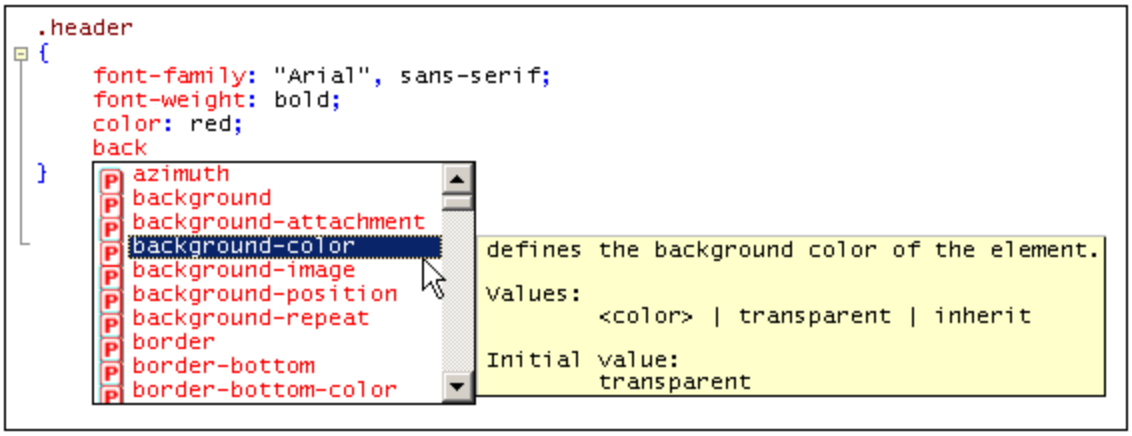
Die Eingabehilfe "Eigenschaften" (*Abbildung unten*) bietet eine Liste aller CSS-Eigenschaften in alphabetischer Reihenfolge. Eine Eigenschaft kann durch Doppelklick auf die Eigenschaft an der Cursorposition eingefügt werden.

© 2016-2022 Altova GmbH



Autokomplettierung von Eigenschaften und Tooltipps für Eigenschaften

Während Sie den Namen einer Eigenschaft eingeben, wird eine Liste von Eigenschaften eingeblendet, die mit diesen Buchstaben beginnen (*Abbildung unten*). Alternativ dazu können Sie den Cursor auch an eine beliebige Stelle innerhalb eines Eigenschaftsnamens platzieren und anschließend **Strg+Leertaste** drücken, um die Liste der CSS-Eigenschaften aufzurufen.



Sie können einen Tooltipp mit der Definition der Eigenschaft und ihren möglichen Werten anzeigen, indem Sie die Liste hinunterscrollen oder mit der Nach oben- und Nach unten-Taste Ihrer Tastatur zu der Eigenschaft navigieren. Der Tooltipp der markierten Eigenschaft wird angezeigt. Um eine Eigenschaft einzufügen, drücken Sie entweder die Eingabetaste, wenn die Eigenschaft ausgewählt ist, oder klicken Sie darauf.

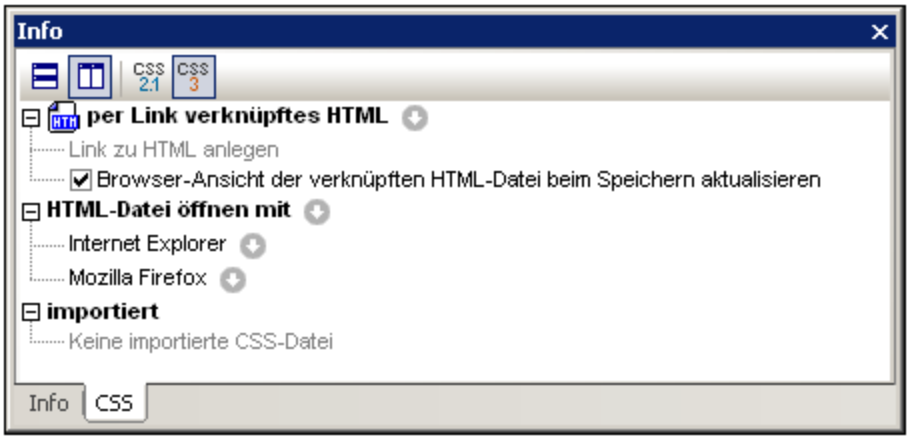
CSS-Infofenster

Wenn eine CSS-Datei aktiv ist, ist das CSS-Infofenster ebenfalls aktiv (*Abbildung unten*). Dieses Fenster bietet die folgenden Möglichkeiten:

- Über dieses Fenster können Sie zwischen CSS 2.1 und CSS 3.0 wechseln. Je nachdem, welche CSS-Version in der Symbolleiste im Fenster "Info" ausgewählt ist, werden die entsprechenden Auswahlmöglichkeiten in den Eingabehilfen und intelligenten Bearbeitungsfunktionen angezeigt.
- Über dieses Fenster können Sie die CSS-Datei mittels Link mit einer HTML-Datei verknüpfen Mit Hilfe dieser Funktion können Sie das CSS-Dokument ändern und die Auswirkung der Änderung sofort anzeigen. Außerdem kann die verknüpfte HTML-Datei über das CSS-Info-Fenster in mehreren

Browsern geöffnet werden, sodass die Änderungen im CSS-Dokument in mehreren Browsern angezeigt werden.

- Im CSS-Infofenster werden die importierten Stylesheets aufgelistet, sodass Sie einen Überblick über die Importstruktur des aktiven CSS-Stylesheet erhalten.



Beachten Sie die folgenden Punkte:

- Die Symbolleiste des Fensters "Info" enthält Symbole für CSS 2.1 und CSS 3.0. Wählen Sie die gewünschte Version aus, damit die Eingabehilfen und die intelligenten Bearbeitungsfunktionen gemäß der ausgewählten CSS-Version angepasst werden.
- Es kann nur eine HTML-Datei mit dem aktiven CSS-Dokument verknüpft werden. Klicken Sie dazu auf das Symbol rechts vom Eintrag *per Link verknüpftes HTML*, wählen Sie dann den Befehl **Link zu HTML anlegen** und navigieren Sie anschließend zur gewünschten HTML-Datei. Die verknüpfte HTML-Datei wird anschließend im Info-Fenster unter dem Eintrag *per Link verknüpftes HTML* aufgelistet (*Abbildung oben*). Durch die Erstellung dieser Verknüpfung wird das CSS-Dokument bzw. die HTML-Datei nicht geändert. Der Link dient dazu, eine HTML-Datei anzulegen, auf die das aktive CSS-Dokument zu Testzwecken angewendet werden kann.
- Wenn Sie auf die aufgelistete HML-Datei doppelklicken, wird die HTML-Datei in XMLSpy geöffnet.
- Mit Hilfe der Symbolleisten-Schaltflächen können Sie das CSS-Dokument und die HTML-Datei horizontal nebeneinander und vertikal übereinander anordnen.
- Wenn Änderungen am CSS-Dokument gespeichert werden, kann die in XMLSpy geöffnete HTML-Datei automatisch aktualisiert werden. Um diese automatischen Aktualisierungen zu aktivieren, aktivieren Sie das Kontrollkästchen *Browser-Ansicht der verknüpften HTML-Datei beim Speichern aktualisieren.* Beachten Sie, dass diese Aktualisierungen nur stattfinden, wenn die HTML-Datei eine Referenz auf das gerade bearbeitete CSS-Dokument enthält.
- Um die verknüpfte HTML-Datei zu ändern, wählen Sie mit dem Befehl **Link zu HTML anlegen** eine andere HTML-Datei aus.
- Um den Link zur HTML-Datei zu entfernen, klicken Sie auf das Symbol rechts neben dem Eintrag *Linked HTML* und wählen Sie den Befehl **Link entfernen**.
- Über das Symbol rechts neben dem Eintrag *HTML-Datei öffnen mit* können Sie Applikationen zur Liste *HTML-Datei öffnen mit* hinzufügen. Alle auf Ihrem System installierten Browser und jede andere Applikation (wie z.B. ein Texteditor) können über die Menübefehle, die Sie über dieses Symbol aufrufen können, hinzugefügt werden. Bei den damit verknüpften Applikationen handelt es sich normalerweise um Browser- oder Editor-Applikationen.
- Nachdem Sie eine Applikation zur Liste *HTML-Datei öffnen mit* hinzugefügt haben, können Sie ihren Namen in der Liste *HTML-Datei öffnen mit* ändern (außer Sie haben die Applikation mit dem Befehl *Installierte Browser hinzufügen*) hinzugefügt, indem Sie ihn auswählen, die Taste **F2** drücken und den Namen bearbeiten.

© 2016-2022 Altova GmbH

- Mit den Symbolen rechts von den einzelnen in der Liste *HTML-Datei öffnen mit* angeführten Applikationen wird ein Menü geöffnet, das Befehle zum (i) Öffnen der Applikation; (ii) Öffnen der Applikation und Laden der verknüpften HTML-Datei; (iii) Entfernen der Applikation aus der Liste enthält. Wenn Sie auf einen Applikationsnamen doppelklicken, wird die damit verknüpfte HTML-Datei in dieser Applikation geöffnet.
- Applikationen, die zur Liste *HTML-Datei öffnen mit* hinzugefügt wurden oder daraus entfernt wurden, werden auch zum HTML-Info-Fenster hinzugefügt bzw. daraus gelöscht.
- Unter dem Eintrag *Imported* wird eine Liste der von der aktiven CSS-Datei importierten CSS-Dateien angezeigt.

13 JSON, JSON-Schema

JSON (*JavaScript Object Notation*) ist ein kompaktes Format für die Speicherung und den Austausch von Daten, für das die JavaScript-Syntax verwendet wird. Wie auch das XML-Format ist JSON ein vom Menschen lesbares reines Textformat. Da JSON-Text von jeder Programmiersprache gelesen werden kann, wird es häufig als Format für den Datenaustausch im Internet verwendet.

XMLSpy bietet als IDE Unterstützung für die Bearbeitung und Validierung von JSON-Datendokumenten (Instanzdokumente)[679] und die Erstellung von syntaktisch und semantisch korrekten JSON-Schema[682]-Dokumenten.

XMLSpy bietet außerdem auch Unterstützung für Avro und Avro-Schema[779].

JSON5

JSON5 bildet eine Erweiterung zu JSON, die zusätzlich über einige ECMAScript 5-Erweiterungen verfügt (Nähere Informationen dazu siehe json5.org). JSON5 ist eine streng eingegrenzte Untermenge von JavaScript, enthält zusätzlich zu den vorhandenen JSON-Typen keine neuen Datentypen und lässt sich mit jedem bestehenden JSON-Content verwenden.

Alle XMLSpy-Funktionen, die für JSON-Instanzdokumente zur Verfügung stehen, stehen auch für JSON5-Instanzdokumente zur Verfügung. Beachten Sie jedoch die folgenden wichtigen Unterschiede zwischen JSON5 und JSON hinsichtlich der Art, wie XMLSpy die beiden Formate behandelt:

- JSON5 ist kein offizieller Nachfolger von JSON. Es wird daher eine eigene Dateierweiterung dafür verwendet: `json5`.
- XMLSpy erkennt Dateien mit der Dateierweiterung `.json` standardmäßig als JSON-Instanzdokumente und Dateien mit der Erweiterung `.json5` als JSON5-Instanzdokumente.
- JSON5-Instanzdokumente können anhand von JSON-Schemas validiert werden. JSON-Instanzdokumente, die Avro-Instanzen repräsentieren können, können sowohl anhand von JSON-Schemas (JSON-Instanzen) als auch anhand von Avro-Schemas validiert werden. *Näheres zu diesem Thema finden Sie im Abschnitt Validieren von JSON-Dokumenten*[765].

Mit dem Begriff *JSON-Instanzen* werden in dieser Dokumentation, falls nicht anders angegeben, sowohl JSON- als auch JSON5-Instanzdokumente bezeichnet. Siehe auch Abschnitt Unterschiede zwischen JSON5 und JSON[681].

JSON und JSON-Schema in XMLSpy

Beide Dokumenttypen - JSON-Instanz und JSON-Schema - sind im JSON-Format verfasst und müssen den JSON-Regeln für Wohlgeformtheit und Gültigkeit entsprechen. Beide Dokumenttypen (Instanz und Schema) haben normalerweise die Dateierweiterung `.json`. JSON-Instanzen können in der Text-[685] und der Grid-Ansicht[690] bearbeitet werden und ein JSON-Schema-Dokument kann in diesen beiden Ansichten sowie in der JSON-Schema-Ansicht[726], einem grafischen Schema-Editor, bearbeitet werden.

XMLSpy bietet für das Arbeiten mit JSON-Instanz- und JSON-Schema-Dokumenten die folgenden Funktionalitäten:

- In der Textansicht[685] Syntaxfärbung und Syntaxüberprüfung; Autokomplettierung in JSON-Schemas und in Instanzdokumenten, wenn diese mit einem Schema verknüpft sind, Klappleisten und Strukturmarkierungen. All diese Hilfsmittel vereinfachen die Bearbeitung gültiger JSON-Instanzen und

© 2016-2022 Altova GmbH

JSON-Schema-Dokumente. in der Textansicht[685] können Instanz- und Schema-Dokumente validiert werden.
- eine Strukturansicht in Tabellenform in der Grid-Ansicht[690], mit der die Dokumentenstruktur besser visualisiert werden kann. Sie können die Bearbeitungen direkt in der Grid-Ansicht[690] vornehmen und während der Bearbeitung je nach Bedarf zwischen Text-[685] und Grid-Ansicht[690] hin- und herwechseln. Auch in der Grid-Ansicht[690] können Instanz- und Schema-Dokumente validiert werden.
- Validierung von JSON-Instanzdokumenten in der Textansicht[685] und der Grid-Ansicht[690]. Die Validierung wird anhand eines im Info-Fenster[765] zugewiesenen JSON-Schemas durchgeführt.
- In der JSON-Schema-Ansicht[726] werden JSON-Schemas in einem grafischen Layout angezeigt. Dadurch können Sie zur schnellen Erstellung von JSON-Schemas (neben Texteingaben) auch Drag-and-Drop verwenden. Die Eingabehilfen in der Ansicht stellen Bearbeitungshilfen zur Verfügung. Zusätzlich dazu wird das Schema ständig auf Gültigkeit überprüft und Fehler werden markiert.

JSON-Instanzen: Öffnen vorhandener Instanzdokumente und Erstellen neuer Instanzdokumente

- Die Standardansicht (Text-[685] oder Grid-Ansicht[690]), in der JSON/JSON5-Instanzdokumente geöffnet werden sollen, kann im Abschnitt **Optionen | Dateiarten**[1606] eingestellt werden. Vorhandene JSON/JSON5-Dokumente werden in der von Ihnen ausgewählten Standardansicht geöffnet. Sie können jederzeit zwischen Text-[685] und Grid-Ansicht[690] wechseln.
- Um ein neues JSON- oder JSON5-Instanzdokument zu erstellen, klicken Sie auf **Datei | Neu** und wählen Sie `json: JavaScript Object Notation` bzw. `json5: JSON with ECMAScript 5 extensions` aus. Sie werden gefragt, ob Sie optional eine JSON- oder (für JSON, nicht aber JSON5) eineAvro[779]-Schema-Datei für die neue Instanzdatei auswählen möchten. Wenn Sie ein Schema zuweisen, erfolgt dies im Info-Fenster[765]. Das neue Instanzdokument wird, abhängig von den Einstellungen im Abschnitt **Optionen | Dateiarten**[1606], in der Text-[685] oder Grid-Ansicht[690] geöffnet.

JSON-Schemas: Öffnen vorhandener Schemas und Erstellen neuer Schemas

- Ein vorhandenes JSON-Schema-Dokument wird in der JSON-Schema-Ansicht[726] geöffnet. Sie können jederzeit in die Text-[685] oder Grid-Ansicht[690] wechseln.
- Um ein neues JSON-Schemadokument zu erstellen, klicken Sie auf **Datei | Neu** und wählen Sie `json: JSON Schema aus.` Das neue JSON-Schema-Dokument wird in der JSON-Schema-Ansicht[726] geöffnet, wobei am Anfang des Dokuments das Schlüsselwort **`$schema`** zu sehen ist. Sie können jederzeit in die Text-[685] oder Grid-Ansicht[690] wechseln.

Alle diese Ansichten (Text[685], Grid[690] und JSON-Schema[726]) sind in den Unterabschnitten dieses Abschnitts beschrieben.

In diesem Abschnitt

Dieser Abschnitt ist in die folgenden Kapitel gegliedert:

- JSON-Daten:[679] Enthält eine Erläuterung der Grundlagen von JSON-Dokumenten.
- JSON-Schema:[682] Enthält eine Beschreibung von JSON-Schemas und deren Funktionsweise.
- JSON Lines und JSON-Kommentare:[684] Enthält Informationen über zwei weitere von XMLSpy unterstützte JSON-Spezifikationen.
- JSON-Dokumente in der Textansicht:[685] Hier wird beschrieben, wie Sie mit JSON-relevanten Funktionen in der Textansicht arbeiten.
- JSON-Dokumente in der Grid-Ansicht:[690] Hier wird beschrieben, wie Sie JSON-Dokumente in der Grid-Ansicht bearbeiten.

- JSON-Schema-Ansicht: 726 Beschreibt die JSON-Schemabearbeitungsfunktionen der Ansicht und wie Sie diese zur Erstellung Ihrer JSON-Projekte verwenden.
- Validieren von JSON-Daten/-Dokumenten: 765 Hier wird beschrieben, wie Sie einem JSON-Dokument ein JSON-Schema zuweisen und JSON-Dokumente validieren.
- Einfügen von JSON-Fragmenten: 767 Hier wird beschrieben, wie Sie JSON-Textfragmente aus externen Quelldokumenten schnell in Ihr JSON-Dokument einfügen.
- JSON-Transformationen mit XSLT/XQuery: 769 Hier wird beschrieben, wie JSON-Dokumente mit Hilfe von XPath/XQuery 3.1 abgefragt werden können.
- XQuery-Ausdrücke für JSON: 772 Hier finden Sie eine allgemeine Einführung in die Verwendung von XQuery mit JSON-Dokumenten.
- Generieren eines JSON-Schemas anhand einer JSON-Instanz: 774 Hier werden die Funktionen zum Generieren eines Schemas anhand eines Instanzdokuments beschrieben.
- Generieren einer JSON-Instanz anhand eines JSON-Schemas: 777 Hier wird beschrieben, wie Sie anhand eines Schemas eine Instanz generieren.
- Konvertieren zwischen JSON und XML: 778 Hier wird beschrieben, wie Sie in XMLSpy zwischen JSON und XML konvertieren.

© 2016-2022 Altova GmbH

13.1 JSON-Daten

Dieser Abschnitt enthält eine kurze Beschreibung zur Struktur von JSON-Daten. JSON-Daten werden normalerweise in einem JSON- (Instanz-)Dokument gespeichert, können aber auch als JSON-Datenfragment in einem Dokument eines anderen Typs gespeichert sein. Ein JSON-Datenfragment oder -Dokument ist eine JSON-Datenstruktur, die einer der unten beschriebenen Datenstrukturen entspricht.

XMLSpy unterstützt zusätzlich **JSON5**, eine Erweiterung von JSON, die zusätlich einige minimale ECMScript 5-Erweiterungen enthält. Nähere Informationen dazu finden Sie unter json5.org.

JSON-Objekte und Arrays

Ein JSON-Dokument (Dateierweiterung lautet normalerweise **.json**) besteht aus den folgenden grundlegenden Datenstrukturen:

Objekt

Ein **Objekt** steht innerhalb von geschweiften Klammern. Es handelt sich hierbei um eine nicht geordnete Sammlung von null oder mehr **Schlüssel:Wert**-Paaren. Diese **Schlüssel:Wert**-Paare sind die **Eigenschaften des Objekts**. Der Schlüssel muss immer ein String sein und daher immer in Anführungszeichen stehen. Der Schlüssel (auch Name der Eigenschaft genannt) wird durch einen Doppelpunkt von seinem Wert getrennt. Ein Eigenschaftswert kann jeden beliebigen JSON-Datentyp haben (*siehe Liste unten)* 680. Eine Eigenschaft wird durch ein Komma von der nächsten getrennt. Das Codefragment unten ist ein Beispiel für ein Objekt mit drei Eigenschaften (alle mit Werten vom atomaren Typ):

```
{
  "emailtype": "home",
  "emailaddress": "contact01.home@altova.com",
  "citycode": 22
}
```

Array

Ein **Array** steht innerhalb von eckigen Klammern und ist eine kommagetrennte, geordnete Liste bestehend aus null oder mehr **Elementen**. Bei diesen Elementen kann es sich um jeden beliebigen JSON-Datentyp (*siehe Liste unten* 680) handeln.

- *Beispeil für ein Array, das zwei Objekte enthält*

  Das unten gezeigte Array besteht aus zwei Objekten (von denen jedes innerhalb von geschweiften Klammern steht). Das Array selbst steht innerhalb von eckigen Klammern.

```
[
      {
      "emailtype": "home",
      "emailaddress": "contact01.office@altova.com",
      "citycode": 22
      },
      {
      "emailtype": "office",
      "emailaddress": "contact01.office@altova.com",
```

```
        "citycode": 22
      }
    ]
```

Beispiel für Arrays, bei denen es sich um die Werte der Eigenschaften eines Objekts handelt

Im Codefragment unten sehen Sie ein Objekt mit drei `Schlüssel:Wert` -Paaren. Jeder Wert ist ein Array, das eine Tuple(sequenz) enthält. (Ein Tuple kann als eindimensionales Array betrachtet werden.) Bei den drei Elementen in den einzelnen Tuples handelt es sich um atomare Typen.

```
{
      "x": [ 1, 2, "abc" ],
      "y": [ 3, 4, "def" ],
      "z": [ 5, 6, "ghi" ]
}
```

JSON-Datentypen

Objekteigenschaftswerte und Array-Elemente können die folgenden Typen haben:

- `String` (muss innerhalb von Anführungszeichen stehen). Für einen String kann zusätzlich ein Format wie z.B. `date-time` oder `email` definiert sein.
- `Zahl`: Eine Zahl mit Nachkommastellen; dazu gehören auch Ganzzahlen
- `Ganzzahl`: Eine Zahl ohne Nachkommastellen; eine Teilmenge des Typs `Zahl`
- `Boolesch` (true/false, steht nicht innerhalb von Anführungszeichen)
- `Objekt`: Ermöglicht die Verschachtelung von Daten, wenn es innerhalb eines anderen Objekts verwendet wird.
- `Array`: Bietet die Möglichkeit, komplexere als in Objekten zulässige Strukturen zu erstellen
- `Null` (nicht in Anführungszeichen eingeschlossen)

Beispiel für JSON-Daten

Hier sehen Sie ein Beispiel für ein JSON-Datenfragment. Beachten Sie, dass das Dokument aus Objekten und Arrays besteht. Beachten Sie außerdem den Datentyp von Schlüsselwerten; String-Werte stehen innerhalb von Anführungszeichen, andere Typen werden grün angezeigt.

```
{
  "first": "Jason",
  "last": "Jones"
  "isManager": true,
  "age": 35,
  "address": {
    "street": "Jason Avenue",
    "city": "Jasonville",
    "state": "JS",
    "postcode": "JS12 ON34"
  },
  "phone": [
    {
      "type": "home",
      "number": "12 3456-7890"
```


© 2016-2022 Altova GmbH

```
    },
    {
      "type": "office",
      "number": "789 012-34567"
    }
  ],
  "children": [],
  "partner": null
}
```

Einige Unterschiede zwischen JSON5 und JSON

JSON5 ist eine streng eingegrenzte Untermenge von JavaScript, enthält keine neuen JSON-Datentypen und lässt sich mit jedem bestehenden JSON-Content verwenden. Es bestehen die folgenden Unterschiede zu JSON:

- JSON5 unterstützt Kommentare. Kommentare stehen innerhalb von: `// comment //` oder `/* comment */`.
- Die Schlüssel von `Schlüssel:Wert`-Paaren müssen in JSON5 nicht in von Anführungszeichen gesetzt werden.
- Strings können in JSON5 über mehrere Zeilen reichen.
- JSON5-Dokumente können anhand von JSON-Schemas, nicht aber anhand von Avro-Schemas validiert werden.

13.2 JSON-Schema

Genau wie in einem XML-Schema Struktur und Inhalt eines XML-Dokuments definiert sind, wird in einem JSON-Schema definiert, wie die JSON-Daten in einem JSON-Dokument strukturiert sind. Es wird definiert, welche Datenfelder erwartet werden und wie die Werte repräsentiert werden. Informationen zu JSON-Schemas und die JSON-Schema-Spezifikation finden Sie hier.

Ein JSON-Schema ist selbst ein JSON-Objekt. Lexikalisch steht das gesamte Schema innerhalb von geschweiften Klammern (*siehe Codefragment unten*). Geschweifte Klammern dienen als Trennzeichen für JSON-Objekte. Das Schema ist in JSON-Syntax geschrieben und normalerweise in einer Datei mit der Erweiterung `.json` gespeichert. Die Datei wird durch das Schlüsselwort **`$schema`**, welches das erste Schlüsselwort des Objekts der obersten Ebene sein sollte, als JSON-Schema gekennzeichnet. Dieses Schlüsselwort sollte einen der folgenden Werte haben:

- Versionen bis draft-07: **`"http://json-schema.org/draft-N/(hyper-)schema#"`**, wobei **`N`** für die Versionsnummer (**`04`**, **`06`** oder **`07`**) steht.
- Versionen ab draft-2019-09: **`"https://json-schema.org/draft/JJJJ-MM/(hyper-)schema"`**, wobei **`JJJJ`** und **`MM`** das Jahr bzw. den Monat des Entwurfs darstellen, z.B. **`2019-09`**.

Hier finden Sie ein Beispiel dafür, wie das Schlüsselwort `$schema` verwendet wird.

```
{
   "$schema": "https://json-schema.org/draft/2020-12/schema",
   ...
}
```

Anmerkung: Zwar kann das Schlüsselwort `$schema` den Wert **`"http://json-schema.org/schema#"`** haben. Dieser Wert definiert die neueste Version des Schemas. Am besten verwenden Sie eine URL zur Angabe einer bestimmten Version.. Nähere Informationen dazu finden Sie unter JSON-Schema-Version[727].

Sie können ein JSON-Schema in XMLSpy grafisch in der JSON-Schema-Ansicht erstellen. Eine Anleitung dazu finden Sie im Abschnitt JSON-Schema-Ansicht[726]. Neben den Schemabearbeitungsfunktionen in der Schema-Ansicht stehen im Zusammenhang mit Schemas die folgenden Funktionen zur Verfügung:

- Validierung mit dem JSON Validator von XMLSpy: Weisen Sie einem JSON-Instanzdokument ein JSON-Schema zu und validieren Sie das Instanzdokument in XMLSpy. Nähere Informationen dazu finden Sie unter Validieren von JSON-Dokumenten[765].
- Konfigurieren von JSON-Validierungsoptionen[1615].
- Generierung eines JSON-Schemas anhand einer JSON-Instanz[774]: Wenn bereits ein JSON-Instanzdokument vorhanden ist, können Sie anhand dieses Dokuments ein JSON-Schema generieren. Falls nötig, können Sie das Schema später bearbeiten.
- Konvertierung zwischen JSON und XML[778]: Sie können Dokumente zwischen diesen beiden Formaten konvertieren.

Terminologie

Unten finden Sie Definitionen häufig verwendeter JSON-Schema-Begriffe, die in diesem Zusammenhang auf der Benutzeroberfläche und in dieser Dokumentation vorkommen.

© 2016-2022 Altova GmbH

Begriff	Definition
Schema	Das Schema-Objekt der obersten Ebene in einem JSON-Schema-Dokument; die Schema-Datei.
Objekt	Ein JSON-Typ, der null oder mehr Eigenschaften enthält
Eigenschaft	Ein `Schlüssel:Wert`[679]-Paar eines Objekts. Der Wert dieses Objekts kann jeden beliebigen JSON-Datentyp haben.
Schlüsselwort	Der `Schlüssel`-Teil des `Schlüssel:Wert`[679]-Paars eines Objekts. Dies ist immer ein String.
Subschema	Ein Objekt, das ein Child eines Operators oder einer Abhängigkeit ist.
Definition	Die komplette Definition eines jeden JSON-Typs. Definitionen können global oder lokal[733] sein.
Array	Eine kommagetrennte geordnete Liste bestehend aus null oder mehr `Elementen` jedes beliebigen JSON-Datentyps.
Atomare Typen	Die JSON-Datentypen `String`, `Zahl`, `Ganzzahl`, `Boolesch` und `Null`.
Typ-Selektoren	Die Typen `any` und `multiple`, mit denen ein beliebiger bzw. mehrere Typen ausgewählt werden[752].
Operatoren	Occurrence-Selektoren, die als Children von Definitionen hinzugefügt werden können. *Siehe dazu Abschnitt Operatoren*[758].

JSON-Datentypen

Objekteigenschaftswerte und Array-Elemente können die folgenden Typen haben:

- `String` (muss innerhalb von Anführungszeichen stehen). Für einen String kann zusätzlich ein Format wie z.B. `date-time` oder `email` definiert sein.
- `Zahl`: Eine Zahl mit Nachkommastellen; dazu gehören auch Ganzzahlen.
- `Ganzzahl`: Eine Zahl ohne Nachkommastellen; eine Teilmenge des Typs `Zahl`
- `Boolesch` (`true/false`, steht nicht innerhalb von Anführungszeichen)
- `Objekt`: Ermöglicht die Verschachtelung von Daten, wenn es innerhalb eines anderen Objekts verwendet wird.
- `Array`: Bietet die Möglichkeit, komplexere als in Objekten zulässige Strukturen zu erstellen
- `Null` (nicht in Anführungszeichen eingeschlossen)

13.3 JSON Lines und JSON-Kommentare

XMLSpy unterstützt JSON Lines (JSONL)- und JSON with Comments (JSONC = JSON-Dokumente mit Kommentaren)-Dokumente. Das bedeutet, dass die Validierungsfunktionen und die intelligenten Bearbeitungsfunktionen für diese Dokumente im gleichen Ausmaß wie für andere Arten von JSON-Dokumenten zur Verfügung stehen. In diesem Abschnitt werden die wichtigsten Funktionen dieser JSON-Dokumenttypen behandelt.

JSON Lines

JSON Lines (JSONL) ist ein Format zum Speichern von strukturierten Daten, bei dem jeder Datensatz durch eine Zeilenschaltung vom nächsten getrennt ist, d.h. jeder Datensatz befindet sich in einer eigenen Zeile. In Folge dessen kann jeder Datensatz einzeln verarbeitet werden, wodurch sich das Format sehr gut zum Verarbeiten von Daten wie Log-Dateien eignet.

Beispiel für ein JSON Lines-Dokument

```
["Team", "Played", "Won", "Drew", "Lost", "Points"]
["USA", 2, 1, 1, 0, 4]
["France", 3, 1, 1, 1, 4]
["Germany", 1, 0, 1, 0, 1]
["USA", 1, 0, 0, 1, 0]
```

Weitere Beispiele finden Sie unter http://jsonlines.org/examples/.

JSON Lines-Dateien werden in XMLSpy anhand der Dateierweiterung `.jsonl` erkannt.

JSON with Comments

Mit Ausnahme von JSON5-Dokumenten sind in JSON-Dokumenten keine Kommentare zulässig. Das Format "JSON with Comments" (JSONC) wurde eingeführt, um Kommentare in JSON-Dokumenten zu ermöglichen. Die folgenden Kommentare werden in JSONC verwendet und in XMLSpy unterstützt:

- Einzeilige Kommentare: Erhalten `//` vorangestellt, z.B: `// Mein Kommentar`
- Mehrzeilige Kommentare: Getrennt durch `/*` und `*/`. z.B: `/* Mein Kommentar */`

JSONC-Dateien werden in XMLSpy anhand der Dateierweiterung `.jsonc` erkannt.

Anmerkung: Auch in JSON5-Dokumenten sind Kommentare zulässig. Kommentare in anderen Arten von JSON-Dateien (mit Ausnahme von JSONC und JSON5) können bei der Verarbeitung zu Fehlern führen.

© 2016-2022 Altova GmbH

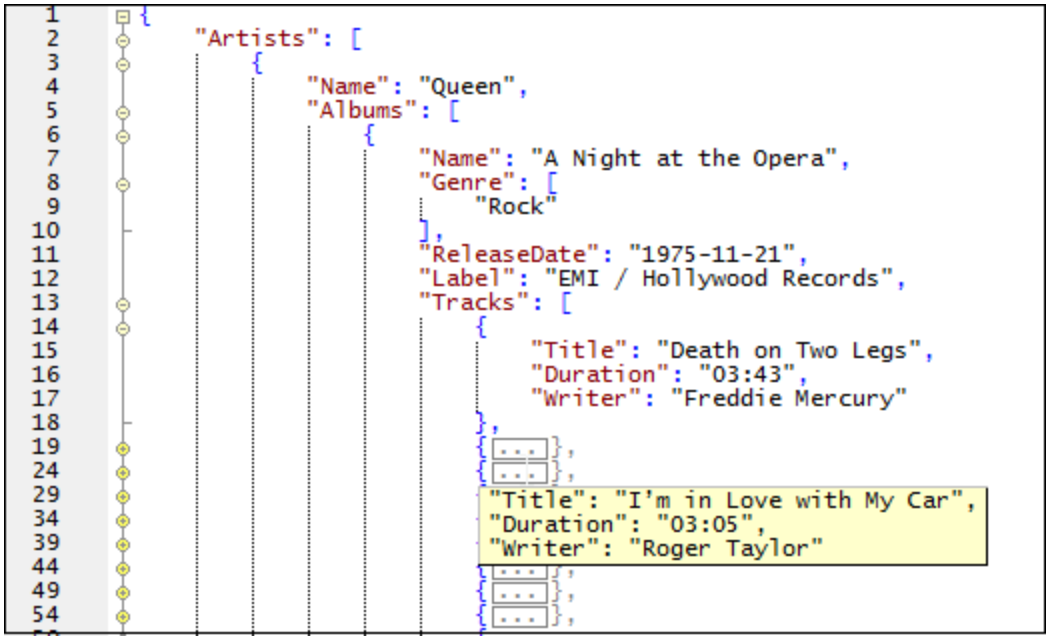
13.4 JSON-Textansicht

Altova Website: JSON Editor

JSON-Schemas, Avro-Schemas[782] und JSON/JSON5-Instanzdokumente (einschließlich Avro-Dateninstanzen im JSON-Format) können mit Hilfe der intelligenten Bearbeitungsfunktionen der Textansicht bearbeitet werden. Zu diesen Funktionen gehören Klappleisten[685], Strukturmarkierungen[686], Syntaxfärbung[686], Syntaxprüfung[687], das Speichern von Base64-kodierten Bildstrings in ihrem Bildformat[689] und Autokomplettierung[688]. XMLSpy ermöglicht auch die bidirektionale Konvertierung zwischen JSON/JSON5 und XML[778] sowie die Generierung eins JSON-Schemas anhand einer JSON/JSON5-Instanz[774].

Klappleisten

Klappleisten sind bei JSON Keywords und -Definitionen aktiv. Mit Hilfe der Klappleiste können Sie diese Nodes erweitert bzw. reduziert anzeigen. Dies wird in der Klappleiste durch ein +/--Zeichen markiert (*siehe Abbildung unten*). Die Klappleiste kann über das Dialogfeld "Einstellungen für Textansicht"[1506] ein- und ausgeblendet werden. Ein eingeklappter Node wird durch Auslassungszeichen gekennzeichnet (*siehe Abbildung unten*). Wenn Sie den Mauszeiger über die Auslassungspunkte platzieren, wird der Inhalt des eingeklappten Node in einem Popup-Fenster angezeigt (*in der Abbildung unten blau markiert*). Falls der Inhalt für das Popup-Fenster zu groß ist, so wird dies durch Auslassungspunkte am unteren Rand des Popup-Fenster angezeigt.



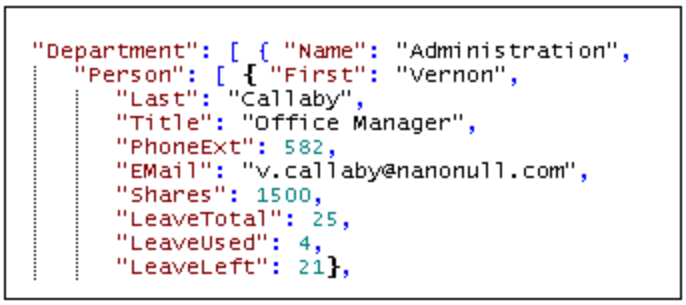
Mit der Schaltfläche **Alle ein-/ausklappen** in der Symbolleiste "Text" werden **alle** Nodes in ihrer erweiterten Form angezeigt bzw. auf das Dokumentelement der obersten Ebene reduziert.
Wenn Sie auf das +/--Symbol eines Node klicken, stehen die folgenden Optionen zur Verfügung:

Klick [–]	Klappt den Node ein.
Klick [+]	Zeigt den Node ausgeklappt an, sodass seine untergeordneten Nodes, je nachdem, in welchem Zustand sie sich vor dem Einklappen befunden haben, im erweiterten bzw.

	reduzierten Zustand zu sehen sind.
Umschalt+Klick [-]	Reduziert alle untergeordneten Nodes, belässt jedoch den angeklickten Node in seiner erweiterten Form.
Strg+Klick [+]	Erweitert den angeklickten Node sowie alle seine untergeordneten Nodes.

Strukturmarkierungen

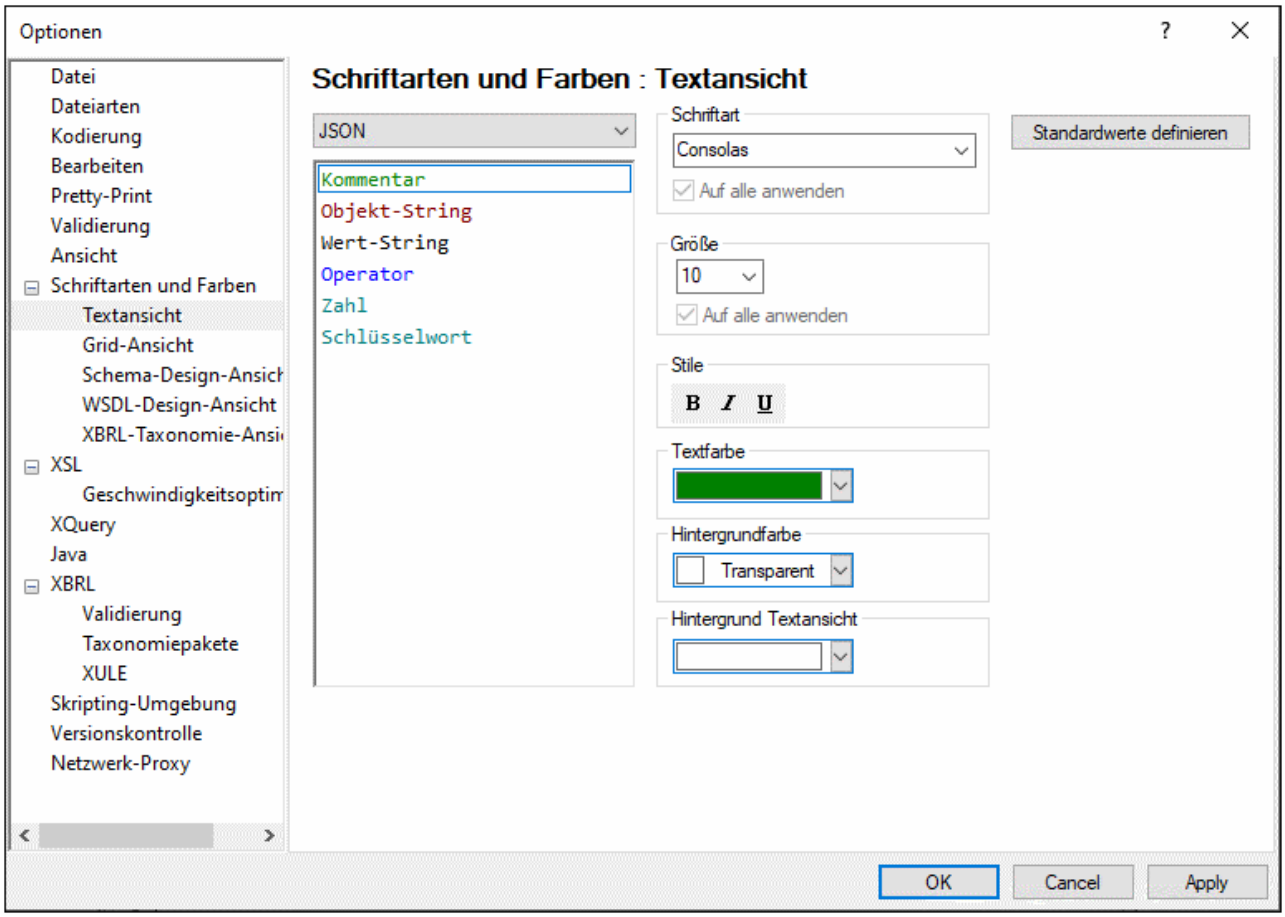
Die zwei geschweiften Klammern bzw. die eckigen Klammern, in die ein JSON-Objekt bzw. Array eingeschlossen ist (*siehe Abbildung unten*) werden fett angezeigt, wenn der Cursor entweder vor oder hinter eine dieser Klammern platziert wird. Dies zeigt an, wo die Definition eines bestimmten Elements beginnt und endet.



Syntaxfärbung

Ein JSON-Dokument (JSON- oder Avro-[779] Instanz/Schema) sowie ein JSON5-Dokument besteht aus Objektstrings, Wertestrings, Operatoren, Zahlen und Schlüsselwörtern. In der Textansicht kann jede Elementkategorie gemäß den im Dialogfeld "Optionen"[1602] (*Abbildung unten*) vorgenommenen Einstellungen in einer anderen Farbe angezeigt (*siehe Abbildung oben*) werden. Sie können die Farben der verschiedenen JSON-Komponenten im Dialogfeld "Optionen" im Abschnitt "Schriftarten" (*Abbildung unten*) definieren. Wählen Sie in der Auswahlliste oben links den Eintrag *JSON* aus und anschließend (im Fenster "Stile") für jedes JSON-Element die gewünschte Farbe.

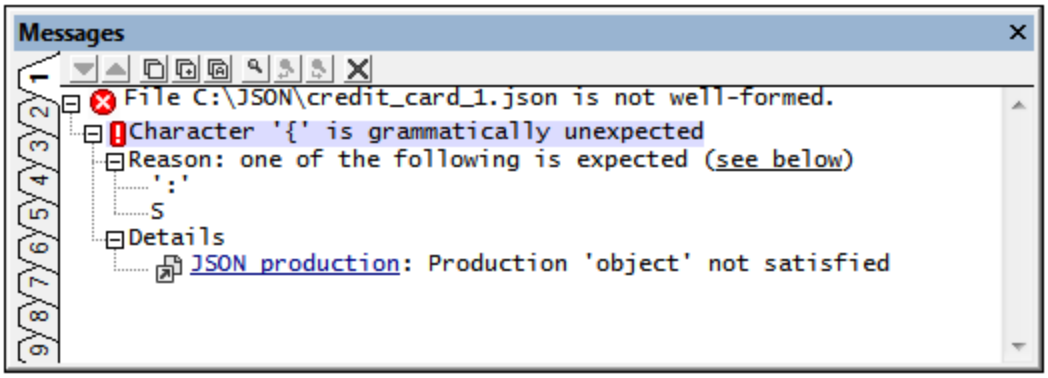
© 2016-2022 Altova GmbH



Anmerkung: In der JSON5-Syntax - nicht aber der JSON-Syntax - sind Kommentare möglich. Kommentare in JSON5 stehen innerhalb von: `// comment //` oder `/* comment */`.

Syntaxprüfung

Die Syntax eines JSON-Dokuments (JSON- oder Avro-[779] Instanz/Schema) kann durch Auswahl des Befehls **XML | Wohlgeformtheit prüfen (F7)** überprüft werden. Das Ergebnis dieser Überprüfung wird im Fenster "Meldungen" angezeigt (*Abbildung unten*).



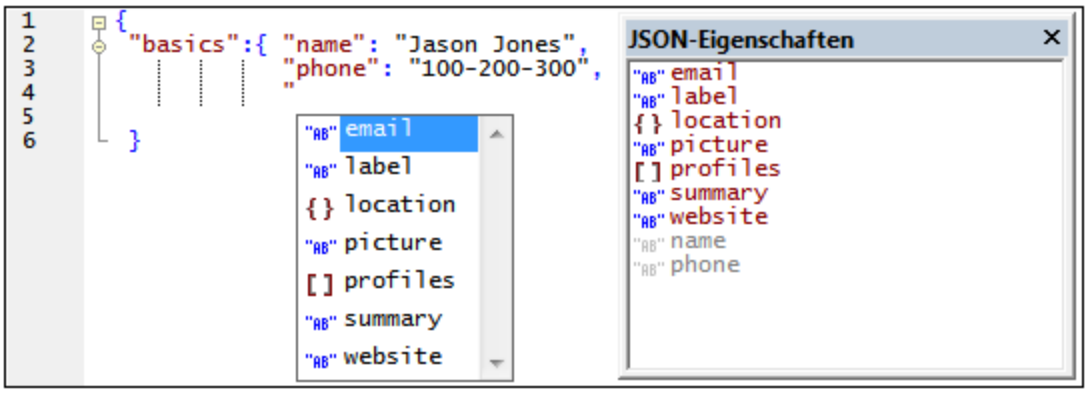
Die Fehlermeldung in der Abbildung oben verweist auf einen Fehler im Dokument: An der Stelle, an der ein Doppelpunkt erwartet wird, steht eine öffnende geschweifte Klammer.

Autokomplettierung

Die Autokomplettierung ist aktiv, wenn das bearbeitete JSON-Dokument (JSON-Instanz/Schema oder Avro-Schema) mit einem Schema verknüpft ist.

- Wenn es sich beim Dokument um ein JSON-Schema handelt, basiert die Autokomplettierung auf der durch das Schlüsselwort `$schema` 682 angegebenen Schemaversion. Nähere Informationen dazu finden Sie unter JSON-Schema-Version 727.
- Wenn es sich beim Dokument um eine JSON/JSON5-Instanz handelt, so muss der Instanz ein JSON-Schema zugewiesen sein 765, damit die Autokomplettierung aktiv ist.
- Wenn es sich beim Dokument um ein Avro-Datendokument im JSON-Format handelt, so muss der Instanz ein Avro-Schema zugewiesen werden 765, damit die Autokomplettierung funktioniert.
- Wenn es sich beim Dokument um ein Avro-Schema 782 handelt, so wird es automatisch mit dem Schema für Avro-Schema verknüpft und die Autokomplettierung erfolgt auf Basis dieses Schemas.

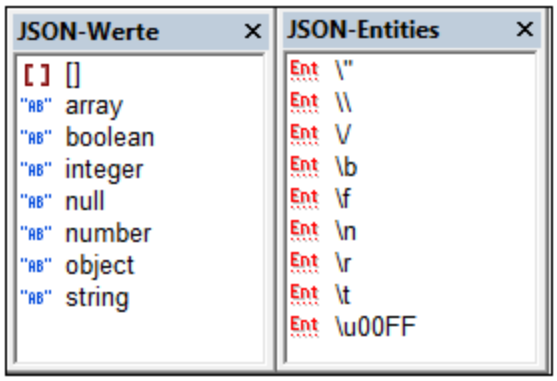
Bei der Autokomplettierung werden Ihnen an der Cursorposition verfügbare Eingabeoptionen vorgeschlagen. Die Vorschläge erfolgen (i) über Pop-ups im Hauptfenster und (ii) die Eingabehilfen (*siehe Abbildung unten*). In den Pop-ups und Eingabehilfen wird jeweils eine Liste von Einträgen angezeigt, die an dieser Cursorposition gültig sind. Mit Hilfe der Pfeiltasten können Sie durch die Einträge in der Pop-up-Liste navigieren. Wenn das Schema (im Schlüsselwort `description` des Schemas) eine Beschreibung zum Eintrag enthält, so wird die Beschreibung neben dem markierten Popup-Eintrag angezeigt. Wählen Sie einen Eintrag aus dem Pop-up-Fenster aus oder doppelklicken Sie in der Eingabehilfe auf einen Eintrag, um diesen einzufügen.



In dem in der Abbildung oben gezeigten Instanzdokument werden das Pop-up-Fenster und die Eingabehilfe "JSON-Eigenschaften" angezeigt, wenn sich der Cursor hinter den Anführungszeichen, die den Anfang eines Eigenschaftsnamens kennzeichnen, befindet. In der Eingabehilfe werden alle Eigenschaften angezeigt, die an dieser Stelle zulässig sind; bereits eingegebene Eigenschaften werden grau und deaktiviert angezeigt. Im Pop-up-Fenster werden nur die Eigenschaften angezeigt, die an dieser Stelle zulässig sind.

Es gibt zwei weitere Eingabehilfen: JSON-Werte und JSON-Entities (*Abbildung unten*). In diesen Eingabehilfen sehen Sie die zulässigen Werte von `Schlüssel:Wert`-Paaren bzw. Entities für Escape-Zeichen in JSON-Strings. In der Eingabehilfe "JSON-Werte" in der Abbildung unten sehen Sie die Werte, die bei der Bearbeitung eines JSON-Schemas für das Schlüsselwort `type` zulässig sind. Der letzte Eintrag in der Eingabehilfe "JSON-Entities" , `\uOOFF`, ist ein Platzhalter, der für ein Unicode-Zeichen steht. Ersetzen Sie den blau markierten Teil durch den Code des gewünschten Unicode-Zeichens.

© 2016-2022 Altova GmbH



Des Weiteren stehen die folgenden kontextsensitiven Autokomplettierungseinträge oder Tipps zur Verfügung, falls diese im Schema definiert wurden: Enumerationen, Beschreibungen, obligatorische Einträge (Occurrences) und Standardwerte.

Speichern eines Base64-kodierten String als Bild

Um einen Base64-kodierten String in seinem Bildformat zu speichern, klicken Sie mit der rechten Maustaste auf das Bild oder dessen Zelle und wählen Sie den Befehl **Als Bild speichern**. Wählen Sie im daraufhin angezeigten Dialogfeld den Pfad, unter dem Sie das Bild speichern möchten und geben Sie einen Namen für die Bilddatei ein. Die Erweiterung der Bilddatei (`.png`, `.gif`, `.svg`, usw.) wird automatisch anhand der Base64-Kodierung ermittelt und im Speichern-Dialogfeld angezeigt. Klicken Sie anschließend auf **Speichern**.

Diese Aktion kann auch über den Menübefehl **Bearbeiten | Als Bild speichern** durchgeführt werden.

13.5 JSON-Grid-Ansicht

In der JSON-Grid-Ansicht sehen Sie die Struktur des JSON-Dokuments (JSON-Instanz/Schema oder Avro-Schema[782]) in einem Raster. Hier können Sie das Dokument in einem Raster leichter bearbeiten und Blöcke aus strukturiertem Text neu strukturieren. Dadurch behalten Sie vor allem bei langen und kompliziert strukturierten Dokumenten die Übersicht besser als in der Textansicht und haben zusätzliche Bearbeitungsfunktionen zur Verfügung. JSON-Dokumente sind oft aufgrund von Arrays und Objekten, die wiederum auf mehreren Ebenen innerhalb von Arrays und Objekten verschachtelt sind, komplex. Betrachten Sie zum Vergleich das JSON-Codefragment, so wie es in der Textansicht angezeigt würde, und dessen Darstellung in der Grid-Ansicht (wie in der Abbildung weiter unten gezeigt).

Anmerkung: **Avro-Unterstützung** ist nur in der **Enterprise Edition verfügbar**.

JSON-Codefragment in der Textansicht

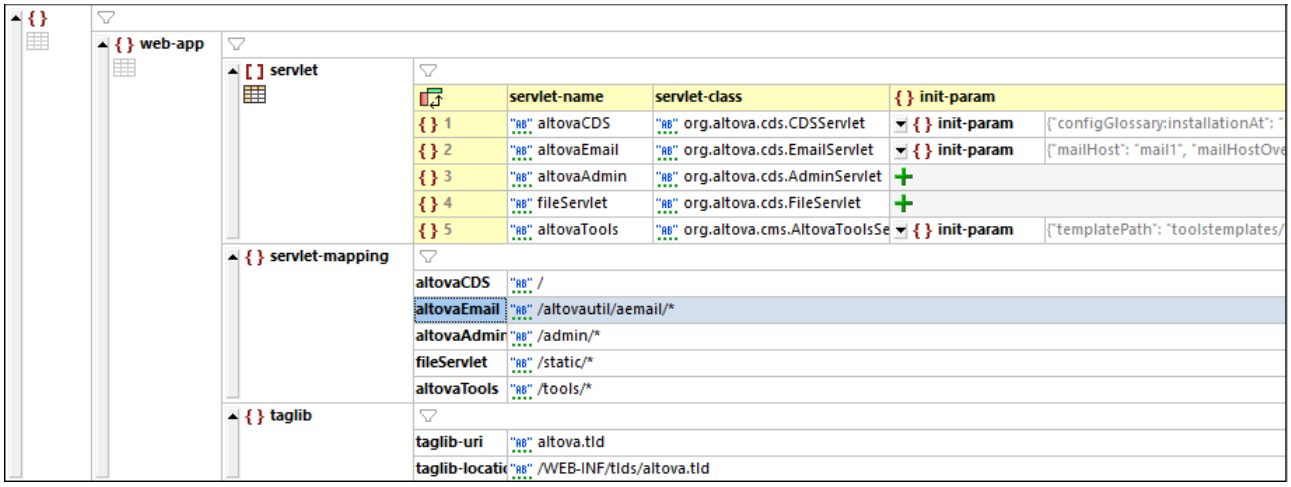
```
{
  "web-app": {
  "servlet": [
    {
        "servlet-name": "altovaCDS",
        "servlet-class": "org.altova.cds.CDSServlet",
        "init-param": {
        "configGlossary:installationAt": "Philadelphia, PA",
        "configGlossary:adminEmail": "ksm@pobox.com",
        "configGlossary:poweredBy": "Altova",
        "configGlossary:poweredByIcon": "/images/altova.gif",
        "configGlossary:staticPath": "/content/static",
        "templateProcessorClass": "org.altova.WysiwygTemplate",
        "templateLoaderClass": "org.altova.FilesTemplateLoader",
        "templatePath": "templates",
        "templateOverridePath": "",
        "defaultListTemplate": "listTemplate.htm",
        "defaultFileTemplate": "articleTemplate.htm",
        "useJSP": false,
        "jspListTemplate": "listTemplate.jsp",
        "jspFileTemplate": "articleTemplate.jsp",
        "cachePackageTagsTrack": 200,
        "cachePackageTagsStore": 200,
        "cachePackageTagsRefresh": 60,
        "cacheTemplatesTrack": 100,
        "cacheTemplatesStore": 50,
        "cacheTemplatesRefresh": 15,
        "cachePagesTrack": 200,
        "cachePagesStore": 100,
        "cachePagesRefresh": 10,
        "cachePagesDirtyRead": 10,
        "searchEngineListTemplate": "forSearchEnginesList.htm",
        "searchEngineFileTemplate": "forSearchEngines.htm",
        "searchEngineRobotsDb": "WEB-INF/robots.db",
        "useDataStore": true,
        "dataStoreClass": "org.altova.SqlDataStore",
        "redirectionClass": "org.altova.SqlRedirection",
```


© 2016-2022 Altova GmbH

```
      "dataStoreName": "altova",
      "dataStoreDriver": "com.microsoft.jdbc.sqlserver.SQLServerDriver",
      "dataStoreUrl": "jdbc:microsoft:sqlserver://LOCALHOST:1433;DatabaseName=goon",
      "dataStoreUser": "sa",
      "dataStorePassword": "dataStoreTestQuery",
      "dataStoreTestQuery": "SET NOCOUNT ON;select test='test';",
      "dataStoreLogFile": "/usr/local/tomcat/logs/datastore.log",
      "dataStoreInitConns": 10,
      "dataStoreMaxConns": 100,
      "dataStoreConnUsageLimit": 100,
      "dataStoreLogLevel": "debug",
      "maxUrlLength": 500
    }
    }, {
      "servlet-name": "altovaEmail",
      "servlet-class": "org.altova.cds.EmailServlet",
      "init-param": {
        "mailHost": "mail1",
        "mailHostOverride": "mail2"
      }
    }, {
      "servlet-name": "altovaAdmin",
      "servlet-class": "org.altova.cds.AdminServlet"
    }, {
      "servlet-name": "fileServlet",
      "servlet-class": "org.altova.cds.FileServlet"
    }, {
      "servlet-name": "altovaTools",
      "servlet-class": "org.altova.cms.AltovaToolsServlet",
      "init-param": {
        "templatePath": "toolstemplates/",
        "log": 1,
        "logLocation": "/usr/local/tomcat/logs/AltovaTools.log",
        "logMaxSize": "",
        "dataLog": 1,
        "dataLogLocation": "/usr/local/tomcat/logs/dataLog.log",
        "dataLogMaxSize": "",
        "removePageCache": "/content/admin/remove?cache=pages&id=",
        "removeTemplateCache": "/content/admin/remove?cache=templates&id=",
        "fileTransferFolder": "/usr/local/tomcat/webapps/content/fileTransferFolder",
        "lookInContext": 1,
        "adminGroupID": 4,
        "betaServer": true
      }
    }
  ],
  "servlet-mapping": {
    "altovaCDS": "/",
    "altovaEmail": "/altovautil/aemail/*",
    "altovaAdmin": "/admin/*",
    "fileServlet": "/static/*",
    "altovaTools": "/tools/*"
```

```
      },
      "taglib": {
        "taglib-uri": "altova.tld",
        "taglib-location": "/WEB-INF/tlds/altova.tld"
      }
    }
  }
```

Während die Dokumentstruktur in der Textansicht (*siehe Codefragment oben*) nicht sehr übersichtlich ist und erst genauer studiert werden muss, kann man die Struktur in der Grid-Ansicht (*Abbildung unten*) sofort auf einen Blick erfassen.



So können Sie hier die Struktur durch Hinzufügen, Löschen oder Verschieben von Objekten im Raster ganz einfach verändern. So können Sie hier z.B. ganze Textblöcke (z.B. durch Sortieren oder Verschieben) neu strukturieren. Auch der Inhalt kann in der Grid-Ansicht einfach bearbeitet werden. Noch einfacher wird dies dank zelleninterner Befehle in einzelnen Zellen[698].

Symbolleiste der Grid-Ansicht

Befehle im Zusammenhang mit der Grid-Ansicht können schnell über die Grid-Ansichts-Symbolleiste *(Abbildung unten)* aufgerufen werden. Diese Symbolleiste befindet sich im Symbolleistenbereich am oberen Rand des Applikationsfensters. Wenn Sie die Maus über eine Symbolleisten-Schaltfläche platzieren, sehen Sie deren Namen und das dazugehörige Tastaturkürzel.



Anmerkung: Diese Befehle stehen auch im Menü "JSON"[1367] zur Verfügung.

© 2016-2022 Altova GmbH

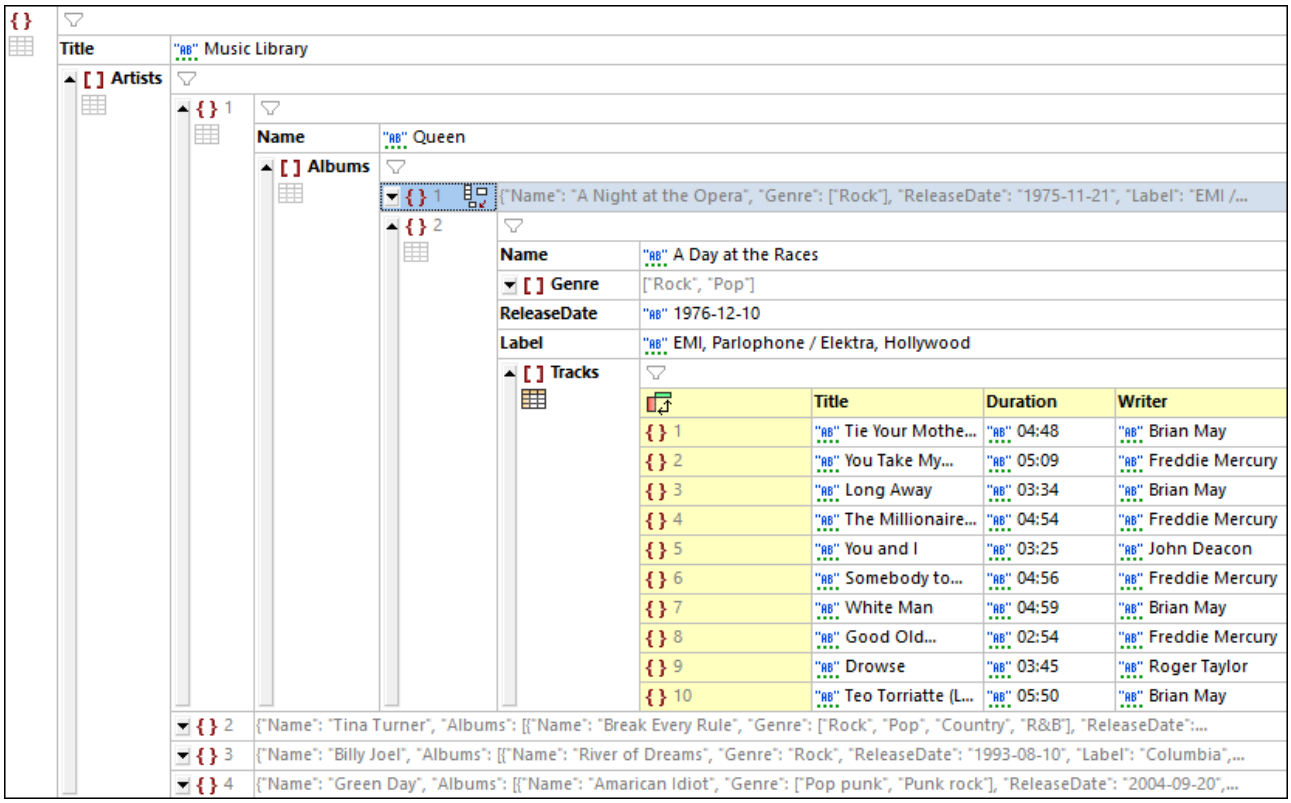
In diesem Abschnitt

Dieser Abschnitt ist in die unten aufgelisteten Kapitel gegliedert. Darin werden Funktionalitäten beschrieben, die spezifisch für die JSON-Grid-Ansicht sind.

- Dokumentanzeige in der Grid-Ansicht 693
- Bearbeiten der Dokumentstruktur 697
- Bearbeiten des Dokumentinhalts 699
- Drag-and-Drop in der Grid-Ansicht 702
- Tabellenansicht 704
- XQuery-Filter 709
- XQuery-Formeln 710
- Bildanzeige in der Grid-Ansicht 715
- Diagramme in der Grid-Ansicht 716
- Kontextmenü in der Grid-Ansicht 718
- Einstellungen der Grid-Ansicht 721

13.5.1 Dokumentanzeige in der Grid-Ansicht

In der JSON-Grid-Ansicht wird das JSON-Dokument hierarchisch innerhalb eines Rasters (*siehe Abbildung unten*) angezeigt.



In jeder Rasterzeile wird eine der folgenden Datenstrukturen angezeigt. In der nachstehenden Tabelle sehen Sie eine Liste der einzelnen Datenstrukturen mit dem jeweiligen Symbol dazu. Beachten Sie, dass sowohl Arrays als auch Objekte Child-Komponenten enthalten können, die Objekte, Arrays oder atomare Werte sein können.

{}	Ein Objekt *(siehe Definition[679])*. Objekte enthalten `Schlüssel:Wert`-Paare.
[]	Ein Array *(siehe Definition[679])*. Jedes Array enthält Elemente, bei denen es sich normalerweise um Objekte oder Werte handelt.
	ein `Schlüssel:Wert`-Paar *(siehe Definition[679])*.

In der Abbildung oben sehen wir ein Root-Objekt, das zwei `Schlüssel:Wert`-Paare enthält: (i) **`"Title":"Music Library"`**, (ii) **`"Artists":[Array]`**. Das mit **`Artists`** verbundene Array enthält vier Elemente, die Objekte sind.

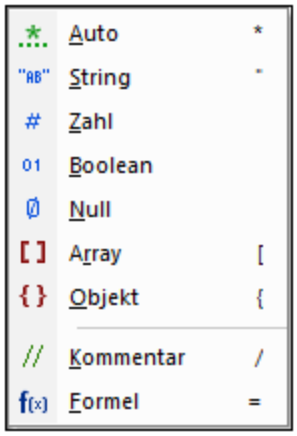
Beachten Sie die folgenden Punkte:

- Sie können mit Hilfe von **Strg + Mausrad** oder **Strg+[Plus]** und **Strg+[Minus]** in das Raster hinein- bzw. daraus hinauszoomen.
- Objekte, Arrays und Array-Elemente, die sich auf derselben Ebene befinden, sind beginnend mit 1 nummeriert. So sind z.B. in der Abbildung oben die Objekte im Array `Artists` von `1` bis `4` nummeriert. Dasselbe gilt für die Objekte in den Arrays `Albums` und `Tracks`. Beachten Sie, dass diese Nummerierung nicht im Dokument selbst enthalten ist, sondern dass dies eine Funktion der JSON-Grid-Ansicht ist, damit die Struktur des Dokuments übersichtlicher wird.
- Objekte und Arrays können durch Klick auf den Pfeil links vom Symbol für die Struktur (*siehe Abbildung oben*) erweitert und reduziert werden. Wenn eine Datenstruktur reduziert wurde, wird ihr Inhalt als einzeiliger Text angezeigt, siehe die letzten drei Rasterzeilen in der Abbildung oben.
- Wenn Sie mehrere Komponenten auf derselben Ebene auswählen, können Sie alle davon durch Drücken der **Umschalttaste** und Klick auf eine die Pfeilspitze einer der ausgewählten Komponenten erweitern/reduzieren.
- Wenn Arrays und Objekte erweitert angezeigt werden, beachten Sie die folgenden Aspekte: (i) Der Inhalt von Arrays und Objekten beginnt in der nächsten Zeile nach dessen Symbol; (iii) der Name eines Array (sein Schlüssel) befindet sich in derselben Zeile wie das Array-Symbol, während die Elemente des Array in der nächsten Zeile beginnen. Wenn ein Array oder Objekt jedoch reduziert angezeigt wird, wird sein Inhalt in derselben Zeile angezeigt.
- Beachten Sie, dass `Schlüssel:Wert`-Paare in einer einzigen Zeile angezeigt werden, außer sie sind Teil der Tabellenansicht[704] (in diesem Fall bildet der Schlüssel die Spaltenüberschrift). Ein Beispiel für eine Tabellenansicht[704] sehen Sie in der Abbildung oben im Inhalt des Array **`Tracks`**.

Symbole von Datentypen (oder Typen)

Objekteigenschaftswerte und Array-Elemente können verschiedene Typen[680] haben; die einzelnen Typen werden durch ein Symbol gekennzeichnet (siehe Abbildung unten). (Beachten Sie, dass der Typ des Werts von allen `Schlüssel:Wert`-Paaren in der Abbildung oben der Typ `string` ist.)

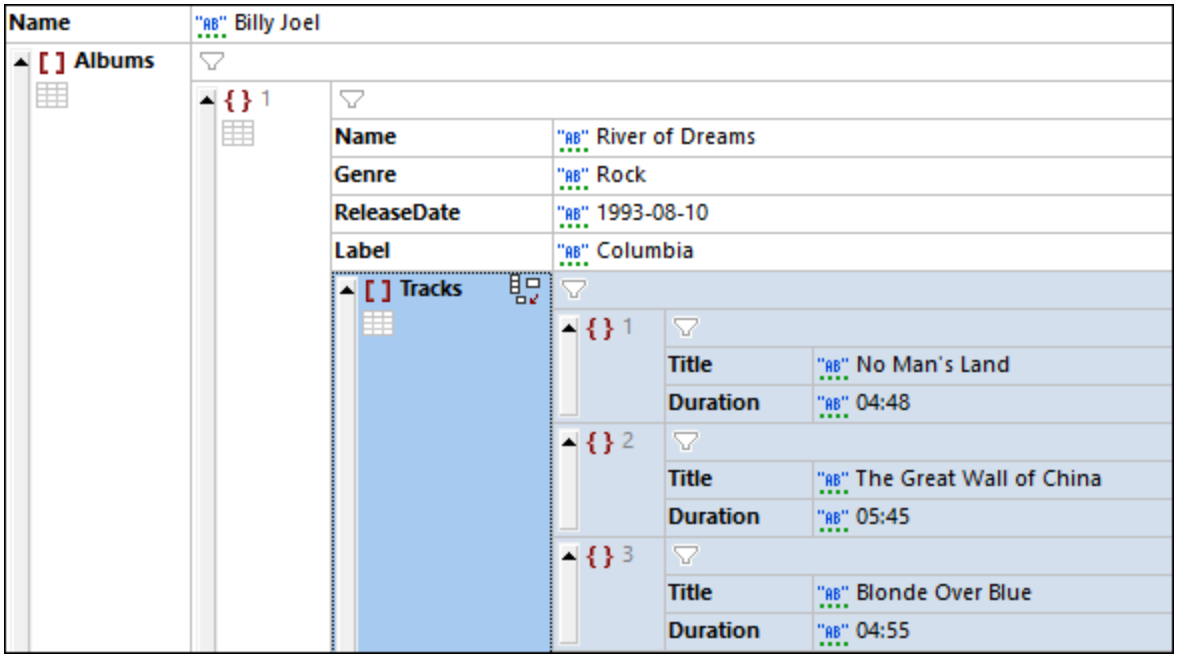
© 2016-2022 Altova GmbH

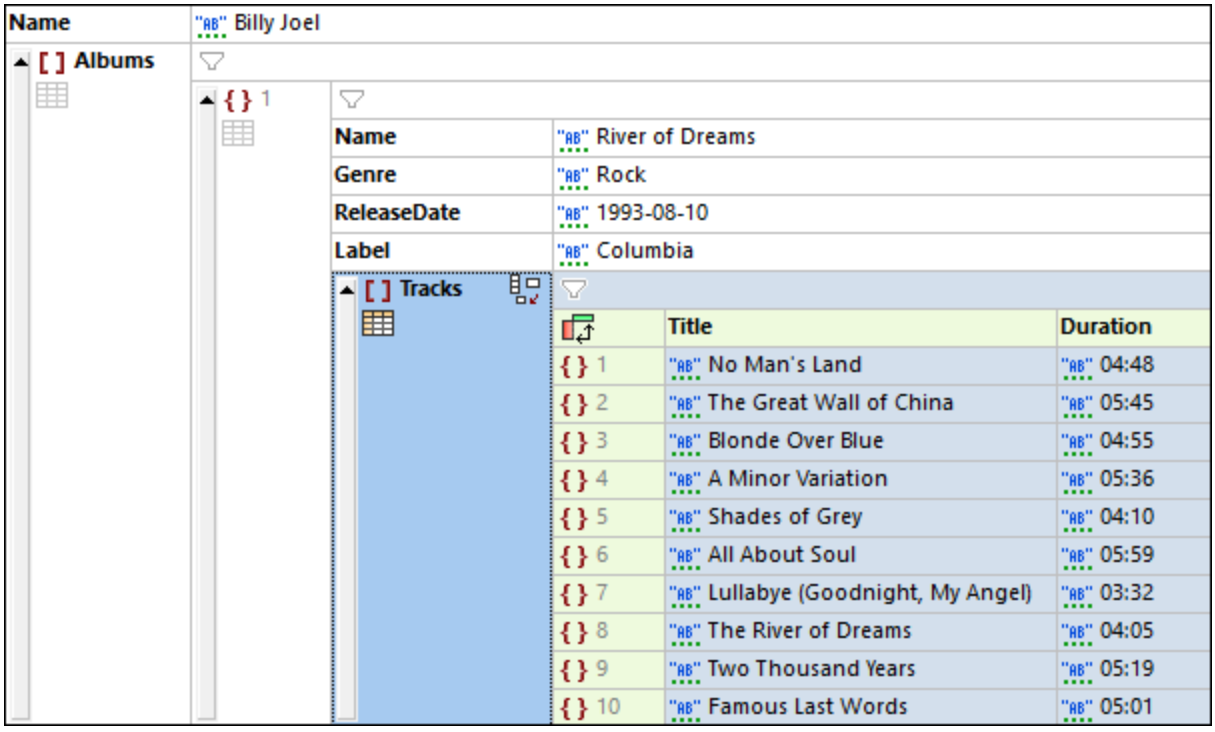


Anmerkung: Kommentare werden in JSONC und JSON5 unterstützt. Formeln[709] sind XMLSpy-spezifisch.

Tabellenansicht

Objekte und Arrays, die mindestens ein Objekt oder Array enthalten können entweder als Liste (*in der Abbildung links markiert*) oder als Tabelle (*in der Abbildung rechts markiert*) angezeigt werden. Sie können für einzelne Objekte und Arrays zwischen der Anzeige als Liste und Tabelle wechseln.





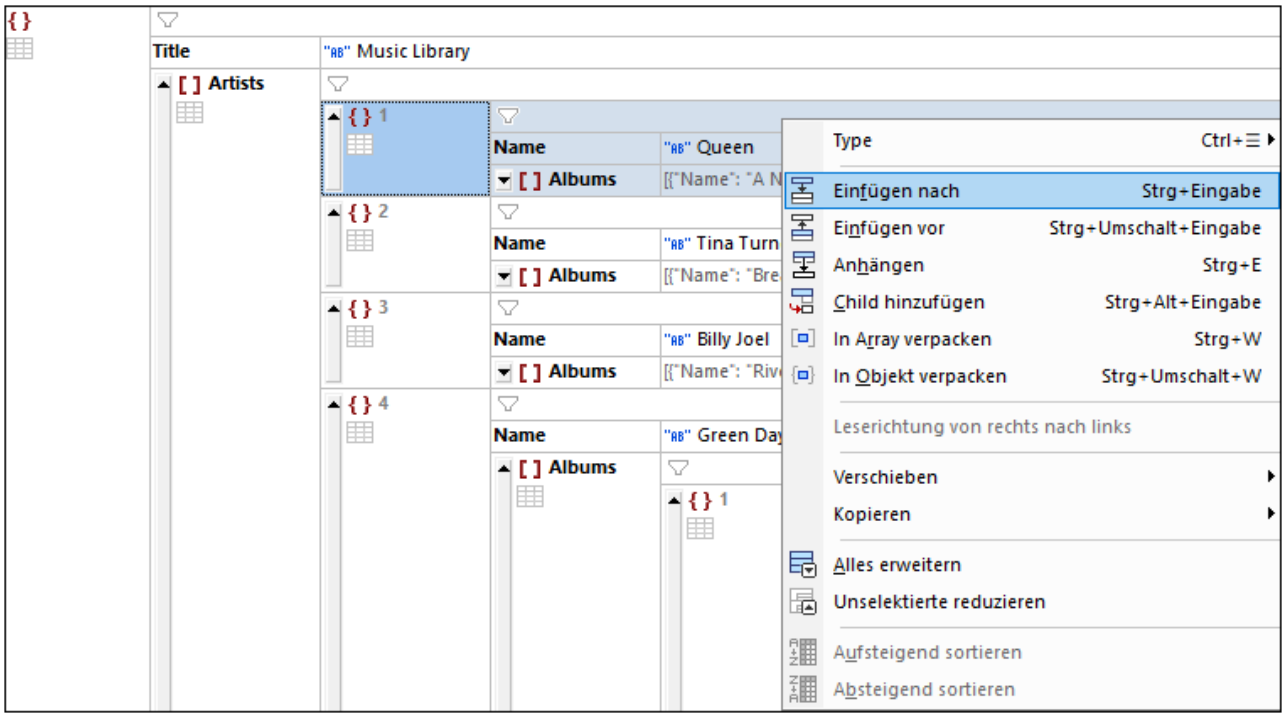
Eine ausführlichere Beschreibung der Tabellenansicht der JSON-Grid-Ansicht finden Sie im Abschnitt Tabellenansicht 704.

 © 2016-2022 Altova GmbH

13.5.2 Bearbeiten der Dokumentstruktur

Sie können die JSON-Dokumentstruktur bearbeiten, indem Sie neue Komponenten hinzufügen, Komponenten löschen oder die ausgewählten Rasterblöcke verschieben. Sie können die Dokumentstruktur auf jede der folgenden Arten bearbeiten:

- Durch Auswahl einer Komponente und Aufruf des entsprechenden Kontextmenübefehls durch Rechtsklick (*siehe Abbildung unten*). Die Befehle werden im Abschnitt Kontextmenü[718] beschrieben.
- Über die Bearbeitungsschaltflächen in den einzelnen Zellen. Diese Schaltflächen werden im Abschnitt Befehlsschaltflächen innerhalb von Zellen[698] weiter unten beschrieben.
- Durch Auswahl einer Komponente und Ziehen mit der Maus an eine neue Stelle, um die Struktur zu ändern.



In der nachstehenden Tabelle finden Sie die Tastaturkürzel für Befehle zur Strukturerstellung.

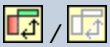
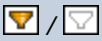
Befehl	Tastaturkürzel
(Gleichrangiges Element) einfügen nach	Strg+Eingabetaste
(Gleichrangiges Element) einfügen vor	Strg+Umschalt+Eingabetaste
(Gleichrangiges) anhängen	Strg+E
Child hinzufügen	Strg+Alt+Eingabetaste

In Array verpacken	Strg+W
In Objekt verpacken	Strg+Umschalt+W

Anmerkung: Wenn Sie eine Komponente z.B. als Eigenschaft hinzufügen (durch Einfügen, Anhängen oder Hinzufügen als Child) und Sie ihren Typ in Objekt oder Array ändern möchten, klicken Sie auf das Typsymbol der Komponente und wählen Sie den gewünschten Typ aus. 699

Befehlsschaltflächen innerhalb von Zellen

In der JSON-Grid-Ansicht stellen Schaltflächen in den Zellen des Rasters Funktionen, die im Kontext der jeweiligen Zelle zur Verfügung stehen, bereit. Häufig verwendete Operationen können dadurch schneller und einfacher aufgerufen werden. In der Tabelle unten sehen Sie eine Liste der Befehle, die innerhalb von Zellen zur Verfügung stehen. Befehle im Zusammenhang mit der Anzeige haben einen blauen Hintergrund. Befehle für die Bearbeitung des Dokuments (Struktur und Inhalt) haben einen beigen Hintergrund. Zusätzliche Befehle im Zusammenhang mit einer bestimmten Zelle können Sie über das Kontextmenü 718 der Zelle aufrufen. Anmerkung: Der Name des Befehls einer Zellen-Schaltfläche wird als Tooltipp angezeigt, wenn Sie den Mauszeiger darüber platzieren.

	Optimiert die Breite von Grid-Spalten entsprechend dem Zelleninhalt.
	Bei farbiger Anzeige ist die Tabellenansicht 704 aktiviert, andernfalls ist sie deaktiviert. Klicken Sie darauf, um die Anzeige zu wechseln.
	Ein Ein/Aus-Befehl in der linken oberen Zelle einer Tabelle. Zeigt Zeilen als Spalten an und umgekehrt.
	Wenn das Symbol farbig angezeigt wird, ist ein Filter 709 für die Tabelle, das Objekt oder Array aktiv. Andernfalls ist der Filter deaktiviert. Klicken Sie darauf, um den Filter zu deaktivieren/zu aktivieren. Um den Ausdruck zu bearbeiten, doppelklicken Sie darauf.
	Ein/Aus-Befehl für den Zeilenumbruch in allen Zellen. Wenn die Schaltfläche ausgewählt ist, ist der Zeilenumbruch aktiviert. Standardmäßig wird nur der Inhalt von Elementen umbrochen. Wenn auch die Namen von Elementen umbrochen werden sollen, wählen Sie diese Option aus der Dropdown-Liste des Symbols aus.
	In einem Array: Array-Element an die Liste oder Tabelle anhängen. In einem Objekt: `Schlüssel:Wert`-Paar (als Zeile in einer Liste oder (in der Tabellenansicht) als Zelle einer neuen Tabellenspalte) anhängen.
	Ein leeres `Schlüssel:Wert`-Paar hinzufügen; der Typ des Werts ist standardmäßig `string`.
	Datentyp eines Eigenschaftswerts auswählen oder einen Kommentar oder eine Formel eingeben.

Anmerkung: Eine Beschreibung weiterer Symbolleistenbefehle finden Sie im Abschnitt Kontextmenü in der Grid-Ansicht 718.

© 2016-2022 Altova GmbH

13.5.3 Bearbeiten des Dokumentinhalts

Inhalt lässt sich in der JSON-Grid-Ansicht ganz einfach bearbeiten: Doppelklicken Sie einfach in die gewünschte Zelle und bearbeiten Sie den Inhalt nach Bedarf. Des Weiteren bietet die JSON-Grid-Ansicht die folgenden Bearbeitungsfunktionalitäten:

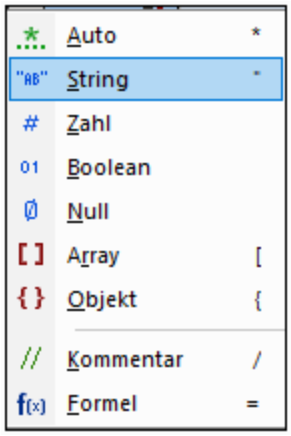
- Funktion zur Typauswahl sowie automatische Typerkennung und Wertekonvertierung
- Autokomplettierung in auf einem JSON-Schema basierenden JSON-Dokument

Typauswahl und automatische Typerkennung

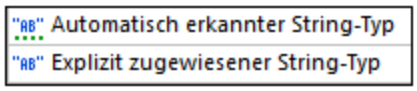
In der JSON-Grid-Ansicht wird zwischen den folgenden Typkategorien unterschieden:

- *Einfache Typen:* String, Zahl, Boolean, Null
- *Spezielle einfache Typen:* Auto (wird anhand des Werts ermittelt), Kommentar (JSONC, JSON5)
- *Container-Typen:* Objekt, Array
- *Spezielle Container-Typen:* Formel

Der Typ eines Werts wird durch ein Symbol vor dem Wert angezeigt (*siehe Abbildung unten*). Um einen Typ zu ändern, klicken Sie auf sein Symbol und wählen Sie den gewünschten Typ aus dem angezeigten Menü aus. In der Abbildung unten sehen Sie die Symbole und Shortcuts von Typen:



Automatisch erkannte Typen weisen grüne Punkte unter dem Symbol auf (*siehe Abbildung unten*). Ein expliziter Typ ist ein Typ, den Sie zuweisen.



Es kommt in zwei Situationen zu Aktionen im Zusammenhang mit dem Typ:

- Wenn ein JSON-Dokument geladen wird: Alle einfachen Typen werden in `Auto` konvertiert. Diese Typen werden automatisch anhand ihres Werts erkannt. So wird z.B. **`"MyString"`** automatisch als String-Typ erkannt, **`123`** als Zahl, **`true`** als Boolean und **`null`** als Null-Typ. Wenn der Typ eines String nicht eindeutig ist, muss er explizit ausgewählt werden.

- Wenn eine neue Datenstruktur oder ein neuer Wert eingegeben wird: Einfache Typen werden automatisch erkannt und der Typ wird automatisch zugewiesen. Sie können den Typ später gegebenenfalls ändern.

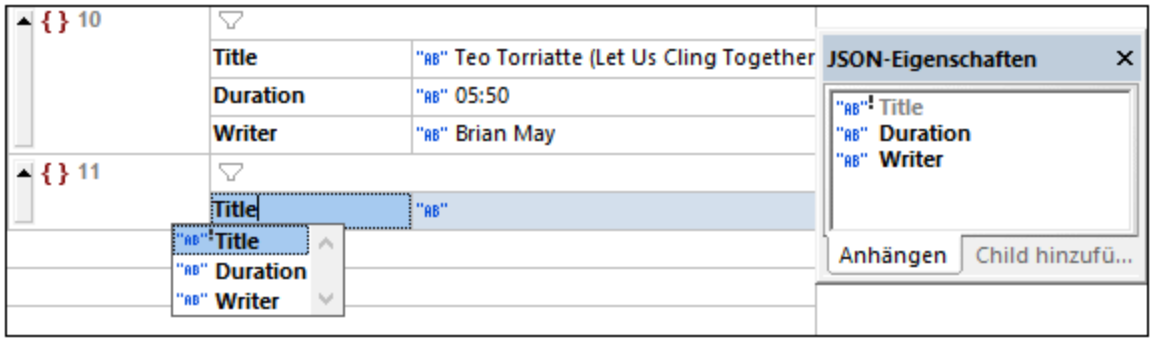
Anmerkung: Mit Hilfe der Einstellungen der JSON-Grid-Ansicht[721] können Sie festlegen, (i) wie mit Typänderungen verfahren werden soll, wenn mehrere Zellen ausgewählt wurden, und (ii) wie Werte von atomaren Typen behandelt werden sollen, wenn der Typ von einem Array in ein Objekt geändert wird.

Autokomplettierung

Die Autokomplettierung ist aktiv, wenn das bearbeitete JSON-Dokument (JSON-Instanz/Schema oder Avro-Schema) mit einem Schema verknüpft ist.

- Wenn es sich beim Dokument um ein JSON-Schema handelt, basiert die Autokomplettierung auf der durch das Schlüsselwort `$schema`[682] angegebenen Schemaversion. Nähere Informationen dazu finden Sie unter JSON-Schema-Version[727].
- Wenn es sich beim Dokument um eine JSON/JSON5-Instanz handelt, so muss der Instanz ein JSON-Schema zugewiesen sein[765], damit die Autokomplettierung aktiv ist.
- Wenn es sich beim Dokument um ein Avro-Datendokument im JSON-Format handelt, so muss der Instanz ein Avro-Schema zugewiesen werden[765], damit die Autokomplettierung funktioniert.
- Wenn es sich beim Dokument um ein Avro-Schema[782] handelt, so wird es automatisch mit dem Schema für Avro-Schema verknüpft und die Autokomplettierung erfolgt auf Basis dieses Schemas.

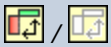
Bei der Autokomplettierung werden Ihnen an der Cursorposition verfügbare Eingabeoptionen vorgeschlagen. Die Vorschläge erfolgen (i) über Pop-ups im Hauptfenster und (ii) die Eingabehilfen (*siehe Abbildung unten*). In den Pop-ups und Eingabehilfen wird jeweils eine Liste von Einträgen angezeigt, die an dieser Cursorposition gültig sind. Mit Hilfe der Pfeiltasten können Sie durch die Einträge in der Pop-up-Liste navigieren. Wählen Sie einen Eintrag aus dem Pop-up-Fenster aus oder doppelklicken Sie in der Eingabehilfe auf einen Eintrag, um diesen einzufügen.



Befehlsschaltflächen innerhalb von Zellen

In der JSON-Grid-Ansicht stellen Schaltflächen in den Zellen des Rasters Funktionen, die im Kontext der jeweiligen Zelle zur Verfügung stehen, bereit. Häufig verwendete Operationen können dadurch schneller und einfacher aufgerufen werden. In der Tabelle unten sehen Sie eine Liste der Befehle, die innerhalb von Zellen zur Verfügung stehen. Befehle im Zusammenhang mit der Anzeige haben einen blauen Hintergrund. Befehle für die Bearbeitung des Dokuments (Struktur und Inhalt) haben einen beigen Hintergrund. Zusätzliche Befehle im Zusammenhang mit einer bestimmten Zelle können Sie über das Kontextmenü[718] der Zelle aufrufen. Anmerkung: Der Name des Befehls einer Zellen-Schaltfläche wird als Tooltipp angezeigt, wenn Sie den Mauszeiger darüber platzieren.

© 2016-2022 Altova GmbH

	Optimiert die Breite von Grid-Spalten entsprechend dem Zelleninhalt.
	Bei farbiger Anzeige ist die Tabellenansicht[704] aktiviert, andernfalls ist sie deaktiviert. Klicken Sie darauf, um die Anzeige zu wechseln.
	Ein Ein/Aus-Befehl in der linken oberen Zelle einer Tabelle. Zeigt Zeilen als Spalten an und umgekehrt.
	Wenn das Symbol farbig angezeigt wird, ist ein Filter[709] für die Tabelle, das Objekt oder Array aktiv. Andernfalls ist der Filter deaktiviert. Klicken Sie darauf, um den Filter zu deaktivieren/zu aktivieren. Um den Ausdruck zu bearbeiten, doppelklicken Sie darauf.
	Ein/Aus-Befehl für den Zeilenumbruch in allen Zellen. Wenn die Schaltfläche ausgewählt ist, ist der Zeilenumbruch aktiviert. Standardmäßig wird nur der Inhalt von Elementen umbrochen. Wenn auch die Namen von Elementen umbrochen werden sollen, wählen Sie diese Option aus der Dropdown-Liste des Symbols aus.
	In einem Array: Array-Element an die Liste oder Tabelle anhängen. In einem Objekt: `Schlüssel:Wert`-Paar (als Zeile in einer Liste oder (in der Tabellenansicht) als Zelle einer neuen Tabellenspalte) anhängen.
	Ein leeres `Schlüssel:Wert`-Paar hinzufügen; der Typ des Werts ist standardmäßig `string`.
	Datentyp eines Eigenschaftswerts auswählen oder einen Kommentar oder eine Formel eingeben.

Anmerkung: Eine Beschreibung weiterer Symbolleistenbefehle finden Sie im Abschnitt Kontextmenü in der Grid-Ansicht[718].

Bei Änderung validieren

Der *Bei Bearbeitung validieren*-Modus ist standardmäßig aktiviert. Wenn dieser Modus aktiv ist, werden, während Sie ein Dokument in der JSON-Grid-Ansicht bearbeiten, Wohlgeformtheits- und Validierungsüberprüfungen durchgeführt. Damit ein JSON-Dokument (nicht nur auf Wohlgeformtheit geprüft, sondern auch) validiert werden kann, muss dem JSON-Dokument ein JSON-Schema zugewiesen sein. Fehler im Text werden rot angezeigt. Die Stelle im Dokument wird durch ein rotes Ausrufezeichen markiert. Nähere Informationen dazu finden Sie unter Validieren von JSON-Dokumenten[765].

Der *Bei Bearbeitung validieren*-Modus kann entweder (i) über den Menübefehl **XML | Bei Bearbeitung validieren**[1359], (ii) die Symbolleisten-Schaltfläche **Bei Bearbeitung validieren** oder (iii) die Option *Bei Bearbeitung* der Validierungseinstellungen des Dialogfelds "Optionen"[1603] aktiviert bzw. deaktiviert werden.

Suchen und Ersetzen

Mit Hilfe der Befehle **Suchen (Strg+F)** und **Ersetzen (Strg+H)** aus dem Menü "Bearbeiten" können Sie nach Text in der JSON-Grid-Ansicht suchen und diesen ersetzen. Die Suchergebnisse werden orange markiert, wobei auch diese enthaltende Komponenten orange markiert werden. Bei Zellen, die lange Textstrings enthalten, wird nur die Zelle markiert.

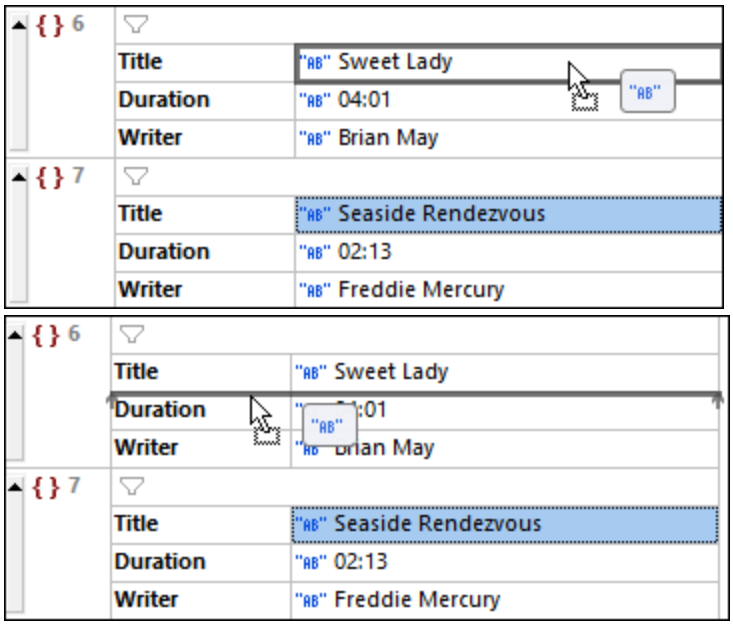
13.5.4 Drag-and-Drop in der Grid-Ansicht

Die Grid-Ansicht bietet ein äußerst nützliches Drag Overlay, mit Hilfe dessen Sie ein JSON-Dokumentfragment aus einem in XMLSpy, einer externen Applikation oder sogar einer Website geöffneten Dokument in die Grid-Ansicht ziehen können.

Wenn Sie den Cursor über den Ziel-Node platzieren, erscheint ein Drag Overlay, das nicht nur Informationen darüber enthält, was gerade mit der Maus gezogen wird, sondern auch darüber, wie dieser Inhalt nach dem Loslassen der Maustaste erstellt wird. So wird etwa in der Abbildung unten der Wert `Seaside Rendezvous` mit der Maus gezogen (verschoben). Dass es sich dabei um einen String-Wert handelt, wird durch das Popup `AB` angezeigt.

- Wenn Sie die Maus über ein Wertefeld platzieren (wie in der Abbildung links), wird das gesamte Feld markiert, wodurch angezeigt wird, dass der Wert des markierten Felds durch den Wert des dorthin gezogenen Felds ersetzt wird.
- Wenn Sie die Maus über einen Node-Namen platzieren (wie `Duration` in der Abbildung rechts), erscheint eine Einfügezeile, wodurch angezeigt wird, dass der Wertestring hierhin gezogen wird, um den Wert-Node eines Schlüssel-Wert-Paars zu erstellen. Der Schlüssel ist in diesem Fall ein leerer String.

Anmerkung: Sie können jede Verschiebung immer mit **Strg+Z** oder über den Menübefehl **Bearbeiten | Rückgängig** rückgängig machen.



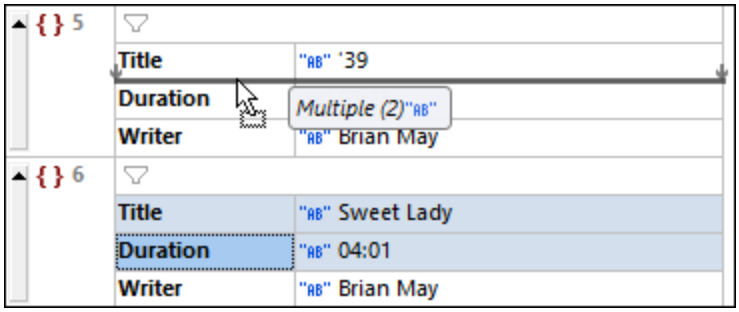
Im Drag Overlay angezeigte Informationen

Es werden die folgenden Arten von Drag Overlay-Informationen angezeigt.

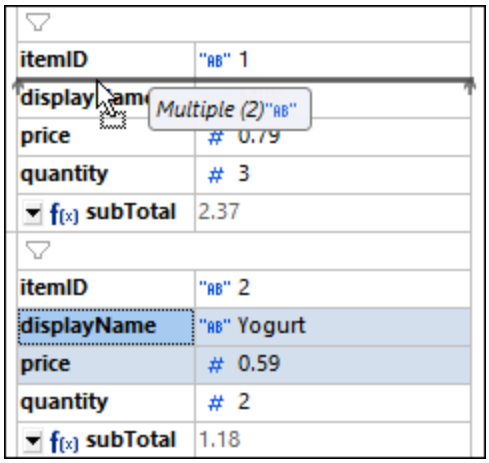
Normale Grid-Ansicht

© 2016-2022 Altova GmbH

- Wenn ein Wert in ein Wertefeld gezogen wird, wird der Wert im Zielfeld durch den Quellwert überschrieben (*siehe Abbildung links oben*).
- Wenn ein Wert als Node in die Struktur gezogen wird, wird der Wert-Node eines Schlüssel-Wert-Paars erstellt (*siehe Abbildung rechts oben*). Der Schlüssel ist dann ein leerer String.
- Im Fall eines Overlay mehrerer Nodes eines einzigen Typs werden der Typ des gezogenen Nodes und die Anzahl dieser Nodes angezeigt. Beachten Sie, dass Sie auf den Namen des Node klicken müssen, um einen Node (und nicht seinen Wert) auszuwählen. In der Abbildung unten sehen Sie anhand der Informationen im Drag Overlay, dass zwei Schlüssel-Wert-Paare vom Typ String gezogen werden.



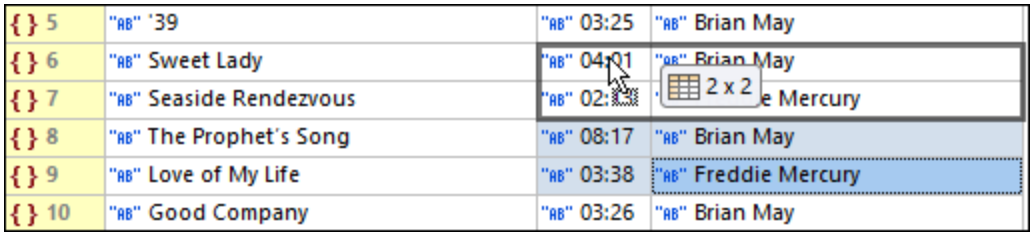
- Das Overlay für mehrere Nodes unterschiedlichen Typs zeigt an, dass die neuen Nodes mit dem Standardtyp "String" erstellt werden (*siehe Abbildung unten*). Sie können den Typ des mit der Maus gezogenen Node durch Klick auf die entsprechenden *Typ bearbeiten*-Symbole ändern. Beachten Sie, dass Sie auf den Namen des Node klicken müssen, um einen Node (und nicht seinen Wert) auszuwählen.



Tabellenansicht

- Wenn Werte mit der Maus gezogen werden, wird die Anzahl der ausgewählten Zellen durch **`Spalten x Zeilen`** (*siehe Abbildung unten*) angegeben. Die entsprechende Matrix wird durch die Matrix der mit der Maus gezogenen Zellen ersetzt. Die Zielzelle (d.h. die Zelle auf die Sie die Matrix ziehen) erhält den Wert der obersten linken Zelle der Matrix. Die Zellen rechts und unterhalb dieser Zielzelle werden durch die restlichen mit der Maus gezogenen Zellen ausgefüllt. So werden etwa in der Abbildung unten die blauen Zellen auf die Zelle **`Duration`** von Zeile 6 gezogen. Dadurch werden die markierten Zellen durch die neuen Zellen ersetzt. Wenn die Matrix der mit der Maus gezogenen Zellen die Tabellengröße übersteigt, werden entsprechend viele Spalten und/oder Zeilen hinzugefügt, um Platz für alle mit der

Maus gezogenen Zellen zu schaffen. In diesem Fall werden die erweiterten Grenzen durch gestrichelte Linien angezeigt.



- Wenn das Ziel die Node-Struktur und nicht eine Zelle ist, wird der hinzugefügte Node zusammen mit (i) den Spalten, die hinzugefügt werden (innerhalb von eckigen Klammern) und (ii) der Anzahl der Instanzen des neuen Eintrags angezeigt. So wird z.B. in der Abbildung unten ein Array bestehend aus zwei Objekten, mit jeweils einem **`Title`**, **`Duration`** und **`Writer`** Schlüssel-Wert-Paar hinzugefügt.



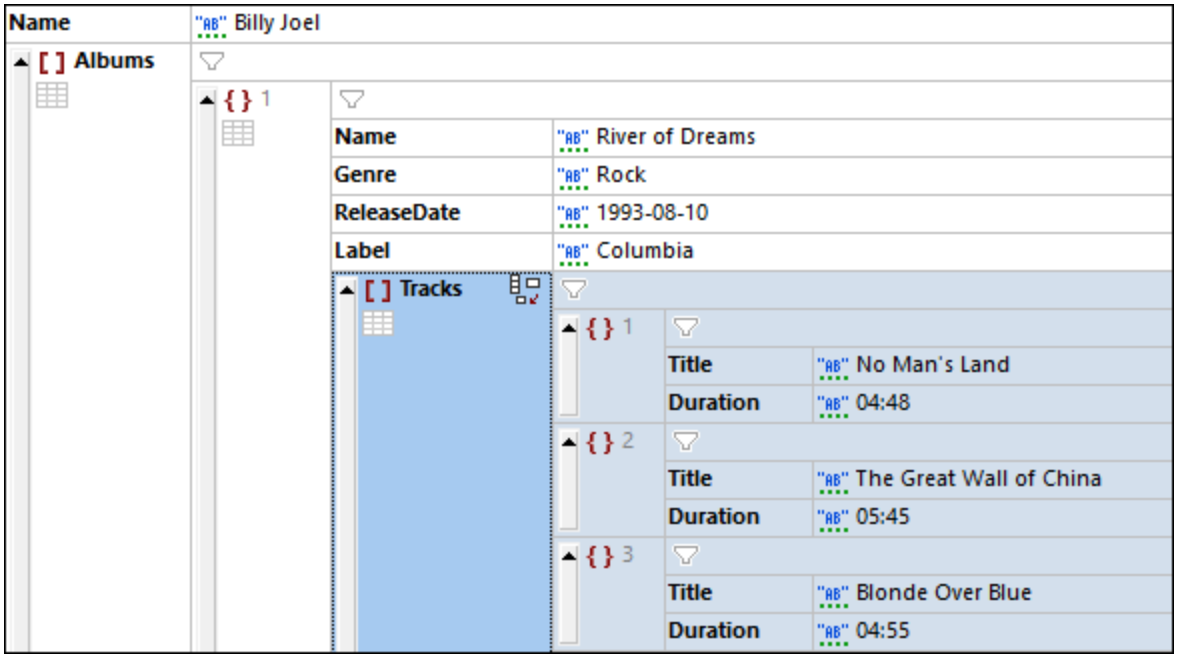
- Um eine Tabellenspalte zu verschieben, wählen Sie diese durch Klick auf ihre Kopfzeile aus und ziehen Sie sie auf die Spaltenüberschrift neben derjenigen, in die sie die Spalte verschieben möchten. Daraufhin wird mittels einer Einfügelinie angezeigt, auf welcher Seite der Zielspalte die verschobene Spalte eingefügt wird.

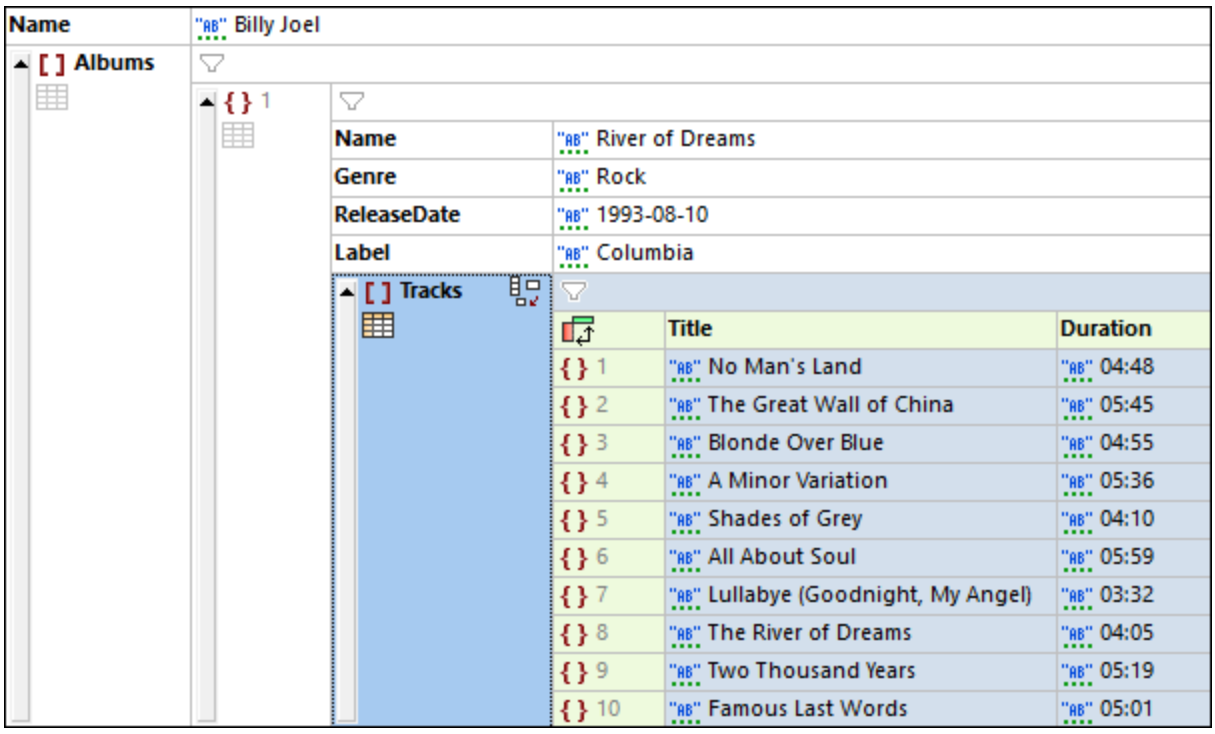
13.5.5 Tabellenansicht

Die Tabellenansicht

Objekte und Arrays, die mindestens ein Objekt oder Array enthalten, können entweder als Liste (*in der Abbildung links markiert*) oder als Tabelle (*in der Abbildung rechts markiert*) angezeigt werden. Um zwischen der Anzeige als Liste und Tabelle zu wechseln, klicken Sie auf das Tabellensymbol unterhalb des Objekt- oder Array-Symbols *(siehe Abbildung)*. In Array-Tabellen werden Array-Elemente als Zeilen angezeigt. So wird z.B. das Array `Tracks` in der Abbildung rechts unten als Tabelle angezeigt. Dieses Array besteht aus Child-Objekten, die als Zeilen angezeigt werden. Die Eigenschaften der einzelnen Objekte (`Title` und `Duration` in den Abbildungen unten) werden als Spalten angezeigt, wobei die Namen gemeinsamer Eigenschaften als Spaltenüberschriften angezeigt werden. Um zwischen Zeilen und Spalten zu wechseln, klicken Sie auf das Symbol in der linken oberen Zelle der Tabelle.

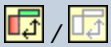
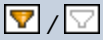
 © 2016-2022 Altova GmbH





Symbole zur Anzeige und Bearbeitung in der Tabellenansicht

Über die in der Tabelle unten aufgelisteten Symbole stehen Anzeige- und Bearbeitungsfunktionen zur Verfügung.

	Optimiert die Breite von Grid-Spalten entsprechend dem Zelleninhalt.
	Bei farbiger Anzeige ist die Tabellenansicht[704] aktiviert, andernfalls ist sie deaktiviert. Klicken Sie darauf, um die Anzeige zu wechseln.
	Ein Ein/Aus-Befehl in der linken oberen Zelle einer Tabelle. Zeigt Zeilen als Spalten an und umgekehrt.
	Wenn das Symbol farbig angezeigt wird, ist ein Filter[709] für die Tabelle, das Objekt oder Array aktiv. Andernfalls ist der Filter deaktiviert. Klicken Sie darauf, um den Filter zu deaktivieren/zu aktivieren. Um den Ausdruck zu bearbeiten, doppelklicken Sie darauf.
	Ein/Aus-Befehl für den Zeilenumbruch in allen Zellen. Wenn die Schaltfläche ausgewählt ist, ist der Zeilenumbruch aktiviert. Standardmäßig wird nur der Inhalt von Elementen umbrochen. Wenn auch die Namen von Elementen umbrochen werden sollen, wählen Sie diese Option aus der Dropdown-Liste des Symbols aus.
	In einem Array: Array-Element an die Liste oder Tabelle anhängen. In einem Objekt: `Schlüssel:Wert`-Paar (als Zeile in einer Liste oder (in der Tabellenansicht) als Zelle einer neuen Tabellenspalte) anhängen.
	Ein leeres `Schlüssel:Wert`-Paar hinzufügen; der Typ des Werts ist standardmäßig `string`.
	Datentyp eines Eigenschaftswerts auswählen oder einen Kommentar oder eine Formel eingeben.

Bearbeiten in der Tabellenansicht

In der Tabellenansicht können die folgenden unten aufgelisteten Bearbeitungsvorgänge durchgeführt werden.

Hinzufügen von Zeilen zu einer Tabelle

Zeilen können auf die folgenden Arten hinzugefügt werden:

- *Hinzufügen einer Child-Zeile zur Tabelle:* Wählen Sie die Tabelle aus und klicken Sie im Kontextmenü auf den Befehl **Child hinzufügen (Strg+Umschalt+Eingabetaste**). Klicken Sie alternativ dazu auf das **Child anhängen**-Symbol *(siehe Liste der Symbole oben)*. Daraufhin wird am unteren Rand der Tabelle eine Zeile angehängt.
- *Anhängen einer gleichrangigen Zeile an die Tabelle, wenn eine Zeile ausgewählt ist:* Wählen Sie eine Zeile aus und klicken Sie im Kontextmenü auf den Befehl **Anhängen (Strg+Eingabetaste**). Daraufhin wird am unteren Rand der Tabelle eine Zeile angehängt.
- *Einfügen einer gleichrangigen Zeile oberhalb der ausgewählten Zeile:* Wählen Sie eine Zeile aus und klicken Sie im Kontextmenü auf den Befehl **Einfügen (Strg+Alt+Eingabetaste**).

Eingabe oder Bearbeitung eines Eigenschaftswerts

Wählen Sie die Tabellenzelle, in der sich der Eigenschaftswert befindet, aus und geben Sie den Wert ein. Wenn es sich beim Wert um einen String, eine Zahl, einen Booleschen Wert oder Null handelt, wird der Datentyp automatisch erkannt. Wenn der Typ nicht eindeutig ist, wird er auf `string` gesetzt. Sie können die automatische Auswahl durch Verwendung des Symbols zur Datentypauswahl *(siehe Symbolliste oben)* außer Kraft setzen. Alternativ dazu können Sie den entsprechenden Shortcut wählen (*siehe Shortcut-Tabelle unten*).

*	Auto (erkennt String, Zahl, Boolean, Null und definiert den Typ entsprechend)

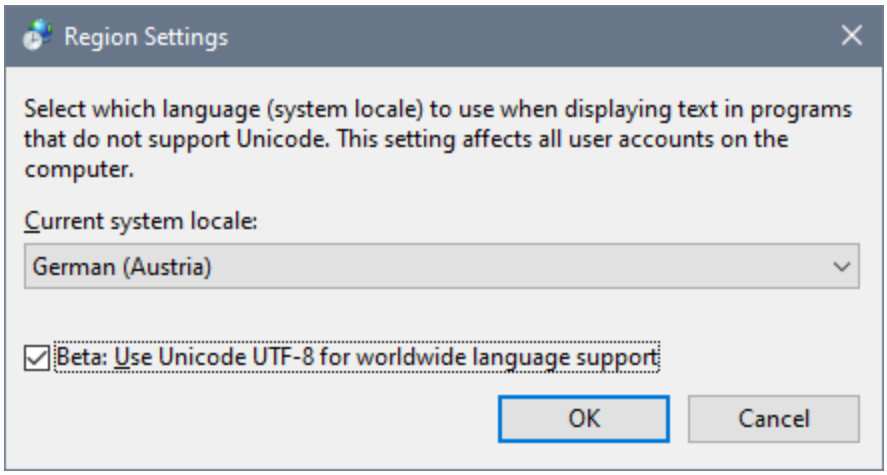
© 2016-2022 Altova GmbH

"	String
[	Array
{	Objekt
/	Kommentar (Dokument muss JSONC oder JSON5 sein)
=	Formel (Dokument muss JSONC oder JSON5 sein)

Hinzufügen einer leeren Zeilenzelle in einer neuen Spalte
Sie können zu einer Zeile eine neue Zelle hinzufügen. Die neue Zelle bildet dann Teil einer neu erstellten Spalte. Um die neue Zelle hinzuzufügen, wählen Sie die gewünschte Zelle aus und klicken Sie im Kontextmenü auf **Child hinzufügen (Strg+Umschalt+Eingabetaste**). Klicken Sie alternativ dazu auf das **Zeilenzelle anhängen**-Symbol der Zeile *(siehe Liste der Symbole oben)*. Daraufhin wird die Zeilenzelle in einer neuen Spalte erstellt. Der Typ des Werts in der Zelle ist standardmäßig `string`. Geben Sie den Wert der Eigenschaft in die Zelle und den Namen der Eigenschaft als Spaltenüberschrift ein. Die anderen Zellen in der neu erstellten Spalte sind leer.

Sortieren von Tabellenzeilen nach den Werten einer ausgewählten Spalte
Sie können die Zeilen einer Tabelle nach den relativen Werten eines ihrer Child-Nodes (als Spalte) sortieren. So können Sie etwa Zeilen auf Basis der Spalte *LastName* sortieren, damit die sich wiederholenden Elemente der Tabelle alphabetisch sortiert werden. Um eine Spalte zu sortieren, wählen Sie die Spaltenüberschrift aus und klicken Sie anschließend auf die Schaltfläche **Aufsteigend sortieren** oder **Absteigend sortieren** der Grid-Ansichts-Symbolleiste[692]. Diese Befehle stehen auch im Menü "JSON"[1369] zur Verfügung.

Für die Sortierreihenfolge ist es in manchen Sprachen, v.a. für solche mit einem nicht lateinischen Alphabet, von Vorteil, im Dialogfeld "Sprache Region" von Versionen ab Windows 10 die beta Unicode UTF-8-Unterstützung zu aktivieren. Gehen Sie dazu folgendermaßen vor: Gehen Sie zu Ihrem Windows-*Einstellungsdialogfeld* und suchen Sie nach *Spracheinstellungen*. Klicken Sie unter *Verwandte Einstellungen* auf *Verwaltung Spracheinstellungen*. Gehen Sie im daraufhin angezeigten Dialogfeld "Region" zum Register *Verwaltung* und klicken Sie unter *Sprache für Unicode-inkompatible Programme* auf *Gebietsschema ändern*. Aktivieren Sie im daraufhin angezeigten Dialogfeld "Region" (*Abbildung unten*) die Option*Beta: Unicode UTF-8 für weltweite Sprachunterstützung* und klicken Sie auf **OK**.



Tabellenansicht und externe Applikationen

Mit Hilfe der Tabellenstruktur können Sie Daten zwischen der Tabellenansicht und einem Tabellenkalkulationsprogramm wie z.B. MS Excel austauschen. Um Daten aus der Tabellenansicht zu verschieben, gehen Sie folgendermaßen vor:

1. Wählen Sie in die zu kopierenen Nodes in der Tabellenansicht aus. Klicken Sie dazu auf die Zellen selbst, auf die Spalten- oder Zeilenüberschriften oder die gesamte Tabelle. Wenn Sie die gesamte Tabelle oder Spaltenüberschriften auswählen, wird auch der Text der Spaltenüberschriften kopiert; ansonsten nicht. (In der Abbildung unten wurden die Zeilen 1 bis 8 zusammen mit ihre Spaltenüberschriften ausgewählt).



[ ] Tracks	Title	Duration	Writer
{ } 1	"AB" Death on Two Legs	"AB" 03:43	"AB" Freddie Mercury
{ } 2	"AB" Lazing on a Sunday Afternoon	"AB" 01:08	"AB" Freddie Mercury
{ } 3	"AB" I'm in Love with My Car	"AB" 03:05	"AB" Roger Taylor
{ } 4	"AB" You're My Best Friend	"AB" 02:50	"AB" John Deacon
{ } 5	"AB" '39	"AB" 03:25	"AB" Brian May
{ } 6	"AB" Sweet Lady	"AB" 04:01	"AB" Brian May
{ } 7	"AB" Seaside Rendezvous	"AB" 02:13	"AB" Freddie Mercury
{ } 8	"AB" The Prophet's Song	"AB" 08:17	"AB" Brian May

2. Wählen Sie den Kontextmenübefehl Kopieren | Als tabulatorgetrennten Text kopieren[1300] aus.
3. Fügen Sie die Daten direkt in das Tabellenkalkulationsprogramm ein.

	A	B	C
1	Title	Duration	Writer
2	Death on Two Legs	3:43	Freddie Mercury
3	Lazing on a Sunday Afternoon	1:08	Freddie Mercury
4	I'm in Love with My Car	3:05	Roger Taylor
5	You're My Best Friend	2:50	John Deacon
6	'39	3:25	Brian May
7	Sweet Lady	4:01	Brian May
8	Seaside Rendezvous	2:13	Freddie Mercury
9	The Prophet's Song	8:17	Brian May

Der Datenaustausch funktioniert in beide Richtungen. Sie können auch Daten aus einem Arbeitsblatt in die Tabellenansicht kopieren. Gehen Sie dazu folgendermaßen vor:

1. Wählen Sie in der externen Applikation einen Bereich aus und kopieren Sie ihn (mit **Strg+C**) in die Zwischenablage.
2. Wählen Sie in der Tabellenansicht eine einzelne Zelle aus.
3. Fügen Sie die kopierten Daten mit **Strg+V** ein.

Die Daten werden mit der gleichen Struktur wie in der Originalstruktur ab der in der Tabellenansicht ausgewählten Zelle in die Tabelle kopiert. Beachten Sie bitte Folgendes:

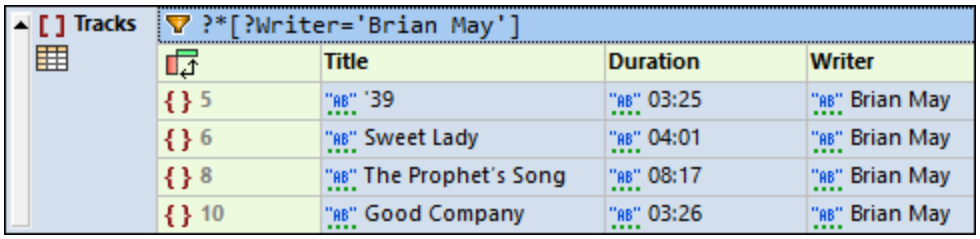
© 2016-2022 Altova GmbH

- Wenn die Daten in eine Zelle kopiert werden, in der neue Zeilen und/oder Spalten erstellt werden können, ohne dass die bestehenden Daten dadurch ungültig werden, so werden neue Zeilen und/oder Spalten erstellt.
- Zeilen werden als neue Objekte erstellt, während Spalten als Eigenschaften der (Zeilen)-Objekte der Tabelle erstellt werden.
- Wenn die Tabellenstruktur nicht geändert werden kann, so dass sie gültig ist, werden die ursprünglichen Daten der ausgewählten Zelle durch die neuen Daten in Form von Text überschrieben.

Verwenden Sie für komplexere Datenaustauschvorgänge die einzigartigen Import/Export[1470]-Funktionen von XMLSpy.

13.5.6 Filter

Mit Hilfe von Filtern können Sie die Anzeige von Objekten und Arrays filtern. So wurde z.B. in der Abbildung unten ein Filter (bei dem es sich um einen XQuery 3.1-Ausdruck handelt) auf ein Array angewendet, sodass nur CD-Tracks, die von Brian May geschrieben wurden, angezeigt werden. In JSON-Terminologie bedeutet dies, dass nur die Objekt-Children von **`Tracks`**, die eine Eigenschaft **`Writer`** mit dem String-Wert `Brian May` haben, angezeigt werden. Mit dem XQuery-Ausdruck des Filters werden alle Child-Objekte von **`Tracks`** ermittelt und nur diejenigen ausgewählt, bei denen die Eigenschaft **`Writer`** den String `'Brian May'` enthält.



Jeder Filter wird separat ausgeführt. Andere Filter oder Formeln im Dokument haben keine Auswirkung darauf.

Informationen zur Erstellung von XQuery-Ausdrücken für JSON-Dokumente finden Sie im Abschnitt XQuery-Ausdrücke für JSON[772].

Anmerkung: Sie können Ausdrücke für Filter und Formeln in der Grid-Ansicht aus Gründen der Übersichtlichkeit auch in mehreren Zeilen eingeben. Um im Ausdruck eine neue Zeile hinzuzufügen, drücken Sie **Strg+Eingabetaste**.

Einrichten eines Filters

Um einen Filter einzurichten, klicken Sie mit der rechten Maustaste auf das Element, das Sie filtern möchten, wählen Sie im angezeigten Kontextmenü den **Filter** aus, geben Sie den XQuery-Ausdruck ein und drücken Sie die **Eingabetaste**. Die *Filter*-Zelle ist durch das Symbol *(siehe Abbildung oben)* gekennzeichnet. Durch Klick auf das *Filter*-Symbol können Sie den Filter ein- und ausschalten.

Beachten Sie zu Filtern die folgenden Punkte:

- Filter können nur auf Objekte und Arrays angewendet werden.

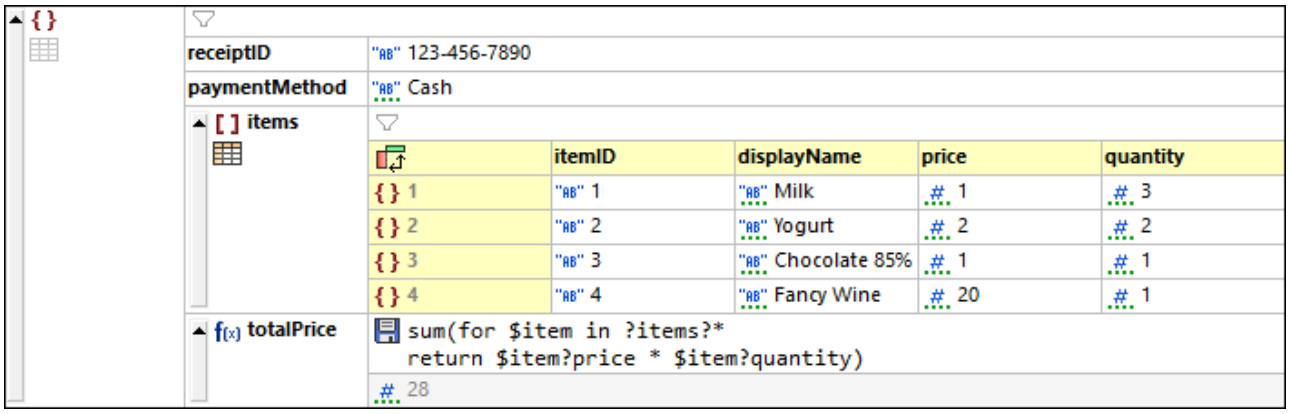
- Der Kontext-Node des XQuery-Ausdrucks des Filters ist der aktuelle Node. So ist z.B. in der Abbildung oben der Kontext-Node des XQuery-Ausdrucks des Filters der Node **`Tracks`**.
- Filter werden im JSON-Dokument **nicht als JSON-Inhalt gespeichert**, sondern für die spätere Verwendung in einer speziellen Metadatendatei gespeichert. Wenn die Filter gespeichert werden, werden Sie automatisch auf das entsprechende Objekt/Array angewendet, wenn das Dokument in der JSON-Grid-Ansicht geöffnet wird.
- Filter können verschachtelt sein. Verschachtelte Filter werden auf den gefilterten Inhalt des übergeordneten Filters angewendet.
- Um in einem Ausdruck eine neue Zeile hinzuzufügen, drücken Sie **Strg+Eingabetaste**.

13.5.7 Formeln

Eine Formel ist ein XQuery 3.1.Ausdruck, der eine Ausgabe (entweder ein Nodeset oder eine Berechnung) für die Anzeige in der JSON-Grid-Ansicht generiert. In der Abbildung unten wird z.B. die Summe der Elemente (Items) 1 bis 4 berechnet und das Resultat (**28**) in einer separaten Zeile angezeigt. Jede Formel wird separat ausgeführt. Andere Filter oder Formeln im Dokument haben keine Auswirkung darauf.

Informationen zur Erstellung von XQuery-Ausdrücken für JSON-Dokumente finden Sie im Abschnitt XQuery-Ausdrücke für JSON 772.

Anmerkung: Sie können Ausdrücke für Filter und Formeln in der Grid-Ansicht aus Gründen der Übersichtlichkeit auch in mehreren Zeilen eingeben. Um im Ausdruck eine neue Zeile hinzuzufügen, drücken Sie **Strg+Eingabetaste**.



Wenn es sich beim Dokument um ein JSON5- oder JSONC-Dokument handelt,

- wird die Formel als JSON-Kommentar in das Dokument geschrieben.
- Optional kann auch **die Ausgabe** der Formel im JSON-Dokument als (nicht kommentierter) Code gespeichert werden. Klicken Sie dazu auf das Diskettensymbol der Formel *(siehe Abbildung unten)*. Wenn die Formel nicht als JSON-Inhalt gespeichert wird, so wird sie nur als Dokument-Metadaten gespeichert, um in der Grid-Ansicht angezeigt werden zu können. Beachten Sie, dass das Diskettensymbol nur in JSON5- und JSONC-Dokumenten und nicht in anderen JSON-Dokumenttypen zur Verfügung steht.

- Das in der Abbildung oben angezeigte JSON-Dokument einschließlich gespeicherter Formeln und Formelergebnis

© 2016-2022 Altova GmbH

*Beachten Sie, dass die **Formel** als JSON-Kommentar, das **Ergebnis der Formel** jedoch direkt im JSON-Code gespeichert wird. Der unten stehende Code (dessen Formelergebnis gespeichert wurde) ist das Resultat, wenn Sie auf das Diskettensymbol der Formel geklickt haben.*

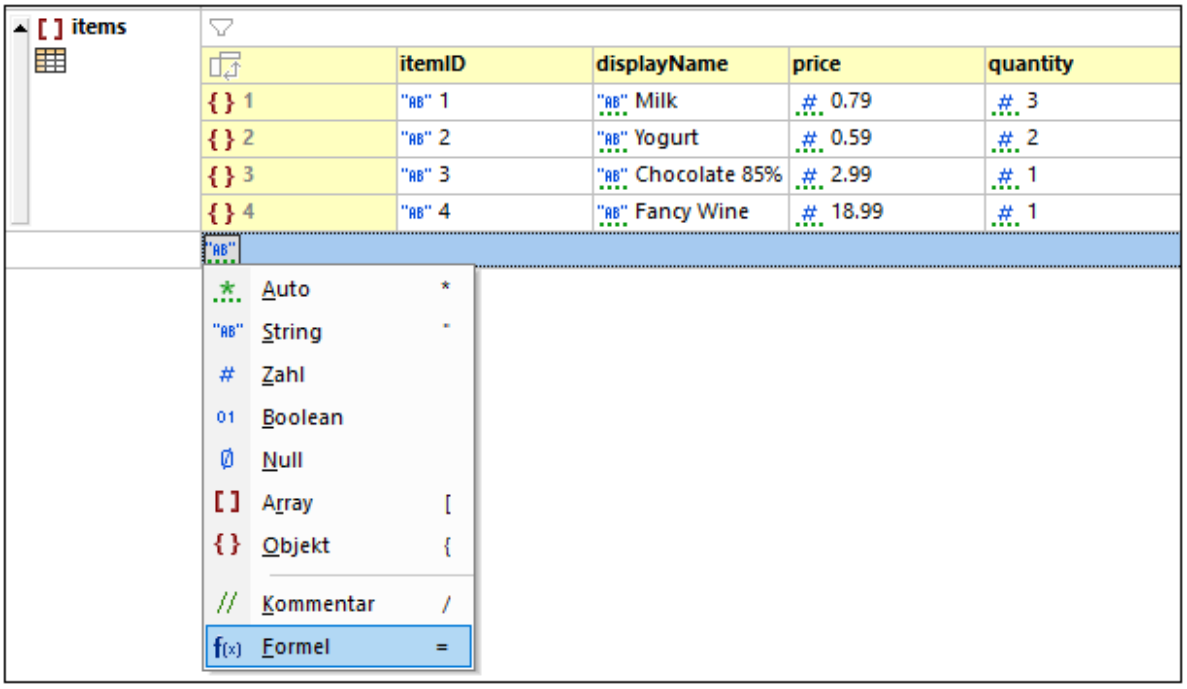
```
{
	"receiptID": "123-456-7890",
	"paymentMethod": "Cash",
	"items": [
		{
			"itemID": "1",
			"displayName": "Milk",
			"price": 1,
			"quantity": 3
		},
		{
			"itemID": "2",
			"displayName": "Yogurt",
			"price": 2,
			"quantity": 2
		},
		{
			"itemID": "3",
			"displayName": "Chocolate 85%",
			"price": 1,
			"quantity": 1
		},
		{
			"itemID": "4",
			"displayName": "Fancy Wine",
			"price": 20,
			"quantity": 1
		}
	],
	//(:altova_xq_embed:)totalPrice(:altova_xq_key:)sum(for $item in ?items?* return
$item?price * $item?quantity)
	"totalPrice": 28
	//(:altova_xq_end:)
}
```

In der obigen Abbildung werden mit der Formel die Mitglieder einer Sequenz addiert. Bei diesen Mitgliedern handelt es such jeweils um das Produkt der Werte **price** und **quantity** der einzelnen Objekte des Array **items**. Die Iteration, um die einzelnen Objekte auszuwählen und diese jeweils der Variablen **$item** zuzuweisen, wird folgendermaßen definiert: **for $item in ?items?***. Wichtig in diesem Zusammenhang ist der Kontext-Node. Es handelt sich hierbei um den Parent Node der Formel und folglich um den Parent-Node des Node **items**. Jedes Produkt wird durch Ermittlung der Child-Nodes **price** und **quantity** des aktuellen Objekts in der Variablen **$item** und durch Multiplikation dieser beiden Werte miteinander berechnet. Die so berechneten Produkte bilden die Mitglieder der Sequenz, die anschließend addiert wird, um den Gesamtpreis zu generieren.

Erstellen einer Formel

Um eine Formel zu erstellen, gehen Sie folgendermaßen vor:

1. Wählen Sie den gewünschten Node, zu dem Sie die Formel hinzufügen möchten, entweder als gleichrangigen Node oder als Child-Node aus. Klicken Sie mit der rechten Maustaste darauf und fügen Sie den gleichrangigen Node bzw. den Child-Node hinzu. Beachten Sie bei der Entscheidung, wo die Formel angezeigt werden soll (als gleichrangiger Node oder als Child-Node), dass der Kontext-Node des XQuery-Ausdrucks der Formel den Parent-Node der Formel bildet. So ist z.B. der Kontext-Node im XQuery-Ausdruck in der Abbildung oben der Parent Node der Formel (`totalPrice`) und ihres gleichrangigen Node, nämlich des Array `items`. Um die Formel in der Abbildung oben zu erstellen, wurde an das `items`-Array ein gleichrangiger Node angehängt *(siehe Abbildung unten)*.
2. Ändern Sie den Typ des Node in "Formel" *(siehe Abbildung unten)*.



3. Doppelklicken Sie in die Zelle, die das Symbol `f(x)` enthält, und geben Sie den Namen der Formel ein *(siehe Abbildung unten)*. Wenn es sich beim Dokument um ein JSON5- oder JSONC-Dokument handelt, wird ein Diskettensymbol angezeigt. Sie können auf dieses Symbol klicken, um die Ausgabe der Formel im Dokument zu speichern.



4. Standardmäßig ist der XQuery-Ausdruck der String `'XQuery'`, d.h. die Ausgabe ist der String `XQuery` (wird in der Zelle unterhalb des Ausdrucks angezeigt). Durch Doppelklick auf die Zelle des Ausdrucks, Eingabe des gewünschten Ausdruck und Drücken der **Eingabetaste** können Sie Ihren Ausdruck definieren. Dadurch wird die Formel ausgewertet und ihr Ergebnis wird angezeigt. Anderen Formeln werden nicht geändert.
5. Wenn das Dokument mehrere Formeln enthält, klicken Sie auf dem Menübefehl **JSON | Alle erneut auswerten,** um die Ergebnisse aller Formeln zu aktualisieren. Dieser Befehl ist vor allem dann hilfreich, wenn mit Formeln im Dokument Daten nachgeschlagen werden, die sich dynamisch ändern (z.B. Wechselkurse).

Beachten Sie zu Formeln die folgenden Punkte:

© 2016-2022 Altova GmbH

- Der Kontext-Node des XQuery-Ausdrucks der Formel ist der Parent-Node des Formel-Node.
- Formeln werden in den Metadaten des Dokuments (in einer speziellen applikationsweiten JSON-Metadatendatei) gespeichert. Wenn es sich beim Dokument um ein JSON5- oder JSONC-Dokument handelt, wird die Formel zusätzlich dazu als Kommentar im JSON-Inhalt gespeichert.
- Die Ausgabe einer Formel wird in der Zelle unterhalb des XQuery-Ausdrucks der Formel angezeigt. Bei JSON5- und JSONC kann die Ausgabe durch Klick auf das Diskettensymbol neben dem XQuery-Ausdruck im Dokument gespeichert werden. Unabhängig davon, ob Sie auf das Diskettensymbol klicken oder nicht, wird die Ausgabe der Formel berechnet und in den Metadaten das Dokuments gespeichert.

	Formelausgabe wird nicht im JSON-Inhalt gespeichert; Zum Speichern anklicken. *Nur in JSON5 und JSONC.*
	Formelausgabe wird im JSON-Inhalt gespeichert. Um diesen nicht zu speichern, klicken Sie auf das Symbol. *Nur in JSON5 und JSONC.*

- Wenn es sich bei der Ausgabe um eine Berechnung handelt, die im Inhalt gespeichert wird, wird diese unter dem Namen, den Sie der Formel zugewiesen haben, als Eigenschaft gespeichert. So wird die Ausgabe etwa in der oben beschriebenen Formel `totalPrice` folgendermaßen gespeichert: `"totalPrice": 28`.
- Beachten Sie diesen Unterschied: In JSON5- und JSONC-Dokumenten werden Formeln als JSON-Kommentare gespeichert. Deren Ausgabe wird in Form von JSON-Eigenschaften gespeichert.

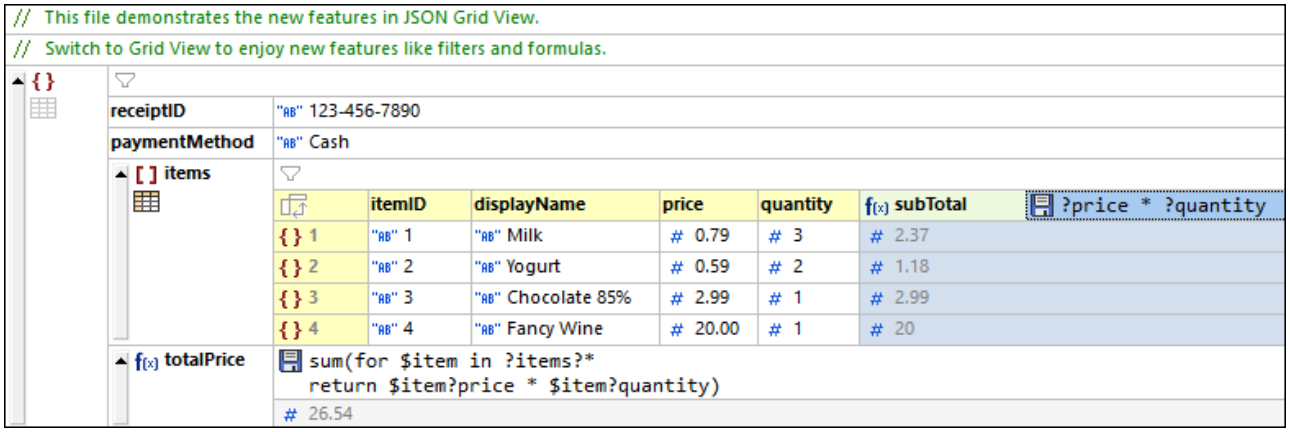
Anmerkung: Über die Einstellungen der JSON-Grid-Ansicht[721] können Sie festlegen, ob Formeln in JSON5- und JSONC-Dokumenten gespeichert werden sollen oder nicht. Standardmäßig ist die Option zum Speichern von Formeln aktiviert.

Anmerkung: Über das Dialogfeld Einstellungen für Grid-Ansicht[721] können Sie (i) Metadaten-Optionen festlegen (ob Formeln automatisch, nur auf eine Aufforderung hin oder nie gespeichert werden sollen).

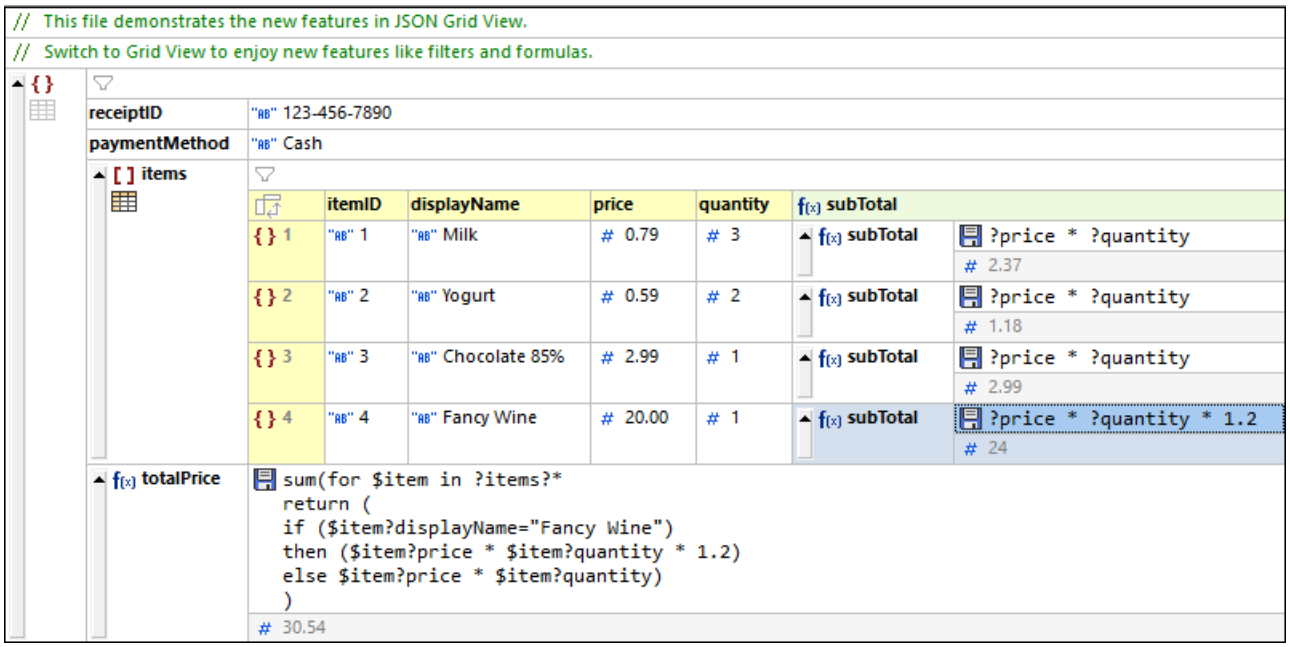
Formeln in Tabellen

Wenn alle Zellen einer Tabellenspalte (in der Tabellenansicht[704]) dieselbe Formel enthalten, so wird die Formel nur einmal angezeigt, und zwar in der Spaltenüberschrift (*siehe Abbildung unten*). Die *Berechnungsergebnisse* der Formel werden jedoch in den jeweiligen Zellen angezeigt.

Die Formel in der Spaltenüberschrift ist eine Darstellung in der Grid-Ansicht. Im Inhalt des JSON-Dokuments (in der Textansicht) wird die Formel für jedes Tabellenzeilenelement wiederholt.



Wenn sich auch nur eine einzige Formel von den anderen unterscheidet (wie in der markierten Zelle in der Abbildung unten), so wird jede Formel in ihrer dazugehörigen Zelle angezeigt.



Alle Formeln erneut auswerten

Um die Ergebnisse aller Formeln im Dokument zu aktualisieren, klicken Sie auf den Menübefehl **JSON | Alle erneut auswerten**.

Vorgangsweise für häufig verwendete Aktionen

- So fügen Sie zur Tabelle eine neue leere Spalte hinzu: Wechseln Sie von der Listenanzeige in die Tabellenansicht. Klicken Sie mit der rechten Maustaste auf ein beliebiges `Schlüssel:Wert`-Paar in der Listenanzeige und fügen Sie über das Kontextmenü des Elements mittels Einfügen oder Anhängen ein neues `Schlüssel:Wert`-Paar hinzu. Wenn Sie in die Tabellenansicht zurückwechseln, wird für das neu angehängte/eingefügte `Schlüssel:Wert`-Paar eine neue Spalte erstellt. Sie können diese Spalte nun in der Grid-Ansicht bearbeiten.

 © 2016-2022 Altova GmbH

- Wenn alle Formeln einer Tabellenspalte identisch sind, sodass die Formel in der Überschrift angezeigt wird und Sie für eine einzelne Zelle eine andere Formel erstellen möchten, wechseln Sie in die Listenanzeige und bearbeiten Sie die Formel dieser Zelle. Wenn Sie in die Tabellenansicht zurückwechseln, werden die Formeln (für alle Zellen) in den einzelnen Zellen angezeigt.
- Um in einem Ausdruck eine neue Zeile hinzuzufügen, drücken Sie **Strg+Eingabetaste**.

13.5.8 Bilder

In der JSON-Grid-Ansicht können Bilder angezeigt werden (*siehe Abbildung oben*). Das Bild wird automatisch in ein Base64-kodiertes Bild konvertiert und angezeigt.

Um ein Bild zu laden, geben Sie den folgenden Ausdruck ein: `unparsed-text("<Image-URL>", "x-binarytobase64")`

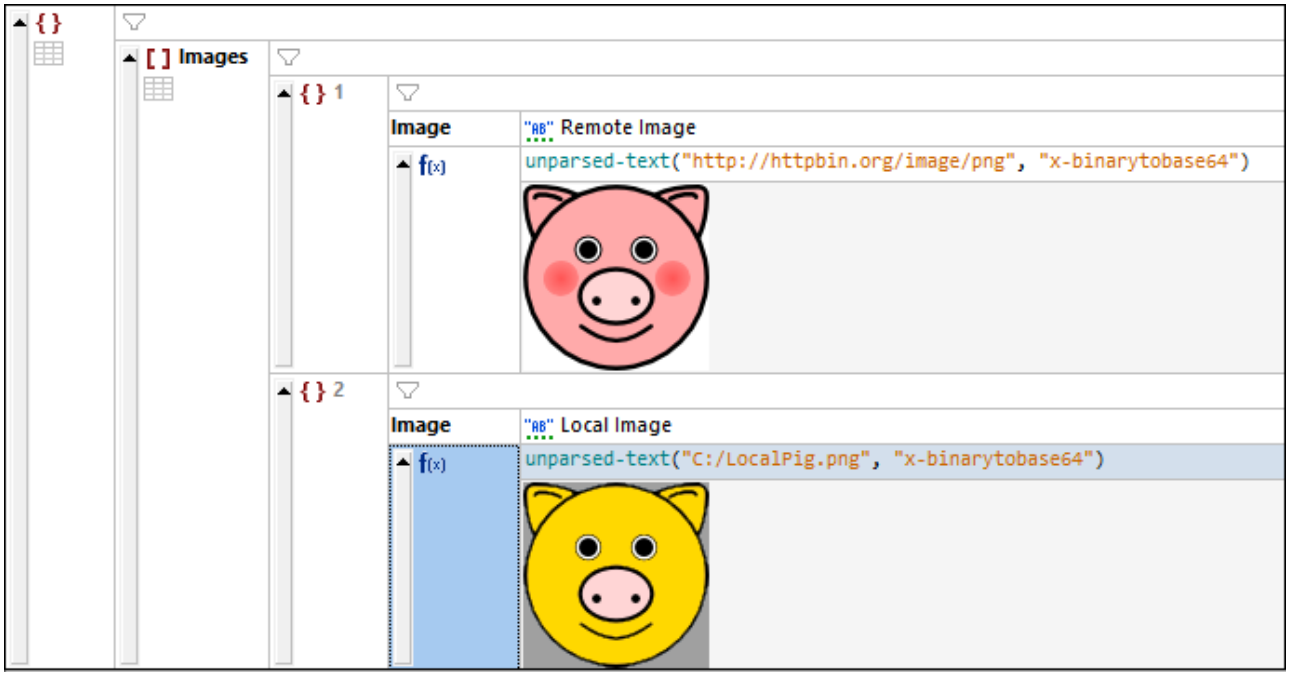


Bild-URLs können einschließlich relativer Pfade in der folgenden Form angegeben werden:

- `http://httpbin.org/image/png`
- `file:///c:/LocalPig.png`
- `C:/LocalPig.png`
- `LocalPig.png`

Speichern eines Base64-kodierten String als Bild

Um einen Base64-kodierten String in seinem Bildformat zu speichern, klicken Sie mit der rechten Maustaste auf das Bild oder dessen Zelle und wählen Sie den Befehl **Als Bild speichern**. (Beachten Sie, dass das Bild in der aktiven Datei als Base64-String kodiert ist, in der Grid-Ansicht aber als Bild angezeigt wird). Wählen Sie im daraufhin angezeigten Dialogfeld den Pfad, unter dem Sie das Bild speichern möchten und geben Sie einen Namen für die Bilddatei ein. Die Erweiterung der Bilddatei (`.png`, `.gif`, `.svg`, usw.) wird automatisch anhand

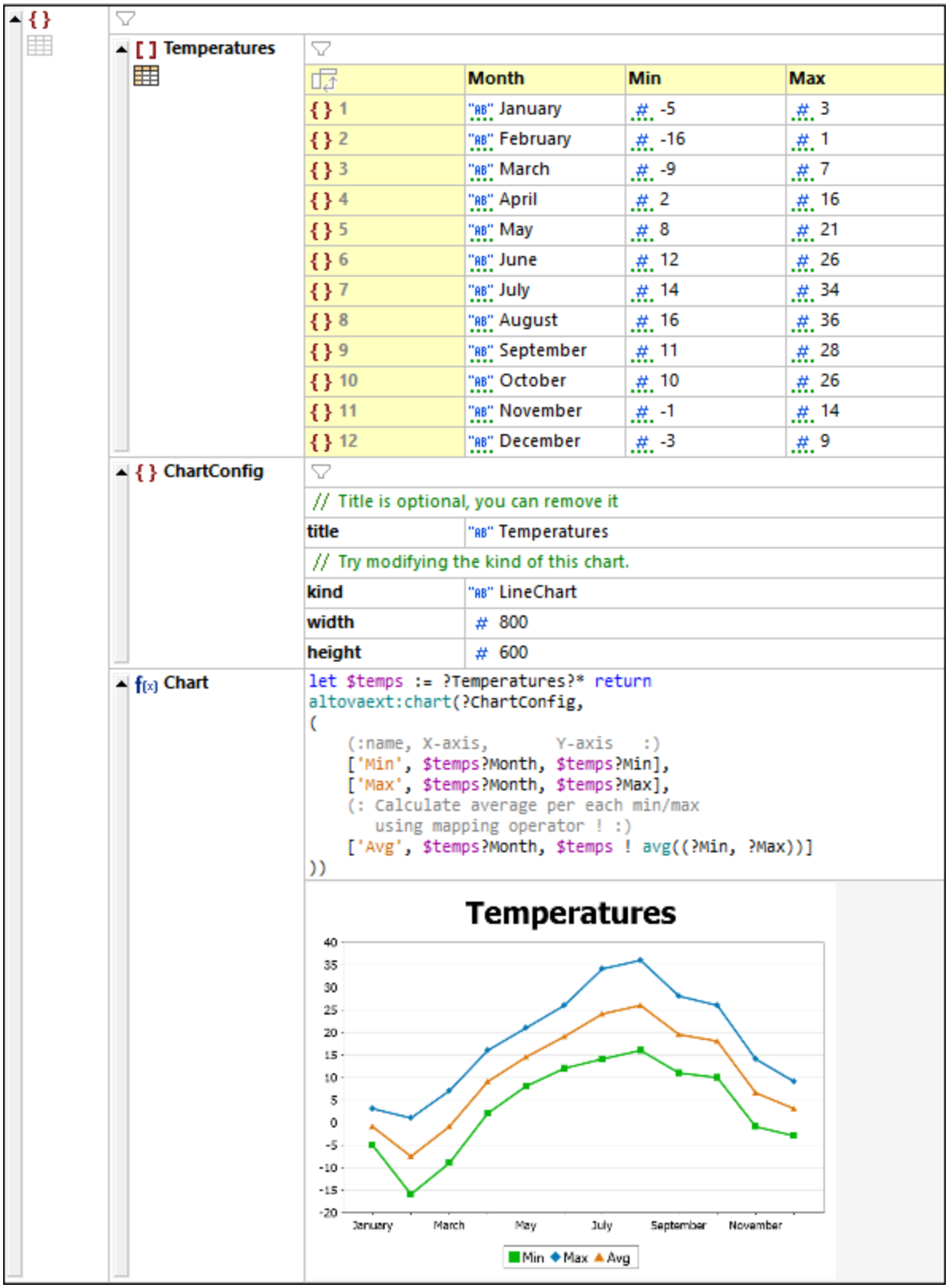
der Base64-Kodierung ermittelt und im Speichern-Dialogfeld angezeigt. Klicken Sie anschließend auf **Speichern**.

Diese Aktion kann auch über den Menübefehl **Bearbeiten | Als Bild speichern** durchgeführt werden.

13.5.9 Diagramme

In der JSON-Grid-Ansicht können mit Hilfe der Altova XPath/XQuery-Erweiterung **altovaext:chart** Diagramme erstellt werden (*siehe Abbildung unten*). Diese Erweiterung wird weiter unten beschrieben. Eine Beschreibung dazu finden Sie zusammen mit anderen Diagrammerweiterungen im Abschnitt Diagrammfunktionen[2223]. Die in der Abbildung unten gezeigte **altovaext:chart**-Erweiterungsfunktion wird in einem XQuery-`Let`-Ausdruck, der innerhalb einer Formel in der JSON-Grid-Ansicht[710] definiert wird, verwendet. Das Diagramm wird unterhalb der Formel als Bild angezeigt. (Das Diagrammbeispiel befindet sich in der Datei **Chart.jsonc**, die sich im *Ordner "Examples"* Ihres Ordners (Eigene) Dokumente[34] befindet und sich auch über das *Projekt "Examples"*[118] aufrufen lässt.)

© 2016-2022 Altova GmbH



Temperatures

	Month	Min	Max
{ } 1	"AB" January	# -5	# 3
{ } 2	"AB" February	# -16	# 1
{ } 3	"AB" March	# -9	# 7
{ } 4	"AB" April	# 2	# 16
{ } 5	"AB" May	# 8	# 21
{ } 6	"AB" June	# 12	# 26
{ } 7	"AB" July	# 14	# 34
{ } 8	"AB" August	# 16	# 36
{ } 9	"AB" September	# 11	# 28
{ } 10	"AB" October	# 10	# 26
{ } 11	"AB" November	# -1	# 14
{ } 12	"AB" December	# -3	# 9

{ } ChartConfig

```
// Title is optional, you can remove it
title  "AB" Temperatures
// Try modifying the kind of this chart.
kind   "AB" LineChart
width  # 800
height # 600
```

f(x) Chart

```
let $temps := ?Temperatures?* return
altovaext:chart(?ChartConfig,
(
    (:name, X-axis,       Y-axis   :)
    ['Min', $temps?Month, $temps?Min],
    ['Max', $temps?Month, $temps?Max],
    (: Calculate average per each min/max
       using mapping operator ! :)
    ['Avg', $temps?Month, $temps ! avg((?Min, ?Max))]
))
```

Codefragment eines XQuery-Ausdrucks zur Generierung des Diagramms in der Abbildung oben

```
let $temps := ?Temperatures?* return
altovaext:chart(?ChartConfig,
(
    (:name, X-axis,       Y-axis   :)
    ['Min', $temps?Month, $temps?Min],
    ['Max', $temps?Month, $temps?Max],
    (: Calculate average per each min/max
       using mapping operator ! :)
    ['Avg', $temps?Month, $temps ! avg((?Min, ?Max))]
))
```

Verwendung der Altova Diagramm-Erweiterung

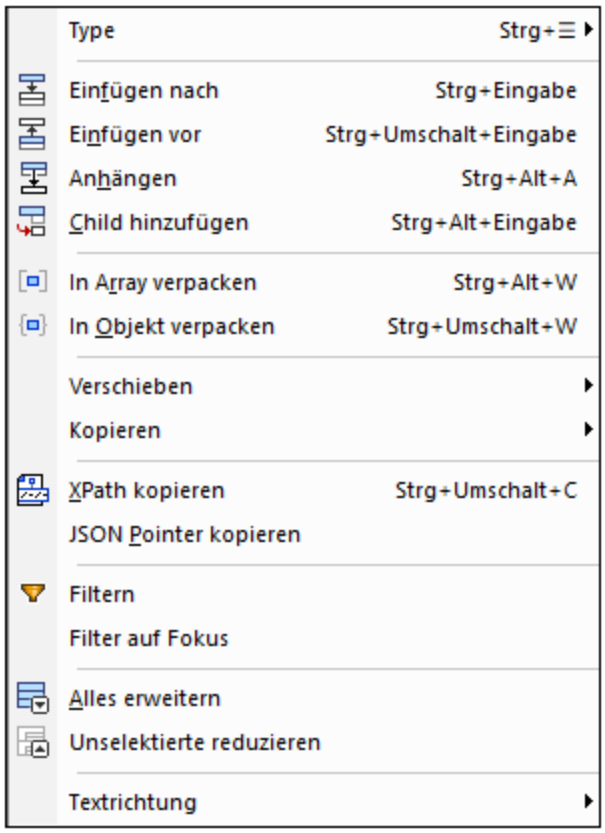
- In der Diagramm-Erweiterungsfunktion `altovaext:chart` muss das Namespace-Präfix `altovaext:` verwendet werden.
- Die Funktion `altovaext:chart` erhält zwei Argumente: (i) Diagrammkonfigurationsinformationen und (ii) Informationen über die Diagrammdatenreihe.
- Die Diagrammkonfigurationsinformationen bilden das erste Argument von `altovaext:chart`. Es handelt sich hierbei um eine nicht geordnete Reihe von vier Schlüssel-Wert-Paaren. Diese Paare sind für (i) den Titel des Diagramms (der Schlüssel ist `title`), (ii) die Art des Diagramms, wie z.B. Kreisdiagramm, Liniendiagramm, usw. (der Schlüssel ist `kind`; verfügbare Arten finden Sie unter Diagrammfunktionen[2223]), (iii) die Diagrammbreite in Pixel (nur Ganzzahlen; der Schlüssel ist `width`) und (iv) die Diagrammhöhe in Pixel (nur Ganzzahlen; der Schlüssel ist `height`). Im Beispiel in der Abbildung oben sind die Konfigurationsinformationen im Objekt `ChartConfig` gespeichert, das in der Funktion `altovaext:chart` referenziert wird.
- Die Diagrammdatenreihe bildet das zweite Argument von `altovaext:chart`. Jede Datenreihe ist ein Array der Größe 3: (i) der Name der Datenreihe, (ii) die Werte der x-Achse, (iii) die Werte der y-Achse. Wenn Sie mehrere Datenreihen erstellen möchten (z.B. solche, bei denen jede Datenreihe eine Linie bildet, wie im Beispiel oben), erstellen Sie eine Sequenz mehrerer Arrays. Das Beispiel oben enthält drei Datenreihen; für die Temperaturminima, -maxima und die Durchschnittstemperaturen. Die Daten für die x- und y-Achse werden über das Array namens `Temperatures` referenziert.

Nähere Informationen dazu finden Sie im Abschnitt Diagrammfunktionen[2223].

13.5.10 Kontextmenü in der Grid-Ansicht

Wenn Sie in der JSON-Grid-Ansicht mit der rechten Maustaste auf eine Komponente klicken, wird ein Kontextmenü (*Abbildung unten*) mit Befehlen zum Ändern der Anzeige und Bearbeiten des Dokuments angezeigt. Das Kontextmenü kann auch durch Drücken der Menütaste aufgerufen werden. Eine Beschreibung dieser Befehle finden Sie weiter unten.

© 2016-2022 Altova GmbH



Typ
Wenn Sie den Mauszeiger über den Befehl **Typ** platzieren, wird ein Untermenü angezeigt, in dem Sie den Typ der Komponente 699 auswählen können. Sie können das Typmenü auch über das Tastenkürzel **Strg+Menütaste** aufrufen.

Einfügen nach/vor, Anhängen, Child hinzufügen
Mit den Befehlen **Einfügen** und **Anhängen** fügen Sie ein Element auf derselben Ebene hinzu. Das neue Element wird im Fall von **Einfügen vor** und **Einfüge nach** vor bzw. nach dem ausgewählten Element und im Fall von **Anhängen** als letztes gleichrangiges Element des ausgewählten Elements hinzugefügt. Mit dem Befehl **Child hinzufügen** wird ein neues Element als Child-Element angehängt. Wenn bereits ein oder mehrere Child-Elemente vorhanden sind, erhält das neue Element denselben Typ wie das letzte Child-Element. Wenn kein Child-Element vorhanden ist, ist das neue Element ein leeres `Schlüssel:Wert`-Paar. Wenn es sich beim ausgewählten Element z.B. um ein Array mit Objekt-Child-Elementen handelt, so wird das neue Child-Element als das letzte Child-Objekt angehängt.

In Array verpacken, in Objekt verpacken
Der ausgewählte Teil der Tabelle kann entweder in ein Array oder ein Objekt verpackt werden.

Nach oben/nach unten/höher stufen/tiefer stufen
Eine Komponente kann im Raster nach oben, unten, links oder rechts verschoben werden. In diesem Fall sind die entsprechende Befehle aktiv. Wählen Sie den entsprechenden Befehl aus, um die Komponente zu verschieben.

Kopieren
Mit Hilfe dieser Befehle kann die aktuelle Auswahl in die Zwischenablage kopiert werden. Es sind nur diejenige Optionen aktiv, die für diese Komponente zulässig sind.

Befehl	Beschreibung
Als JSON-Text kopieren	Die aktuelle Auswahl wird als JSON-Markup-Code serialisiert.
Als tabulatorgetrennten Text kopieren	Die aktuelle Tabellenauswahl wird als TSV (Tab Separated Values = tabulatorgetrennte Werte) serialisiert.
Als Bild kopieren	Die aktuelle Bildzelle wird als Bild kopiert.

Beachten Sie die folgenden Punkte:

- *Als Bild kopieren* kopiert den als Base64-kodierten String des ausgewählten Bilds. Wenn der String in ein Dokument eingefügt wird, in dem der Base-64-kodierte String als Bild dargestellt werden kann (z.B. in eine andere Tabellenzelle in der Tabellenansicht), so wird er als Bild dargestellt. Anderfalls wird er als String eingefügt.
- Um Text in eine Zelle einzufügen, kopieren Sie den Text und fügen Sie ihn in die Zelle ein.
- Um ein Bild, aus einer Datei zu importieren, verwenden Sie den Befehl **Bearbeiten | Einfügen | Kodierte externe Datei**[1301]. Dadurch wird ein Bild als Base64-kodierter String in das Dokument eingefügt und in der JSON-Grid-Ansicht als Bild angezeigt. Für den Import werden die folgenden Bilddateiformate unterstützt: `PNG`, `JPEG`, `BMP`, `GIF`, `TIFF`.

XPath kopieren
Dieser Befehl kopiert einen XPath 3,1 Locator-Ausdruck in die Zwischenablage. Der Ausdruck beginnt an der Dokument-Root, unter der sich der ausgewählte Node befindet.

JSON Pointer kopieren
Dieser Befehl kopiert einen JSON Pointer-Ausdruck zum Aufrufen des ausgewählten Nodes in die Zwischenablage. Beispiel: **`/Artists/1/Albums/1/Tracks`**. Nähere Informationen dazu finden Sie unter **XPointer/JSON-Pointer kopieren**[1301].

Filter, Filter auf Fokus
Der **Filter**-Befehl ist ein Ein/Aus-Befehl. Dieser Befehl fügt einen Filter zum ausgewählten Objekt hinzu oder deaktivert einen Filter. Beachten Sie, dass Sie, nachdem Sie mit diesem Befehl einen Filter hinzugefügt haben, einen Filterausdruck eingeben müssen, da der Filter sonst automatisch entfernt wird. Nachdem ein Filter hinzugefügt wurde, können Sie ihn durch erneute Auswahl des Befehls oder Klick auf den Filter deaktivieren. Um den Filter zu entfernen, klicken Sie in der Symbolleiste auf *Alle Filter entfernen*.

Der Befehl **Filter auf Fokus** (i) erstellt am Root-Element, das einen XPath-Ausdruck zur Suche des ausgewählten Node enthält, einen Filter und (ii) filtert die Anzeige der Grid-Ansicht, um den ausgewählten Node (und dessen Nachfahren) direkt unterhalb des Root-Elements anzuzeigen. Dies ist nützlich, wenn nur der ausgewählte Node angezeigt werden soll. Um den Filter zu entfernen, klicken Sie in der Inhaltszelle des Root-Elements auf das Filtersymbol.

Alles erweitern
Dieser Befehl ist aktiviert, wenn die ausgewählte Komponente oder eine ihrer untergeordneten Komponenten reduziert angezeigt wird. Die Komponente und alle ihre untergeordneten Komponenten werden dadurch erweitert angezeigt.

© 2016-2022 Altova GmbH

Unselektierte reduzieren
Mit diesem Befehl werden alle Komponenten mit Ausnahme der ausgewählten Komponente und ihrer Vorfahren reduziert.

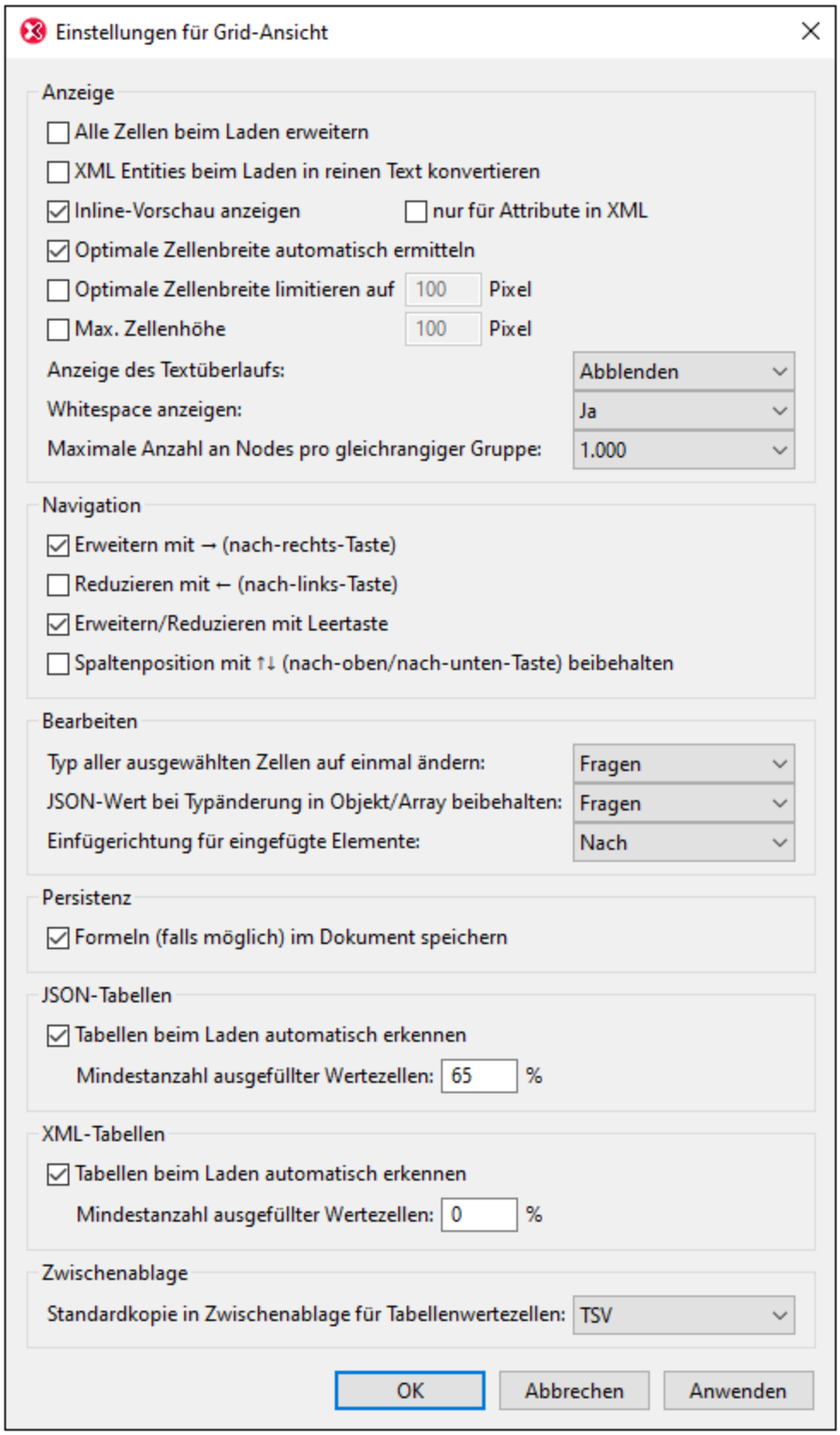
Textrichtung
Dieser Befehl steht für Nodes, die einen String-Typ enthalten, zur Verfügung. Sie wechseln damit die Leserichtung, damit in der Zelle beginnend entweder von links oder von rechts gelesen wird. Dies ist bei Verwendung von Sprachen wie Arabisch oder Hebräisch nützlich.

13.5.11 Einstellungen der Grid-Ansicht

Die Einstellungen der Grid-Ansicht werden im Dialogfeld "Einstellungen für Grid-Ansicht" *(Abbildung unten)* definiert, welches in der Grid-Ansichts-Symbolleiste[692] über die Schaltfläche "Einstellungen" aufgerufen wird..

	Schaltfläche "Einstellungen für Grid-Ansicht) in der Grid-Ansichts-Symbolleiste[692].

Weiter unten finden Sie eine Beschreibung der Einstellungen der Grid-Ansicht. Beachten Sie, dass diese Einstellungen für die Grid-Ansicht aller Dokumente (XML, JSON, DTD) gelten.

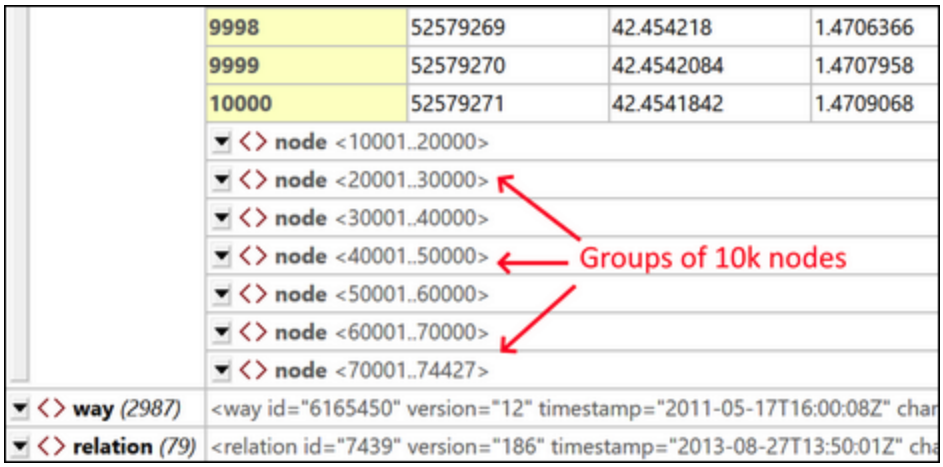


end

© 2016-2022 Altova GmbH

Anzeige
Die Kontrollkästchen im Abschnitt *Anzeige* sind selbsterklärend. Im Folgenden finden Sie einige klarstellende Anmerkungen.

- Wenn beim Laden nicht alle Zelle erweitert werden, werden der Root-Node und alle seine Nachfahren reduziert angezeigt. Sie müssen dann jeden Node erweitern, wenn Sie tiefer in die Hierarchie eintauchen möchten.
- Wenn die Option *XML-Entities beim Laden in reinen Text konvertieren* aktiv ist, werden XML-Entities in der Grid-Ansicht als reiner Text der jeweiligen Entity geladen und nicht zu ihrer jeweiligen Darstellung als Glyph aufgelöst.
- Wenn *Inline-Vorschau anzeigen* nicht aktiviert ist, wird anstelle der Vorschau der Zelle nur die Indexnummer des Elements in der Zelle angezeigt. Wenn die Inline-Vorschau aktiviert ist, haben Sie die Wahl zwischen einer Vorschau, die (i) sowohl den Elementinhalt als auch Attribute enthält oder (ii) einer, die nur Attribute enthält. Wenn Sie sich für zweiteres entscheiden, aktivieren Sie *nur für Attribute...*; wenn Sie ersteres möchten, deaktivieren Sie *nur für Attribute*. Beachten Sie, dass nur der erste Teil des Inline-Inhalts einer Zelle angezeigt wird; um den gesamten Inhalt zu sehen, platzieren Sie den Mauszeiger über den Start-Tag eines Elements.
- Wenn die optimale Breite aktiviert wurde, wird die gesamte Breite des Rasters angezeigt. Zu diesem Zweck wird Text in einigen Zellen umbrochen.
- Wenn Text über die Breite einer Zelle hinausragt, kann dies entweder durch heller werdenden Text oder Auslassungspunkte gekennzeichnet werden.
- Sie können die Anzeige von Whitespaces in Rasterzellen ein- oder ausschalten. Ein Leerzeichen wird in Form eines vertikal zentrierten Punkts und ein Tabulatorzeichen als Pfeil angezeigt. Ein Zeilenendezeichen wird als Wagenrücklauf in der Zelle angezeigt.
- Gleichrangige Nodes können in gleichrangigen Gruppen von 100, 1.000 oder 10.000 Nodes *(siehe Abbildung unten)* angezeigt werden. Dies ist aus zwei Gründen nützlich: (i) Wenn Sie in Dateien mit vielen gleichrangigen Nodes Platz für die Anzeige sparen möchten, können Sie gleichrangige Nodes in großen Gruppen zusammenfassen; (ii) Wenn Sie nur eine Teilmenge der gleichrangigen Nodes verarbeiten möchte (und somit Verarbeitungszeit sparen wollen), können Sie die gewünschten gleichrangigen Gruppen, die verarbeitet werden sollen, erweitern und die Verarbeitungsaktion nur an diesen Nodes ausführen; z.B. wenn Sie eine Suchen-und-Ersetzen-Aktion durchführen möchten oder den Typ eines Node konvertieren möchten (siehe *Bearbeiten* weiter unten).



Navigation
Sie können mit Hilfe der Pfeiltasten durch das Raster navigieren. Diese Einstellungen bieten clevere Optionen für die Verwendung der Tasten.

- *Erweitern mit nach-rechts-Taste:* Wenn eine Zelle reduziert angezeigt wird, können Sie den Eintrag in der Zelle mit Hilfe der *nach-rechts*-Taste erweitern. Falls der Zelleneintrag nicht reduziert angezeigt wird, gelangen Sie mit Hilfe der *nach-rechts*-Taste zur nächsten rechts davon gelegenen Zelle (dies kann auch eine Child-Zelle sein, wenn die nächste rechts gelegene Zelle ein Child ist). Wenn diese Option nicht aktiviert ist, halten Sie mit der *nach-rechts*-Taste bei einer reduziert angezeigten Zelle an. Beachten Sie, dass die Funktion *Erweitern mit nach-rechts-Taste* nicht auf Zellen innerhalb von Tabellen angewendet wird; in Tabellenzellen gelangen Sie damit zur nächsten rechts davon gelegenen Zelle.
- *Reduzieren mit nach-links-Taste:* Wenn Sie mit Hilfe der *nach-links*-Taste nach links gehen, müssen Sie sich in der Dokumenthierarchie irgendwann nach oben bewegen. Wenn diese Option aktiviert ist, werden Elemente, die reduziert werden können, bei Drücken der *nach-links*-Taste reduziert; andernfalls werden solche Elemente nicht reduziert, obwohl sich der Fokus auf das übergeordnete Element verschiebt. Beachten Sie, dass die Funktion *Reduzieren mit nach-links-Taste* nicht auf Zellen innerhalb von Tabellen angewendet wird; in Tabellenzellen gelangen Sie damit zur nächsten links davon gelegenen Zelle.
- *Erweitern/Reduzieren mit Leertaste:* Die Leertaste fungiert als Ein/Aus-Befehl zum Erweitern/Reduzieren eines Elements. Sie kann somit als zusätzliche Hilfstaste beim Navigieren im Raster verwendet werden.
- *Spaltenposition mit nach-oben/nach-unten-Taste beibehalten:* Mit Hilfe der *nach-oben*- und *nach-unten*-Taste können Sie sich durch die Zellen des Rasters nach oben bzw. nach unten bewegen - auch durch übergeordnete und untergeordnete Elemente, die sich auf verschiedenen Ebenen und daher in unterschiedlichen Spalten befinden. Wenn diese Option aktiviert ist, werden Ebenen, die in anderen als der aktuellen Spalte dargestellt werden, übersprungen. Dies funktioniert, z.B. folgendermaßen: Angenommen, der Cursor befindet sich in der Spalte für das Element **`subject/course/books/book/title`**. Wenn die Option *Spaltenposition beibehalten* aktiv ist, können Sie mit Hilfe der *nach-oben*- und *nach-unten*-Pfeiltaste nur durch die Titel der Bücher navigieren (ohne in die Spalten **`book`**, **`books`**, **`course`** oder **`subject`** oder Spalten für untergeordnete Elemente von **`Title`** zu gelangen).

Bearbeiten

Die Kontrollkästchen im Abschnitt *Anzeige* sind selbsterklärend. Im Folgenden finden Sie einige klarstellende Anmerkungen.

- Wenn Sie den Typ mehrerer ausgewählter Zellen ändern, stehen die folgenden Optionen dazu, wie Sie damit fortfahren möchten, zur Verfügung: *Immer, Nie* oder *Fragen* (für die Entscheidung durch den Benutzer).
- Wenn Sie einen JSON-Typ von einem atomaren Typ in ein Objekt oder Array oder umgekehrt ändern, stehen die folgenden Optionen dazu, wie Sie damit fortfahren möchten, zur Verfügung: (i) *Fragen* (ob der Wert des atomaren Typs als Wert eines unbenannten Child-**`Schlüssel:Wert`**-Paars beibehalten werden soll oder ob er verworfen werden soll); (ii) *Immer*: (Der Wert in einem nicht benannten Child-**`Schlüssel:Wert`**-Paar soll immer beibehalten werden); (iii) *Nie*: (Der Wert soll nie beibehalten werden, d.h. der Wert soll verworfen werden).
- Mit der Option *Einfügerichtung* wird festgelegt, ob eine Auswahl in der Zwischenlage oberhalb oder unterhalb der ausgewählten Zelle eingefügt werden soll.

Persistenz

Filter und Formeln können in den Metadaten von XML- und JSON-Dokumenten gespeichert werden. Formeln werden in XML-Dokumenten als Processing Instructions und in einigen Arten von JSON-Dokumenten als Kommentare gespeichert. Mit diesen Einstellungen wird festgelegt, wie beim Speichern von Filtern und Formeln verfahren werden soll. Im Fall von Formeln bezieht sich "*wenn möglich*" darauf, dass Formeln nicht in allen

 © 2016-2022 Altova GmbH

JSON-Dokumenten (sondern nur in JSON5- und JSONC-Dokumenten) als Kommentare gespeichert werden können.

JSON-Tabellen, XML-Tabellen
Wenn die Einstellung zum automatischen Erkennen von Tabellen der-Grid-Ansicht beim Laden aktiviert wurde, können Sie auswählen, wie viele Tabellenzellen mindestens ausgefüllt sein müssen, damit Tabellen als solche erkannt werden. Wenn die Anzahl der ausgefüllten Tabellenzellen diesen Wert nicht übersteigt, wird die Struktur als normales Raster angezeigt, wobei sich wiederholende Elemente nacheinander aufgelistet werden.

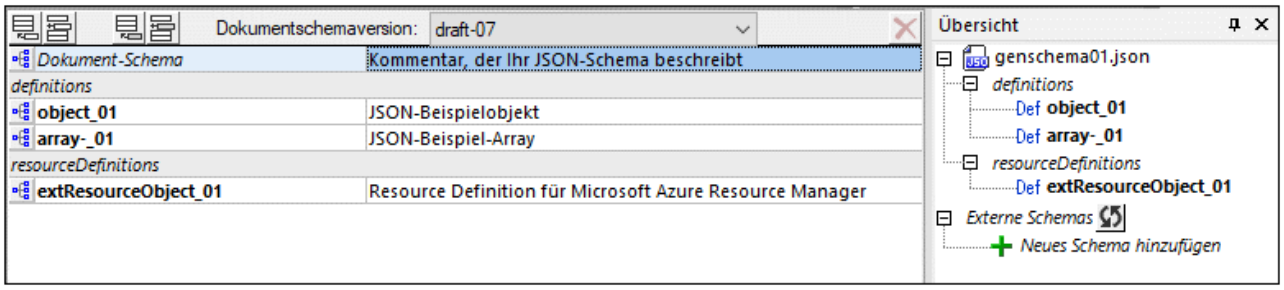
Zwischenablage
Sie können auch auswählen, ob der Inhalt der Zwischenablage als tabellengetrennte Werte (TSV = tab-separated values) oder (je nach Dokumenttyp) als XML/JSON gespeichert werden soll. Dies ist eine sehr nützliche Funktion. Wenn Sie eine Tabelle aus der Zwischenablage in ein anderes Dokument kopieren möchten, können Sie mit dieser Einstellung auswählen, ob die kopierte Tabelle als TSV oder mit Markup-Code gespeichert werden soll. (Um den Unterschied zu sehen, fügen Sie eine Tabelle in einen Text-Editor ein, nachdem Sie diese in jedem dieser Formate in die Zwischenablage kopiert haben.)

13.6 JSON-Schema-Ansicht

In der JSON-Schema-Ansicht können Sie JSON-Schema-Dokumente anzeigen und bearbeiten. Das Fenster der JSON-Schema-Ansicht ist in die folgenden Bereiche gegliedert:

- ein Hauptfenster, in dem Sie zwischen einem **Definitionsübersichtsraster**[729] und einer **Design-Ansicht**[735] wechseln können.
- drei Eingabehilfefenster[730] (standardmäßig rechts vom Hauptfenster): Übersicht, Details und Constraints
- ein Fenster "Meldungen" (standardmäßig unterhalb des Hauptfensters)
- ein Info-Fenster[765] (standardmäßig am linken unteren Rand des Applikationsfensters)

In der Abbildung unten sehen Sie das Hauptfenster und die Eingabehilfe "Übersicht".

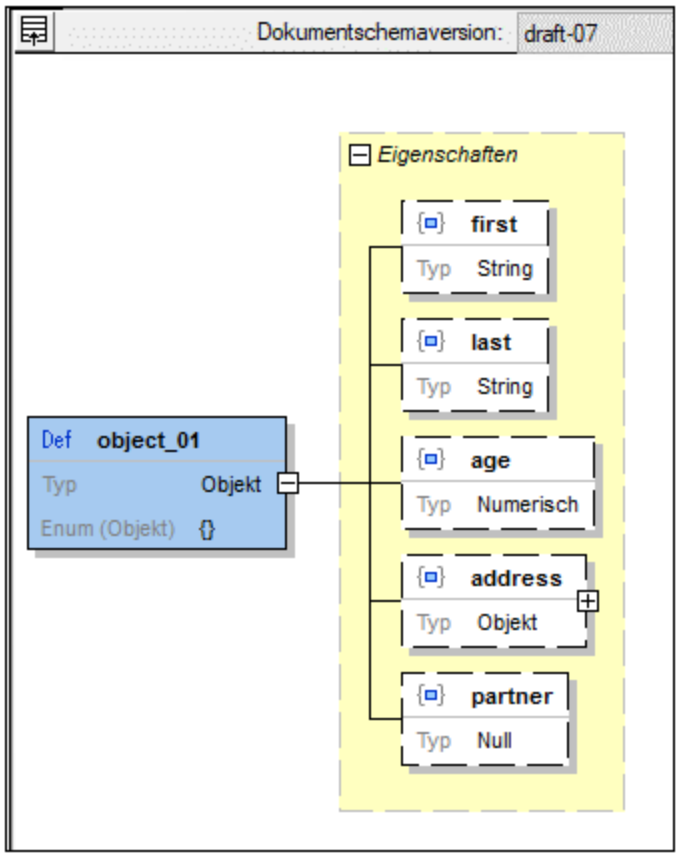


Das Hauptfenster

Im Hauptfenster können Sie zwischen einem Definitionsübersichtsraster[729] (*siehe Abbildung oben*) und einer Design-Ansicht[735] (*Abbildung unten*) wechseln. Im Definitionsübersichtsraster sehen Sie das als *Dokumentschema* aufgelistete Hauptschema des aktuellen Dokuments plus alle Definitionen, die Sie zum Schema hinzugefügt haben. (Eine Definition ist eine Beschreibung einer JSON-Datenstruktur. In der Abbildung oben sind `object_01` und `array_01` Definitionen, und zwar von einem Objekt bzw. einem Array.) Die Definitionen werden auch in der Eingabehilfe "Übersicht" aufgelistet (*siehe Abbildung oben*).

Während Sie im Definitionsübersichtsraster eine Ansicht der obersten Ebenen des JSON-Schemas sehen, wird hier nicht angezeigt, was sich innerhalb der einzelnen in der Übersicht angezeigten Definitionen befindet. Um eine Definition in der Design-Ansicht (*Abbildung unten*) anzuzeigen und zu bearbeiten, klicken Sie auf das Symbol der Definition (*siehe Abbildung oben*) oder doppelklicken Sie in der Eingabehilfe "Übersicht" (*siehe Abbildung oben)* auf die Definition.

© 2016-2022 Altova GmbH



Um von der Design-Ansicht wieder in das Definitionsübersichtsraster zu gelangen, klicken Sie links oben in der Design-Ansicht auf das Symbol **Zum Definitionsraster wechseln** (*siehe Abbildung oben*). Um die Design-Ansicht zu konfigurieren, klicken Sie auf den Menübefehl **Schema-Design | Ansicht konfigurieren** 761.

Die Eingabehilfen

Beide Modi der Schema-Ansicht (Definitionsübersichtsraster und Design-Ansicht) verfügen über drei Eingabehilfen: Übersicht, Details und Constraints. Diese Eingabehilfen bieten Funktionen, um (i) Informationen über das Schema und seine Definitionen anzuzeigen und (ii) Informationen und Werte in Zusammenhang mit Definitionen einzugeben. Eine ausführliche Beschreibung dazu finden Sie im Abschnitt Eingabehilfen: Übersicht, Details, Contstraints 730.

13.6.1 JSON-Schema-Version

Das Schema ist in JSON-Syntax geschrieben und normalerweise in einer Datei mit der Erweiterung `.json` gespeichert. Durch das Schlüsselwort **`$schema`**, das das erste Schlüsselwort des Objekts der obersten Ebene sein und als Wert die URI der gewünschten JSON-Schema-Version haben sollte, ist es als JSON-Schema gekennzeichnet. Hier sehen Sie zwei Beispiele dafür, wie das Schlüsselwort **`$schema`** verwendet wird:

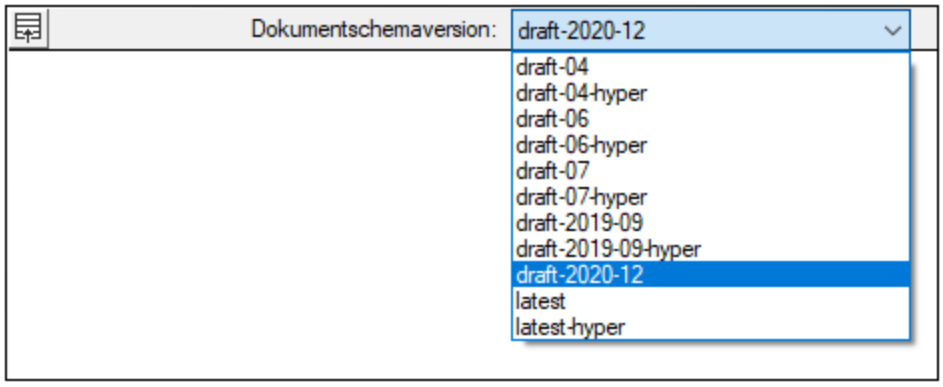
```
{
   "$schema": "http://json-schema.org/draft-07/schema#",
```

```
    ...
}
```

```
{
   "$schema": "http://json-schema.org/schema#",
   ...
}
```

Im ersten Beispiel oben wird die Schema-Version explizit benannt. Anstatt explizit eine Version auszuwählen, können Sie wie im zweiten Beispiel oben `"http://json-schema.org/schema#"` verwenden. Damit wird angegeben, dass die zu verwendende Schema-Version die neueste Version ist (derzeit **`draft-07`**).

Sie können in der JSON-Schema-Ansicht die Version in der Auswahlliste am oberen Rand des Hauptfensters ändern *(siehe Abbildung unten)*.



Funktionen von neuen Schemaversionen, die in einer älteren Version nicht definiert sind

Wenn Sie eine Funktionalität aus einer neueren Schemaversion verwenden und anschließend zu einer älteren Version, die diese Funktionalität nicht unterstützt, wechseln, geschieht Folgendes:

- Es wird eine Meldung angezeigt, in der Sie gefragt werden, ob Sie die Funktion entfernen/konvertieren oder beibehalten möchten.
- Wenn Sie die neue Funktion beibehalten, wird die entsprechende Komponente für diese Funktionalität bzw. das entsprechende Detail in orangem Text angezeigt. Wenn z.B. für das Schlüsselwort **`const`** (neu in **`draft-06`**) ein Wert definiert wurde und Sie die Schemaversion in **`draft-04`** ändern, wird der Wert des Schlüsselworts **`const`** orange angezeigt.

JSON-Schema-Versionen

Informationen zur JSON-Schema-Spezifikationen, insbesondere zu zusätzlichen Funktionalitäten in der jeweiligen Version finden Sie unter den nachstehenden Links:

JSON-Schema-Spezifikation: http://json-schema.org/specification.html
Anmerkungen zur Draft-06 Release: http://json-schema.org/draft-06/json-schema-release-notes.html
Anmerkungen zur Draft-07 Release: http://json-schema.org/draft-07/json-schema-release-notes.html
Draft 2019-09 (früher Draft-08): http://json-schema.org/specification-links.html#2019-09-formerly-known-as-draft-8

© 2016-2022 Altova GmbH

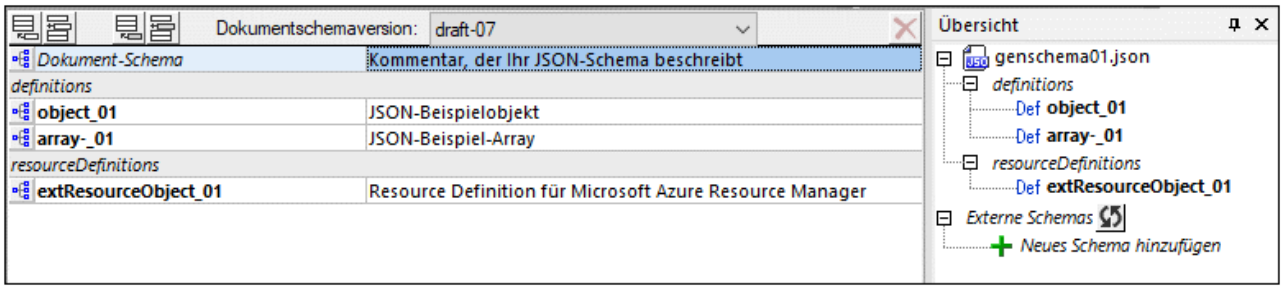
Draft 2020-12: http://json-schema.org/specification-links.html#2020-12

Zusätzlich dazu stehen die folgenden Hyper-Schemas zur Verfügung: **`draft-04-hyper`**, **`draft-06-hyper`**, **`draft-07-hyper`**, **`2019-09-hyper`**.

Links zu den Core- und Hyper-Schemas finden Sie hier: http://json-schema.org/specification-links.html.

13.6.2 Hinzufügen von globalen Definitionen

Im **Definitionsübersichtsraster**[726] im Hauptfenster (*Abbildung unten*), sehen Sie eine Liste der globalen Definitionen des Schemas. Dabei handelt es sich um (i) die Definition des *Hauptdokumentschemas,* (ii) Definitionen (oder in späteren Schemas *$defs*) von *globalen JSON-Typen*, wie z.B. Objekten, Arrays, Strings usw., die JSON-Schema-Typen sind; (iii) Definitionen von externen oder benutzerdefinierten JSON-Typen; derzeit stehen nur Definitionen, die in einem Container mit dem Namen **`resourceDefinitions`** gespeichert sind, zur Verfügung; dies ist der vom Microsoft Azure Resource Manager für JSON-Definitionen verwendete Container. Über die Schaltflächen **Definitionsabschnitt anhängen** und **Definitionsabschnitt einfügen** in der Symbolleiste des Rasters können Sie einen neuen *resourceDefinitions*-Abschnitt zum Schema-Dokument hinzufügen (*siehe Abbildung unten*).

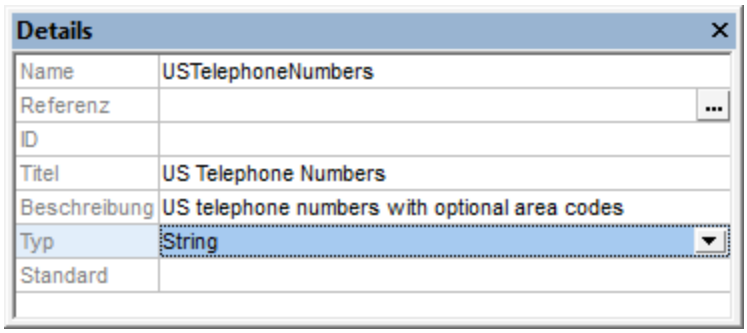


Einen JSON-Typ[683] global zu definieren, ist nützlich, wenn Sie diesen Typ im selben oder einem anderen Schema wiederverwenden möchten. So können Sie etwa in einem JSON-Schema einen JSON-String-Typ für US-Telefonnummern definieren und diese Definition anschließend nicht nur vom selben Schema aus, sondern auch von anderen JSON-Schemas aus referenzieren.

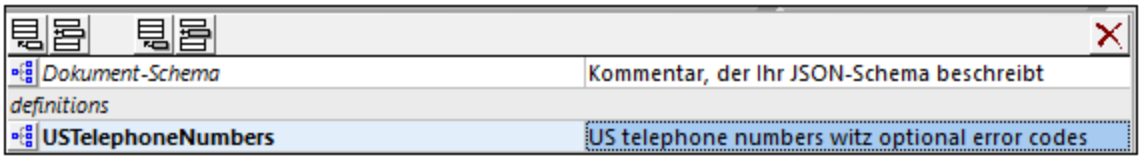
Hinzufügen einer Definition und damit in Zusammenhang stehender Aktionen

Zum Hinzufügen und Bearbeiten von Definitionen stehen im Definitionsübersichtsraster die folgenden Aktionen zur Verfügung:

- *So fügen Sie eine Definition hinzu:* Klicken Sie links oben im Definitionsübersichtsraster auf das Symbol **Definition für benanntes Schema anhängen** oder **Definition für benanntes Schema einfügen** (*siehe Abbildung oben*). Daraufhin wird im Raster an der Stelle, an der Sie die Definition anhängen oder einfügen, eine neue leere Definition mit einem Standardnamen erstellt. Die neue Definition wird auch in der Eingabehilfe "Übersicht" als *Def* aufgelistet (*siehe Abbildung oben*).
- *So ändern Sie den Typ einer Definition:* Jede neue Definition wird mit dem Typ `Beliebig` erstellt. Sie können ihren Typ in der Eingabehilfe "Details" (*siehe Abbildung unten, in der der Typ "String" lautet*) oder durch Bearbeitung der Definition in der Design-Ansicht[735] ändern.



- *So benennen Sie eine Definition um:* Doppelklicken Sie auf den Namen der Definition und bearbeiten Sie den Namen. Alternativ dazu können Sie das Feld *Name* in der Eingabehilfe "Details" [731] bearbeiten.
- *So geben Sie eine Beschreibung der Definition ein:* Bearbeiten Sie in der Eingabehilfe "Details" [731] das Feld "*Beschreibung*". Die Beschreibung wird im Definitionsübersichtsraster neben dem Namen der Definition angezeigt (*siehe Abbildung unten*). Sie können auch in das Beschreibungsfeld des Definitionsübersichtsrasters doppelklicken, um eine Beschreibung zu bearbeiten.



- *So referenzieren Sie eine Definition:* Nähere Informationen dazu finden Sie in der Beschreibung der Eingabehilfe "Übersicht" [730] und im Abschnitt Globale und lokale Definitionen [733].
- *So bearbeiten Sie eine Definition:* Klicken Sie im Definitionsübersichtsraster auf das Symbol der Definition oder doppelklicken Sie in der Eingabehilfe "Übersicht" [730] auf die Definition. Daraufhin wird die Definition in der Design-Ansicht geöffnet, wo sie bearbeitet werden kann.

13.6.3 Eingabehilfen: Übersicht, Details, Constraints

Die Eingabehilfen der JSON-Schema-Ansicht befinden sich standardmäßig auf der rechten Seite des Applikationsfensters. Sie stehen in beiden Modi des Hauptfensters [726] zur Verfügung: (i) im Definitionsübersichtsraster und (ii) in der Design-Ansicht. Sie können Eingabehilfefenster an ihrer Titelleiste an eine andere Stelle des Fensters ziehen und durch Doppelklick auf die Titelleiste einer Eingabehilfe können Sie diese ab- und andocken. Nähere Informationen dazu finden Sie im Abschnitt Eingabehilfen [121].

Eingabehilfe "Übersicht"

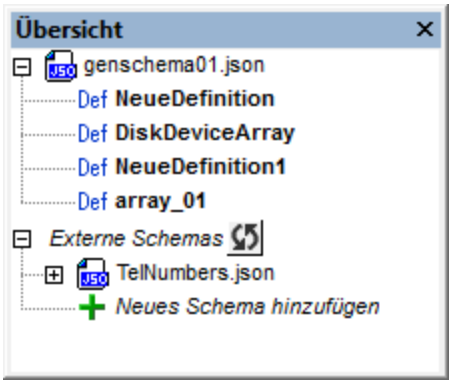
In der Eingabehilfe "Übersicht" (*Abbildung unten*) sind die aktuelle Schemadefinition sowie alle globalen Definitionen des aktuellen Schemas aufgelistet. Wenn Sie auf eine Definition doppelklicken, wird diese Definition in der Design-Ansicht [735] geöffnet, wo Sie diese bearbeiten können. Wenn Sie Definitionen aus externen Schemas verwenden möchten, so fügen Sie zuerst das externe Schema hinzu und verwenden Sie anschließend die gewünschte Definition wieder.

Hinzufügen des externen Schemas

Fügen Sie das externe Schema hinzu, indem Sie in der Eingabehilfe "Übersicht" auf das Symbol **Neues Schema hinzufügen** klicken und anschließend zum gewünschten Schema navigieren. Nachdem ein Schema

© 2016-2022 Altova GmbH

hinzugefügt wurde, werden seine Definitionen in der Eingabehilfe "Übersicht" angezeigt. So sehen Sie z.B. in der Abbildung unten, dass das Schema `TelNumbers.json` hinzugefügt wurde und dass dieses Schema eine Definition namens `USTelephoneNumbers` hat. Sie können beliebig viele externe Schemas hinzufügen.



Wiederverwenden einer externen Definition

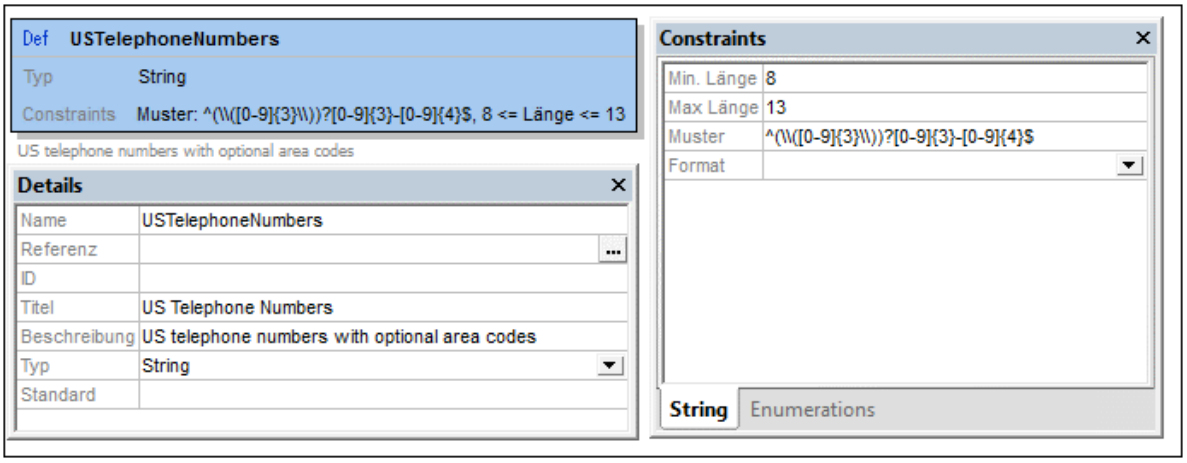
Nachdem ein externes Schema hinzugefügt wurde, stehen seine Definitionen zur Wiederverwendung in den Definitionen des importierenden Schemas zur Verfügung. Wenn in einer Definition eine andere Definition (durch Referenzieren) wiederverwendet wird, so werden die Eigenschaften dieser Definition übernommen. Die Referenzierung kann auf zwei Arten erfolgen:

- *in der Design-Ansicht:* Durch Ziehen einer Definition aus der Eingabehilfe "Übersicht" auf die gewünschte Definition.
- *im Definitionsübersichtsraster oder der Design-Ansicht:* Über das Feld *Referenz* der Eingabehilfe "Details" der Definition, in der die Definition wiederverwendet werden soll. Dies ist weiter unten in der Beschreibung der Eingabehilfe "Details" 731 erläutert.

Anmerkung: Über das Symbol **Aktualisieren** neben dem Eintrag *Externe Schemas* im Übersichtsfenster können alle hinzugefügten externen Schemas aktualisiert werden. Beachten Sie Folgendes: Wenn keine Definition aus einem hinzugefügten externen Schema wiederverwendet wurde, so wird das Schema aus der Liste entfernt, wenn die Liste aktualisiert wird.

Eingabehilfe "Details"

Die Eigenschaften einer Definition können in den Fenstern "Details" und "Constraints" eingegeben werden, wenn die Definition in einem der beiden Modi im Hauptfenster (Definitionsübersichtsraster oder Design-Ansicht 726) ausgewählt ist. In der Abbildung unten sehen Sie die Definition von `USTelephoneNumbers` in der Design-Ansicht 735 zusammen mit den Eingabehilfen "Details" und "Constraints". Beachten Sie, dass die Informationen, die in den beiden Eingabehilfen angezeigt werden, auch im (blauen) Definitionskasten in der Design-Ansicht zu sehen sind. Unten finden Sie eine Liste der Eigenschaften, die in diesen beiden Eingabehilfen definiert werden.



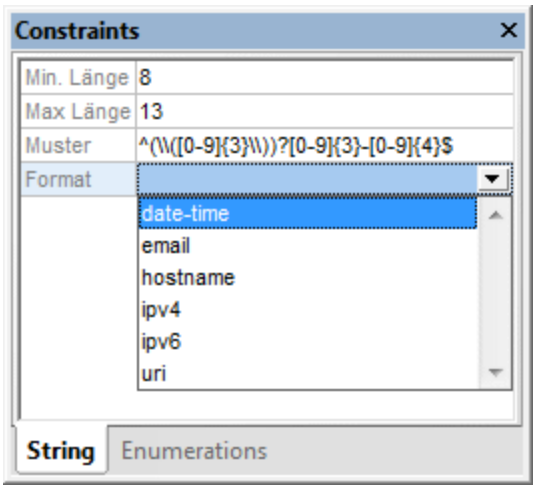
In der Eingabehilfe "Details" können die folgenden Informationen eingegeben werden:

- *Name:* Der Name der Definition.
- *Referenz:* Klicken Sie auf die Schaltfläche **Zusätzliches Dialogfeld** des Felds *Referenz* (*siehe Abbildung oben*), wenn in der Definition eine andere Definition wiederverwendet werden soll. Daraufhin wird das Dialogfeld "Referenz bearbeiten", in dem alle verfügbaren Definitionen (aus dem aktuellen und aus externen Schemas) aufgelistet sind, angezeigt. Wählen Sie die Definition, die Sie wiederverwenden möchten, aus aktivieren Sie die Option *Referenzpfad relativ machen*, wenn der Pfad relativ gemacht werden soll und klicken Sie auf **OK**. Nähere Informationen dazu finden Sie unter Globale und lokale Definitionen[733].
- *Typ:* Wählen Sie den Datentyp der Definition aus der Dropdown-Liste der Auswahlliste aus. Wenn Sie den Typ ändern, werden auch Schlüsselwörter, die für den vorherigen Typ spezifisch waren, entfernt. Wenn Sie die vorherige Definition wiederherstellen möchten, drücken Sie **Rückgängig (Strg+Z)**. Eine Erläuterung zu den Typen finden Sie unter JSON-Daten[680] und Typ-Selektoren (Beliebig, Mehrere, usw.)[752].
- *ID:* Dies ist ein optionales Schlüsselwort, das eine URI für das Schema definiert. Anhand dieser URI kann das Schema referenziert werden. Sie wird als Basis-URI für andere URI-Referenzen im Schema verwendet. Der ID-Wert muss ein String in Form einer URI sein. Beachten Sie, dass im Altova JSON-Validator ausschließlich kanonische Dereferenzierung verwendet wird. Nähere Informationen dazu finden Sie in der JSON-Spezifikation.
- *Anker (neu in draft-2019-09):* Dies ist ein optionales IS-Schlüsselwort, das ein Fragment in Form eines reinen Namens (und nicht wie im Fall von ID eine URI) bereitstellt. Der Wert von *Anker* muss, ein String, wie im jeweiligen Draft beschrieben, sein.
- *Titel, Beschreibung:* Die Werte dieser beiden Schlüsselwörter dienen zu Beschreibungszwecken und können vom Endbenutzer gelesen werden.
- *Kommentar (neu in draft-07):* Dient, im Gegensatz zu "Beschreibung", die für Endbenutzer gedacht ist, für Anmerkungen für Schema-Entwickler.
- *Const (neu in draft-06):* Ein Konstantenwert, wie z.B. eine aus einem Wert bestehende Enumeration.
- *Standard:* Der Standardwert der Definition.
- *Schreibgeschützt, Lesegeschützt (neu in draft-07):* Damit wird ein schreib- bzw. lesegeschütztes Feld gekennzeichnet. Ein Beispiel für ein lesegeschütztes Feld wäre z.B. ein Passwortfeld
- *Veraltet (neu in draft-2019-09):* Gibt an, dass die ausgewählte Definition in Zukunft entfernt werden kann. Applikationen können solche Definitionen auf spezielle Art behandeln.

© 2016-2022 Altova GmbH

Eingabehilfe "Constraints"

Die Constraints (Einschränkungen) einer Definition hängen vom Typ der Definition ab. Unten finden Sie eine Beschreibung der Constraints der einzelnen Typen. (Siehe auch Atomare Typen[750]).



Wenn ein Typ in der nachstehenden Liste nicht angezeigt wird, können keine Constraints dafür definiert werden. Beachten Sie allerdings, dass für alle Typen Enumerationen definiert werden können:

- *String:* die Länge des String und das String-Muster; das Muster wird mit Hilfe einer Regular Expression definiert. Im Feld *Format* können Sie eines der in der Spezifikation definierten String-Formate auswählen (*siehe Abbildung oben, in der Sie die in draf-04 verfügbaren Formate sehen*); In neueren Versionen wurden zusätzliche Formate definiert. MIt *Content Media Type* und *Content Encoding* (beide neu in draft-07) werden der Medientyp und die Kodierung von in einem JSON-String kodierten Nicht-JSON-Daten ausgewählt.).
- *Numerisch:* der zulässige Wertebereich
- *Array:* Die Anzahl der im Array zulässigen Elemente und ob Elemente eindeutig sein müssen.
- *Objekt:* die Anzahl der zulässigen Eigenschaften

Die Eingabehilfe "Constraints" hat für alle Typen ein Register "Enumerations". Auf diesem Register können Sie eine Liste der zulässigen Elemente für diesen Definitionstyp definieren. Zusätzlich dazu steht für alle Typen mit Ausnahme von *Unzulässigen* ein Register Beispiele (neu in draft-06) zur Verfügung. Es handelt sich hierbei um ein Array von Beispielen ohne Validierungsauswirkung; der Wert von `Standard` kann als Beispiel verwendet werden, ohne dass er unter diesem Schlüsselwort wiederholt wird.

13.6.4 Globale und lokale Definitionen

JSON-Schema-Definitionen können global oder lokal erstellt werden.

- **Globale Definitionen** werden im Definitionsübersichtsraster[726] des Hauptfensters durch Hinzufügen einer Definition und anschließendes Definieren ihrer Eigenschaften[729] erstellt. Eine globale Definition kann von anderen Definitionen im selben Schema oder auch von Definitionen in anderen Schemas referenziert werden. Auf diese Art können Sie Definitionen im gesamten Projekt wiederverwenden. Alle globalen Definitionen des aktuellen Schemas werden im Definitionsübersichtsraster[726] des Schemas

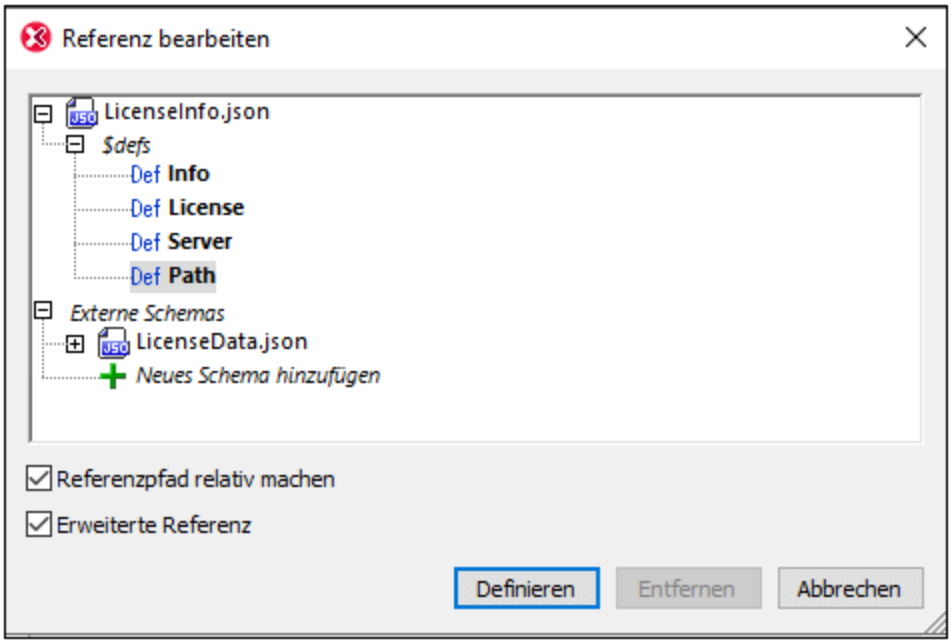
angezeigt. Um globale Definitionen aus anderen Schemas wiederverwenden zu können, müssen Sie das externe Schema[730] in der Eingabehilfe "Übersicht" hinzufügen.

- **Lokale Definitionen** werden innerhalb von globalen Definitionen erstellt, indem Sie untergeordnete oder gleichrangige Definitionen zu einer globalen Definition hinzufügen.

Referenzieren einer globalen Definition

Um eine globale Definition von einer anderen Definition aus zu referenzieren, wählen Sie eine der folgenden Methoden:

- Ziehen Sie die globale Definition aus der Eingabehilfe "Übersicht"[730] in die Design-Ansicht, und zwar auf die Definition, in der sie verwendet werden soll.
- Klicken Sie in der Design-Ansicht[726] mit der rechten Maustaste auf die Definition, für die Sie eine globale Definition referenzieren möchten und wählen Sie **Referenz bearbeiten**. (Gehen Sie alternativ dazu bei ausgewählter Definition in der Design-Ansicht zur Eingabehilfe "Details" und klicken Sie auf die Schaltfläche **Zusätzliches Dialogfeld** des Felds *Referenz*.) Wählen Sie im daraufhin angezeigten Dialogfeld "Referenz bearbeiten" (*Abbildung unten*) die gewünschte globale Referenz aus. Eine andere Methode, das Dialogfeld "Referenz bearbeiten" aufzurufen, ist, die referenzierende Definition auszuwählen, in der Eingabehilfe "Details" zu ihrem Feld *Referenz* zu gehen und auf die Schaltfläche **Zusätzliches Dialogfeld** des Felds der ausgewählten wiederzuverwendenden globalen Definition zu klicken (*nähere Informationen dazu finden Sie in der Beschreibung der Eingabehilfe "Details"*[731]).



Anmerkung: Eine Definition kann (i) eine globale Definition referenzieren und keine lokale Definition oder lokalen Constraint enthalten oder (ii) (ab *draft-2019-09*) sowohl eine globale Definition referenzieren als auch gleichzeitig lokale Definitionen/Constraints enthalten. In letzterem Fall wird die Referenz als **erweiterte Referenz** bezeichnet. Sie können im Dialogfeld "Referenz bearbeiten" eine Referenz auf eine globale Definition als erweiterte Referenz erstellen, indem Sie das Kontrollkästchen *Erweiterte Referenz* des Dialogfelds aktivieren. Eine als erweiterte Referenz erstellte globale Definition wird in der Liste der Definitionen der referenzierenden Komponenten immer als der letzte Eintrag angezeigt. Wenn für das Schlüsselwort `description` der referenzierenden Komponente ein Wert vorhanden ist, wird dieser Wert unterhalb der referenzierenden Komponente angezeigt.

© 2016-2022 Altova GmbH

Anmerkung: Wenn Sie den Namen einer globalen Definition ändern, nachdem dieser von einer anderen Definition im selben Schema referenziert wurde, so ändert sich auch der Name in der Referenz. Referenzen von anderen Schemas müssen jedoch manuell bearbeitet werden, wenn deren Namen geändert wurden.

Konvertieren von lokalen Definitionen in globale Definitionen

Um eine lokale Definition in eine globale zu konvertieren, klicken Sie in der Design-Ansicht[735] mit der rechten Maustaste darauf und wählen Sie den Befehl Als global definieren. Daraufhin wird eine globale Definition erstellt und in der lokalen Definition wird eine Referenz erstellt. Da der Name der globalen Definition automatisch generiert wird, können Sie diesen bearbeiten. Diese Änderung wird in der Referenz der lokalen Definition übernommen.

Ändern einer Referenz zu einer globalen Definition in eine lokale Definition

Sowohl lokale als auch globale Definitionen können Referenzen zu einer globalen Definition enthalten. Um die Referenz zu entfernen und ihre Eigenschaften lokal zu machen, klicken Sie in der Design-Ansicht[735] mit der rechten Maustaste auf die (lokale oder globale) Definition und wählen Sie den Befehl **Als lokal definieren**. Daraufhin werden die Eigenschaften der globalen Definition lokal in der Definition erstellt.

13.6.5 Design-Ansicht

In der Design-Ansicht können Sie die Struktur und die zulässigen Werte einzelner globaler Definitionen definieren. Diese Informationen werden über die folgenden Komponenten der Benutzeroberfläche bzw. mit Hilfe der folgenden Methoden definiert:

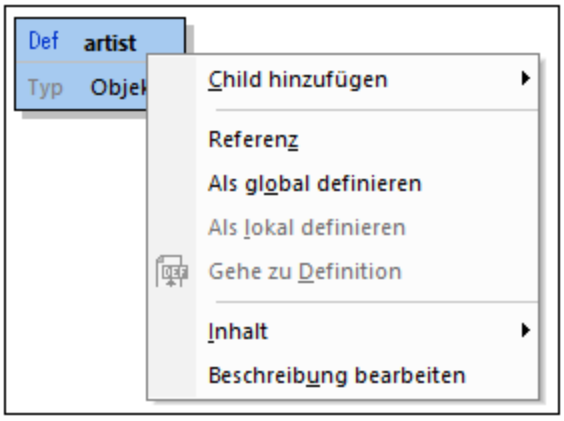
- die Eingabehilfe "Details"[731] (steht auch im Definitionsübersichtsraster zur Verfügung)
- die Eingabehilfe "'Constraints"[733] (steht auch im Definitionsübersichtsraster zur Verfügung)
- das Kontextmenü der Definition (wird durch Rechtsklick auf den Definitionskasten im Hauptfenster aufgerufen)

Im Abschnitt Eingabehilfen: Übersicht, Details, Contstraints[730] finden Sie eine Beschreibung der Definitionen, die über die Eingabehilfen "Details" und "Constraints" definiert werden können. Einige dieser Eigenschaften können auch im Hauptfenster im Kasten der Definition festgelegt werden. In diesem und den folgenden drei Abschnitten werden die hauptsächlich grafischen Mechanismen, die im Hauptfenster zur Verfügung stehen, beschrieben.

Anmerkung: Um eine unabsichtliche oder unerwünschte Änderung rückgängig zu machen, drücken Sie **String + Z**.

Kontextmenü

Über das Kontextmenü einer Definition (*blauer Kasten in der Abbildung unten*) können Sie die Struktur der Definition erstellen und ihre Eigenschaften bearbeiten.



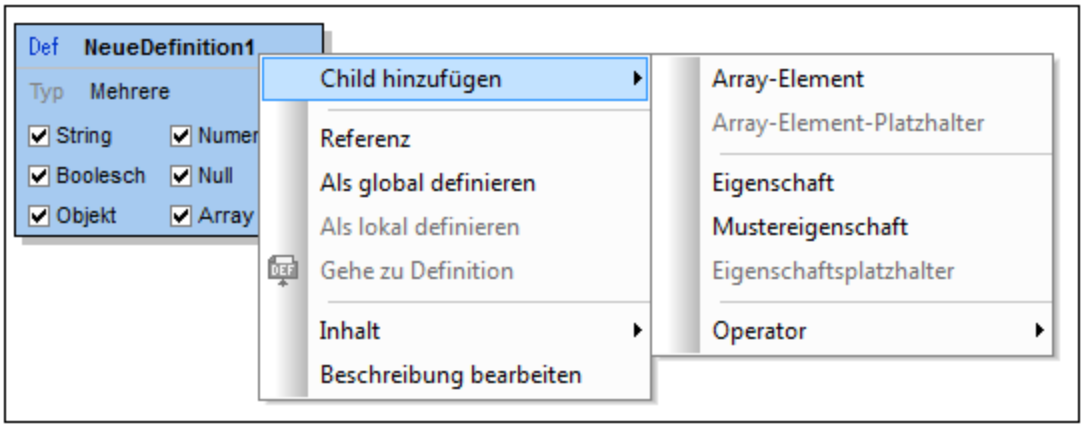
Es stehen die folgenden Befehle zur Verfügung:

- *Child hinzufügen:* Was als Child hinzugefügt werden kann, hängt vom Typ der Definition ab (siehe *Child hinzufügen: Erstellen einer Struktur*[736] weiter unten).
- *Referenz bearbeiten:* Ermöglicht das Referenzieren einer globalen Definition und die Übernahme der Eigenschaften dieser globalen Definition in der Definition. Das Dialogfeld "Referenz bearbeiten", das mit diesem Befehl aufgerufen wird, ist dasselbe, das auch über die Eingabehilfe "Details"[731] aufgerufen werden kann. Eine Beschreibung dazu finden Sie im Abschnitt Eingabehilfen: Übersicht, Details, Contstraints[731].
- *Als global definieren:* Dieser Befehl ist aktiv, wenn die Definition eine lokale Definition[733] ist. Damit wird die aktuell ausgewählte Definition zu einer globalen Definition[733] gemacht und in der aktuell ausgewählten Definition wird eine Referenz zu dieser globalen Definition hinzugefügt.
- *Als lokal definieren:* Dieser Befehl ist aktiv, wenn die Definition eine globale Definition[733] ist. Mit diesem Befehl wird die aktuell ausgewählte Definition in eine lokale Definition[733] konvertiert, indem eine Referenz zur globalen Originaldefinition erstellt wird.
- *Gehe zu Definition:* Wenn die ausgewählte Definition sich innerhalb einer Definition befindet, die eine globale Definition referenziert, so ist dieser Befehl aktiv. Bei Auswahl des Befehls gelangen Sie zur globalen Definition.
- *Inhalt:* Mit dem Befehl **Inhalt** wird ein Submenü mit Befehlen zum Ausschneiden, Kopieren und Zurücksetzen des Inhalts der ausgewählten Definition angezeigt.
- *Beschreibung bearbeiten:* Mit diesem Befehl können Sie das Feld *Beschreibung* der Definition editierbar machen.

Child hinzufügen: Erstellen einer Struktur

Die Struktur einer Definition wird erstellt, indem man mehrere untergeordnete Ebenen hinzufügt. Diese Ebenen werden mit dem Kontextmenübefehl **Child hinzufügen** erstellt. Welche Children zu einer Definition hinzugefügt werden können, hängt vom Typ der Definition ab:

© 2016-2022 Altova GmbH



- *Objekte*[737]*:* haben Eigenschaften und Operatoren
- *Arrays*[748]*:* haben Array-Elemente und Operatoren
- *Atomare Typen (String, Zahl, Boolesch, Null)*[750]*:* haben Operatoren
- *Beliebig*[752]*:* haben Eigenschaften, Array-Elemente und Operatoren
- *Mehrere*[752]*:* Variiert je nachdem, welche Typen inkludiert sind; hat alle der insgesamt für die ausgewählten Typen zulässigen Children
- *Operatoren*[758]: ermöglicht die Verwendung logischer Operatoren zum Festlegen der Struktur

Unter den Links zu den Einträgen der Liste oben sind die Strukturen beschrieben, die für die einzelnen Typen erstellt werden können.

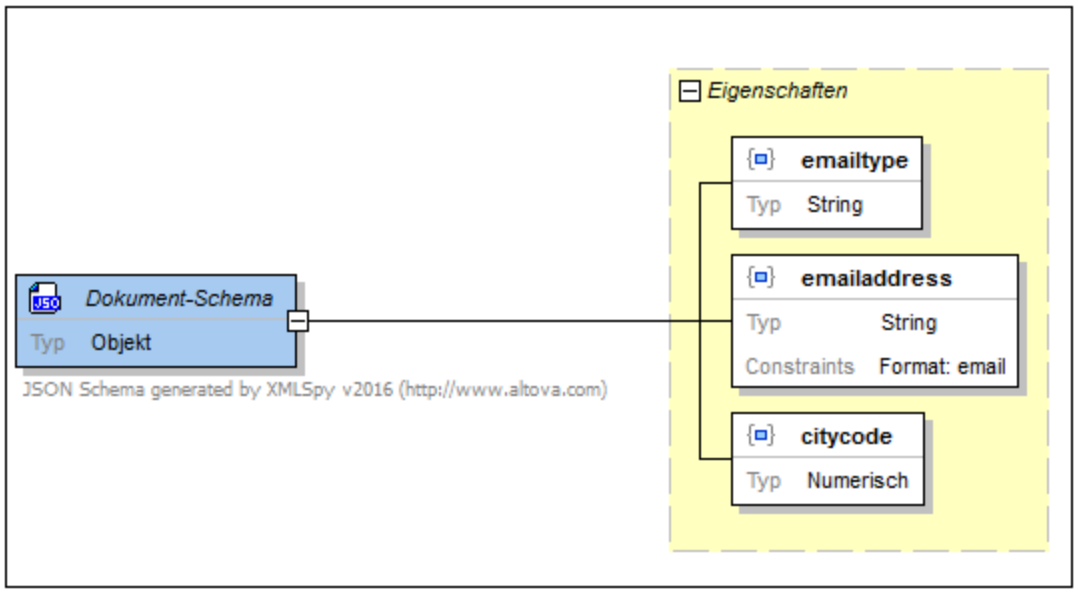
13.6.6 Objekte und Eigenschaften

In einem Objekt wird ein Wert einem Schlüssel auf folgende Art und Weise zugeordnet: `"MeinSchlüssel": Wert`. Beim Schlüssel muss es sich immer um einen String handeln, daher muss dieser immer in Anführungszeichen gesetzt werden. Der Wert kann jeden beliebigen JSON-Datentyp[680] haben. Jedes `Schlüssel:Wert`-Paar wird als Eigenschaft des Objekts bezeichnet (*siehe Abbildung unten*).

Hier sehen Sie ein Beispiel eines instantiierten Objekts mit drei Eigenschaften:

```
{
  "emailtype": "home",
  "emailaddress": "contact01.home@altova.com",
  "citycode": 22
}
```

Das Schema für das Objekt würde in der Design-Ansicht etwa folgendermaßen aussehen:



Beachten Sie bitte die folgenden Punkte:

- Jede der Eigenschaften muss in der Instanz vorhanden sein. Dies wird durch die durchgezogene Umrandung der Eigenschaften gekennzeichnet. Wenn eine Eigenschaft optional ist, so ist die Umrandung eine gestrichelte Linie. Ob eine Eigenschaft obligatorisch oder optional ist, können Sie im Kontextmenü der Eigenschaft oder in der Eingabehilfe "Details" definieren.
- Die Reihenfolge, in der Eigenschaften in der Instanz vorkommen, ist nicht im Schema definiert (und kann dort auch nicht definiert werden), d.h. die Reihenfolge, in der Eigenschaften im Schema definiert sind, spielt keine Rolle.
- Eine Eigenschaft wird durch das Symbol eines blauen Quadrats innerhalb von geschweiften Klammern gekennzeichnet. Im Gegensatz dazu wird eine Mustereigenschaft oder ein Eigenschaftsplatzhalter durch andere Symbole gekennzeichnet; *siehe unten* 738).
- Der Typ einer Eigenschaft kann durch Doppelklick auf den Typ im Diagramm und Auswahl einer Option aus der daraufhin angezeigten Dropdown-Liste bearbeitet werden. Alternativ dazu kann der Typ auch in der Eingabehilfe "Details" ausgewählt werden.
- Der Constraint-Wert der Eigenschaft `emailaddress` wird in der Eingabehilfe "Constraints" definiert.

Eigenschaften, Mustereigenschaften, Eigenschaftsplatzhalter und Eigenschaftsnamenschemas

Ein Objekt kann Eigenschaften, Mustereigenschaften, Eigenschaftsplatzhalter und Eigenschaftsnamenschemas (neu in draft-07 727) haben. Diese können über die Kontextmenüs (i) des Objekts, (ii) des gelben Eigenschaftskastens (Rechtsklick auf den Titel *Eigenschaften* des Kastens) und (iii) einzelner Eigenschaften zum Objekt hinzugefügt werden. Die Eigenschaften wurden oben beschrieben. Nun wollen wir Mustereigenschaften und Eigenschaftsplatzhalter näher beschreiben.

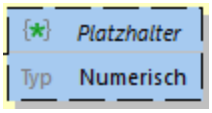
Mustereigenschaft

Eine **Mustereigenschaft** (*Abbildung unten*) definiert den Namen der Eigenschaft mit Hilfe einer Regular Expression. So wird z.B. in der Regular Expression in der Abbildung unten definiert, dass die Eigenschaft (i) einen Namen haben muss, der mit einem Unterstrich beginnt und (ii) als Wert einen Booleschen Wert haben muss. Mustereigenschaften haben keine obligatorisch-Einschränkung. Sie können beliebig viele Mustereigenschaften hinzufügen. Beachten Sie das Symbol für Mustereigenschaften.

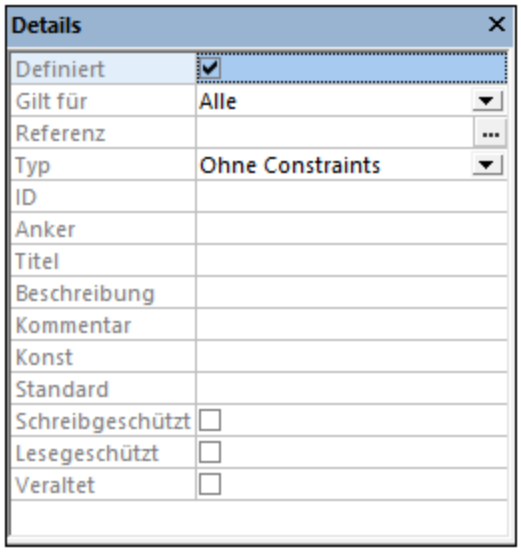
© 2016-2022 Altova GmbH



Eigenschaftsplatzhalter
Ein **Eigenschaftsplatzhalter** (*Abbildung unten*) definiert, dass neben den anderen Eigenschaften der Eigenschaftsgruppe des Objekts beliebig viele Eigenschaften vorhanden sein können. Mit dem Platzhalter kann jedoch ein Typ für diese definiert werden. In der Abbildung links unten sehen Sie einen Eigenschaftsplatzhalter, der Eigenschaften mit einem beliebigen Namen aber einem numerischen Wert definiert. Es kann nur ein Eigenschaftsplatzhalter pro Objekt definiert werden. Wenn für den Platzhalter jedoch der Typ *Beliebig* definiert ist, so können Sie für jeden Typ in der Eingabehilfe "Constraints" Constraints definieren. Beachten Sie das Symbol für Eigenschaftsplatzhalter.



Ab *draft-2019-09* haben Eigenschaftsplatzhalter das neue Schlüsselwort `unevaluatedProperties`, welches nur verarbeitet wird, wenn das Schlüsselwort `additionalProperties` fehlt. Die Werte dieser zwei Schlüsselwörter werden durch Definition entsprechender Werte für die Einträge *Definiert*, *Gilt für* und *Typ* in der Eingabehilfe "Details" eines Platzhalters (*Abbildung unten*) erzeugt.



In der Tabelle unten sehen Sie, wie sich diese Werte auf die Schlüsselwörter `unevaluatedProperties` und `additionalProperties` auswirken (und, wie sich die Schlüsselwörter umgekehrt auf die Werte der Eingabehilfe des Editors auswirken). So ist etwa in der Abbildung oben `unevaluatedProperties=true`.

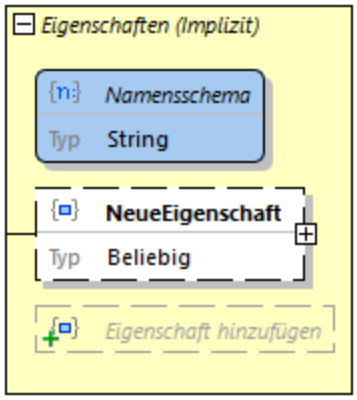
addtionalProperties	unevaluatedProperties		Definiert	Gilt für	Typ
--	--	<=>	false	Alle	Ohne Constraints
true	ignoriert	<=>	true	Alle	Ohne Constraints

false	ignoriert	<=>	--	--	--
Schema	ignoriert	<=>	true	Alle	Schema-Typ
--	true	<=>	true	Nicht ausgewertete	Ohne Constraints
--	false	<=>	true	Nicht ausgewertete	Unzulässig
--	Schema	<=>	true	Nicht ausgewertete	Schema-Typ

Beachten Sie die folgenden Punkte:

- Wenn `addtionalProperties` und `unevaluatedProperties` vorhanden sind, werden `unevaluatedProperties` ignoriert.
- *Definiert*=false funktioniert nur bei *Geltungsbereich*=Alle und *Type*=Ohne Constraints.

Eigenschaftsnamenschema
Ein **Eigenschaftsnamenschema** (*Abbildung unten*) schränkt die Namen der Eigenschaften dieses Objekts ein. (Diese Funktionalität ist neu in draft-07 727.) So sehen wir z.B. in der Abbildung unten, dass die Namen von Eigenschaften Strings sein müssen. Zusätzlich dazu können wir über die Eingabehilfe "Constraints" weitere Constraints für die Eigenschaft definieren: z.B. dass der Name der Eigenschaft in einem bestimmten Zeichenlängenbereich liegen oder einem bestimmten Muster entsprechen muss.



Anmerkung: Für Mustereigenschaften kann keine Mindest- oder Maximalanzahl definiert werden. Nähere Informationen dazu finden Sie im Abschnitt zur Validierung von Eigenschaften. 740

Wie werden Eigenschaften validiert?

Bei der Validierung von Eigenschaften geht der Validierungsprozessor folgendermaßen vor:

1. Zuerst wird der Name der Eigenschaft im Schema mit allen benannten Eigenschaften dieses Objekts verglichen.
2. Wenn keine Übereinstimmung gefunden wird, wird der Name mit allen Mustereigenschaften in der Eigenschaftsgruppe des Objekts verglichen.
3. Wenn immer noch keine Übereinstimmung gefunden wird, so wird der Platzhalter aufgerufen, falls einer vorhanden ist.
4. Wenn für den Namen immer noch keine Übereinstimmung gefunden wird, wird ein Validierungsfehler ausgegeben. Wenn der Name mit dem einer Eigenschaft oder einer Mustereigenschaft übereinstimmt

© 2016-2022 Altova GmbH

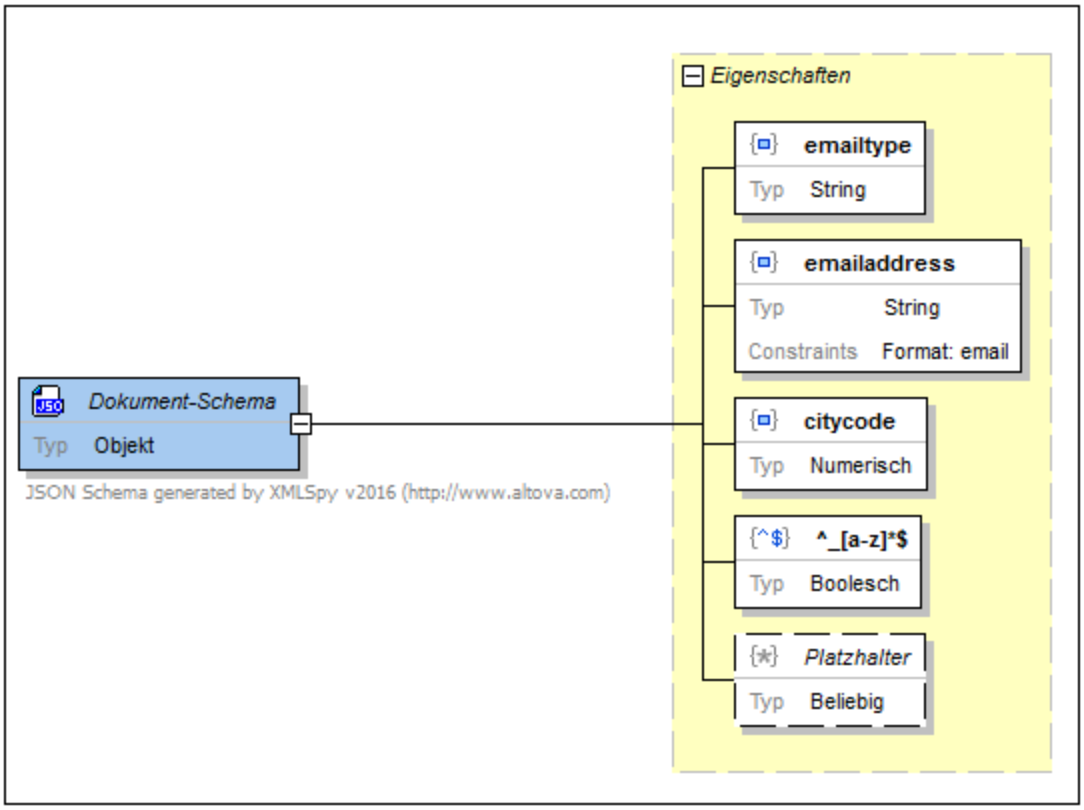
oder wenn ein Platzhalter vorhanden ist, so wird der Wert mit dem Wert der entsprechenden Eigenschaftsdefinition verglichen.

5. Wenn der Instanzwert dem Typ und den Constraints der entsprechenden Eigenschaftsdefinition entspricht, so ist die Eigenschaft gültig. Andernfalls ist sie ungültig.

Beispiel

In der Abbildung unten ist ein Objekt mit den folgenden Eigenschaften definiert:

- Es muss drei Eigenschaften namens `emailtype`, `emailaddress` und `citycode` haben.
- Es kann eine oder mehrere Eigenschaften haben, deren Name mit einem Unterstrich beginnt und deren Wert ein Boolescher Wert ist (siehe Mustereigenschaft in der Abbildung unten).
- Es kann eine oder mehrere zusätzliche Eigenschaften mit einem beliebigen Namen und Wert haben.



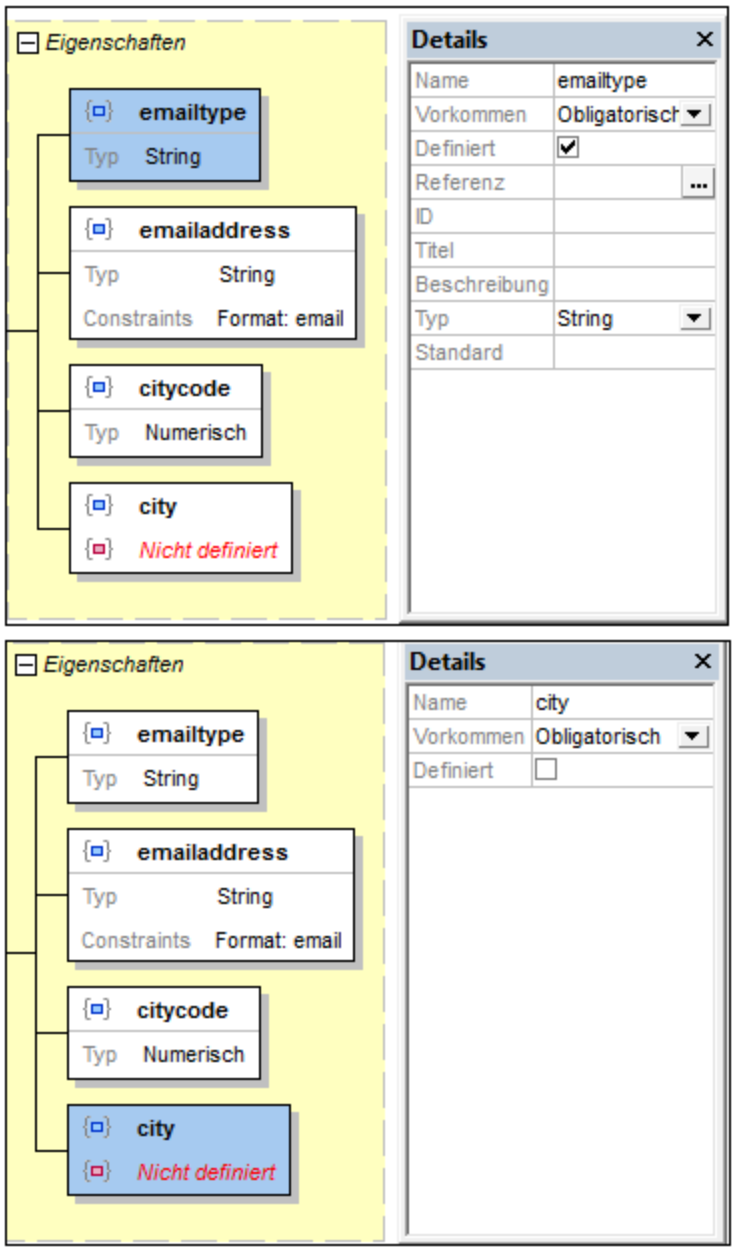
13.6.7 Nicht definierte Eigenschaften

Im Codefragment unten wird mit dem Schlüsselwort **`required`** (obligatorisch) festgelegt, dass für dieses Objekt vier Eigenschaften erforderlich sind. Von den vier obligatorischen Eigenschaften wurde jedoch nur drei definiert. Die vierte Eigenschaft, **`city`**, ist nicht definiert. Die definierten Eigenschaften werden als definiert bezeichnet, während die nicht definierte Eigenschaft als nicht definiert gilt. Siehe Abbildung unterhalb des Codefragments.

Codefragment: definierte und nicht definierte Eigenschaften

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "description": "JSON Schema generated by XMLSpy v2016 (http://www.altova.com)",
    "type": "object",
    "properties": {
        "emailtype": {
            "type": "string"
        },
        "emailaddress": {
            "type": "string",
            "format": "email"
        },
        "citycode": {
            "type": "number"
        }
    },
    "required": [
        "emailtype",
        "emailaddress",
        "citycode",
        "city"
    ],
    "additionalProperties": false
}
```


© 2016-2022 Altova GmbH



In der Design-Ansicht wird die nicht definierte Eigenschaft rot markiert, da sie im Schema als obligatorisch definiert, aber nicht vorhanden ist. Das JSON-Schema selbst ist zwar gültig, doch ist ein Instanzdokument, das anhand des Schemas validiert wird, nicht gültig. Dies hat folgenden Grund: (i) Wenn die Eigenschaft **`city`** nicht vorhanden ist, ist das Dokument ungültig, weil die Eigenschaft **`city`** obligatorisch ist. (ii) Wenn die Eigenschaft **`city`** vorhanden ist, ist das Dokument ungültig, weil die Eigenschaft **`city`** nicht definiert ist und kein Eigenschaftsplatzhalter vorhanden ist, aufgrund dessen die Eigenschaft vorhanden sein darf (*siehe* *Implizites Definieren einer Eigenschaft* 741 *weiter unten*).

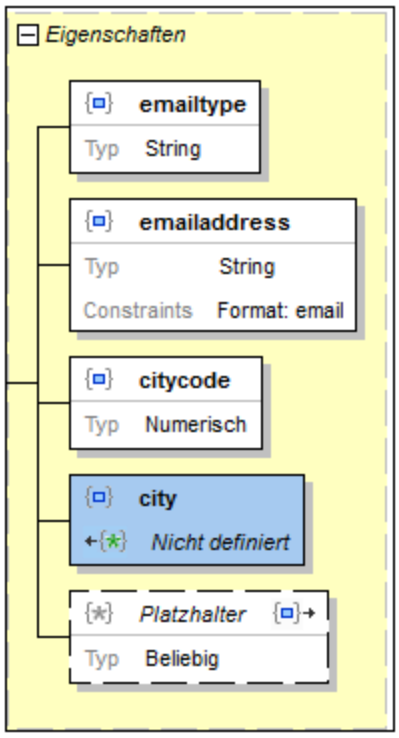
So erstellen Sie eine Definition für eine nicht definierte Eigenschaft:

1. Wählen Sie die nicht definierte Eigenschaft in der Design-Ansicht aus.

2. Aktivieren Sie in der Eingabehilfe "Details" das Kontrollkästchen *Definiert* (*siehe Abbildung oben*). Als Alternative dazu kann das Flag *definiert* über das Kontextmenü geändert werden.
3. Ändern Sie die Definition der Eigenschaft Ihren Wünschen entsprechend.

Implizites Definieren einer Eigenschaft

Eine Eigenschaft kann durch Hinzufügen einer passenden Mustereigenschaft oder eines Eigenschaftsplatzhalters implizit definiert werden. In der Abbildung unten sehen Sie, dass ein Eigenschaftsplatzhalter hinzugefügt wurde. Eine Instanzeigenschaft namens `city` stimmt mit diesem Platzhalter überein. Die Eigenschaft `city` wird somit im Schema durch den Platzhalter implizit definiert. Eine Instanzdatei, die die Eigenschaft `city` enthält, ist dem Schema gemäß gültig.



Beachten Sie die entsprechenden Symbole in der implizit definierten Eigenschaft und dem Eigenschaftsplatzhalter. Jedes Symbol ist ein Link zur anderen Eigenschaft. Wenn Sie auf ein Symbol doppelklicken, wird die andere Eigenschaft ausgewählt.

13.6.8 Objekte und Abhängigkeiten

Sie können innerhalb der Definition eines Objekts definieren, dass eine bestimmte Eigenschaft nur dann vorhanden sein soll, wenn eine andere Eigenschaft vorhanden ist. Die erste Eigenschaft ist von der zweiten Eigenschaft abhängig. Betrachten wir das folgenden Szenario, das eine Abhängigkeit enthält. Ein Objekt (geben wir ihm den Namen `member`) hat eine Eigenschaft namens `credit_card`, die als optional definiert ist. Die Eigenschaft `billing_address` des Objekts kann von der Eigenschaft `credit_card` abhängig gemacht werden: Nur wenn die Eigenschaft `credit_card` vorhanden ist, ist auch die Eigenschaft `billing_address` vorhanden.

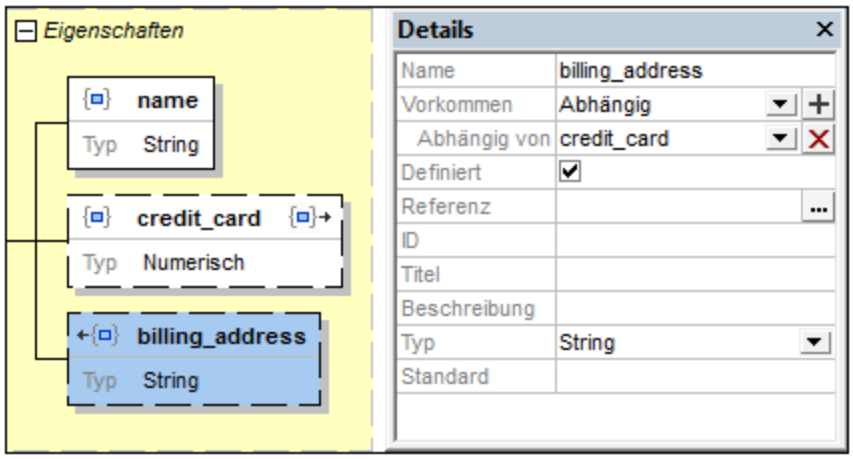
© 2016-2022 Altova GmbH

Diese Art von Abhängigkeit kann auf eine dieser beiden Arten definiert werden:

- als Eigenschaftsabhängigkeit (bei der abhängigen Struktur handelt es sich um eine Eigenschaft)
- als Schemaabhängigkeit (bei der abhängigen Struktur handelt es sich um ein Schema)

Eigenschaftsabhängigkeiten

In der Abbildung unten sehen Sie ein Objekt mit einer Eigenschaft `name` (obligatorisch), einer Eigenschaft `credit_card` (optional) und einer Eigenschaft `billing_address` (abhängig). Die Eigenschaft `billing_address` ist von der Eigenschaft `credit_card` abhängig. Unterhalb der Abbildung sehen Sie das Codefragment dieser JSON-Objektdefinition. Weiter unten erfahren Sie, wie Sie eine Eigenschaftsabhängigkeit erstellen.



Codefragment eines JSON-Objekts mit einer Eigenschaftsabhängigkeit

```
{
   "type": "object",
   "properties": {
      "name": {
      "type": "string"
      },
      "credit_card": {
         "type": "number"
      },
      "billing_address": {
         "type": "string"
      }
   },
   "required": [ "name" ],
   "dependencies": {
      "credit_card": [ "billing_address" ]
   },
   "additionalProperties": false
}
```

So erstellen Sie eine Eigenschaftsabhängigkeit:

1. Klicken Sie mit der rechten Maustaste auf eine Eigenschaft, auf der die Abhängigkeit basieren soll. (In unserem Beispiel ist dies die Eigenschaft `credit_card`.)
2. Wählen Sie im daraufhin angezeigten Kontextmenü den Befehl **Abhängigkeit hinzufügen | Abhängige Eigenschaft**. Daraufhin wird eine neue Eigenschaft mit dem Vorkommen-Wert *Abhängig* hinzugefügt.
3. Definieren Sie den Namen und Wert dieser Eigenschaft und fügen Sie je nach Bedarf alle zusätzlichen Details oder Constraints hinzu.

So definieren Sie eine Eigenschaft als von einer anderen abhängig:

1. Klicken Sie mit der rechten Maustaste auf die Eigenschaft, die Sie von einer anderen abhängig machen möchten. (In unserem Beispiel ist dies die Eigenschaft `billing_address`.)
2. Wählen Sie im daraufhin angezeigten Kontextmenü **Abhängig** aus. Gehen Sie alternativ dazu in der Eingabehilfe "Details" zum Eintrag *Vorkommen* und wählen Sie *Abhängig* (*siehe Abbildung oben*).
3. Klicken Sie in der Eingabehilfe "Details" auf das Dropdown-Listensymbol des Eintrags *Abhängig von*. Die Dropdown-Liste enthält alle anderen Eigenschaften des Objekts. Wählen Sie die Eigenschaft, von der die aktuelle Eigenschaft abhängig gemacht werden soll, aus.

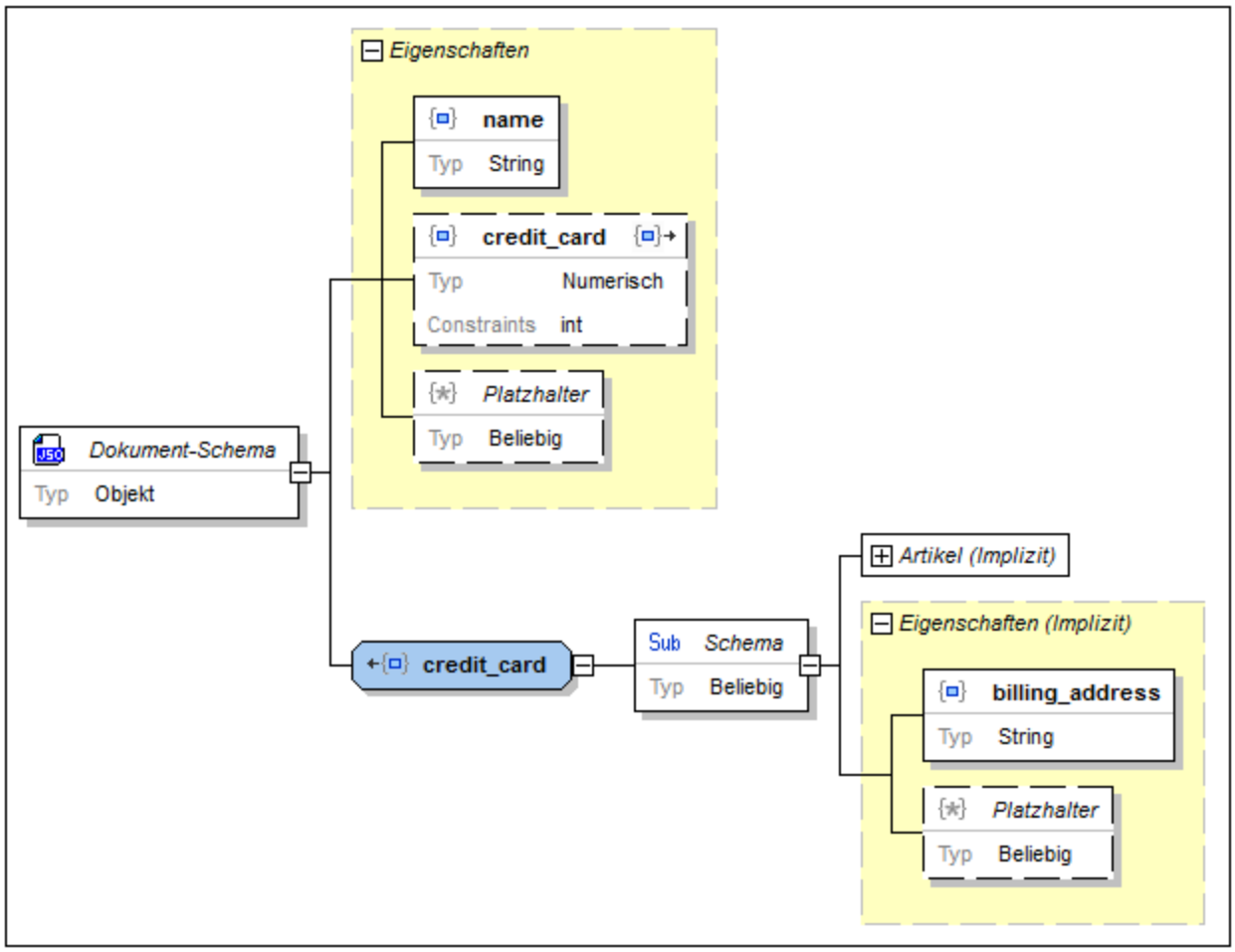
Anmerkung: In den Kästen der beiden an der Abhängigkeit beteiligten Eigenschaften wird ein Symbol angezeigt (*siehe Abbildung oben*). Wenn Sie auf das Symbol der einen Eigenschaft klicken, gelangen Sie zur anderen Eigenschaft.

Anmerkung: Eine Eigenschaft kann mehrere abhängige Eigenschaften haben.

Schemaabhängigkeiten

In der Abbildung unten sehen Sie ein Objekt, das dieselbe Instanzdatenstruktur, wie beim im vorherigen Abschnitt erläuterte Objekt beschreibt. Die Definitionen der beiden Objekte sind allerdings anders. Während in der vorherigen Definition eine *Eigenschaftsabhängigkeit* verwendet wurde, um die Eigenschaft `billing_address` als von der Eigenschaft `credit_card` abhängige Eigenschaft zu definieren, wird in der aktuellen Definition zur Definition dieser Abhängigkeit eine Schemaabhängigkeit verwendet. Unterhalb der Abbildung sehen Sie den Code dieser zweiten JSON-Objektdefinition. Weiter unten wird beschrieben, wie Sie eine Schemaabhängigkeit erstellen.

© 2016-2022 Altova GmbH



Codefragment eines JSON-Objekts mit einer Schemaabhängigkeit

```
{
   "type": "object",
   "properties": {
      "name": {
         "type": "string"
      },
      "credit_card": {
         "type": "integer"
      }
   },
   "required": [ "name" ],
   "dependencies": {
      "credit_card": {
         "properties": {
            "billing_address": {
               "type": "string"
            }
         },
         "required": [ "billing_address" ]
      }
```

```
        }
    }
```

So erstellen Sie eine Schemaabhängigkeit:

1. Klicken Sie mit der rechten Maustaste auf die Eigenschaft, auf der die Abhängigkeit basieren soll. (In unserem Beispiel ist dies die Eigenschaft `credit_card`.)
4. Wählen Sie im daraufhin angezeigten Kontextmenü **Abhängigkeit hinzufügen | Schemaabhängigkeit** aus. Daraufhin wird eine neue Objektdefinition erstellt. Sie erhält denselben Namen wie die Eigenschaft, von der sie abhängig ist (in unserem Beispiel `credit_card`) und hat ein Child-Subschema.
2. Definieren Sie das Subschema Ihren Wünschen entsprechend, fügen Sie alle zusätzlichen Details oder Constraints, die Sie benötigen, hinzu.

Anmerkung: In den Kästen der Eigenschaft und des an der Abhängigkeit beteiligten Objekts, wird ein Symbol angezeigt (*siehe Abbildung oben*). Wenn Sie auf das Symbol in dem einen Kasten klicken, gelangen Sie zum anderen Kasten.

Anmerkung: Wenn Sie mehrere Abhängigkeiten definieren möchten, tun Sie dies im abhängigen Subschema (*siehe Abbildung oben).*

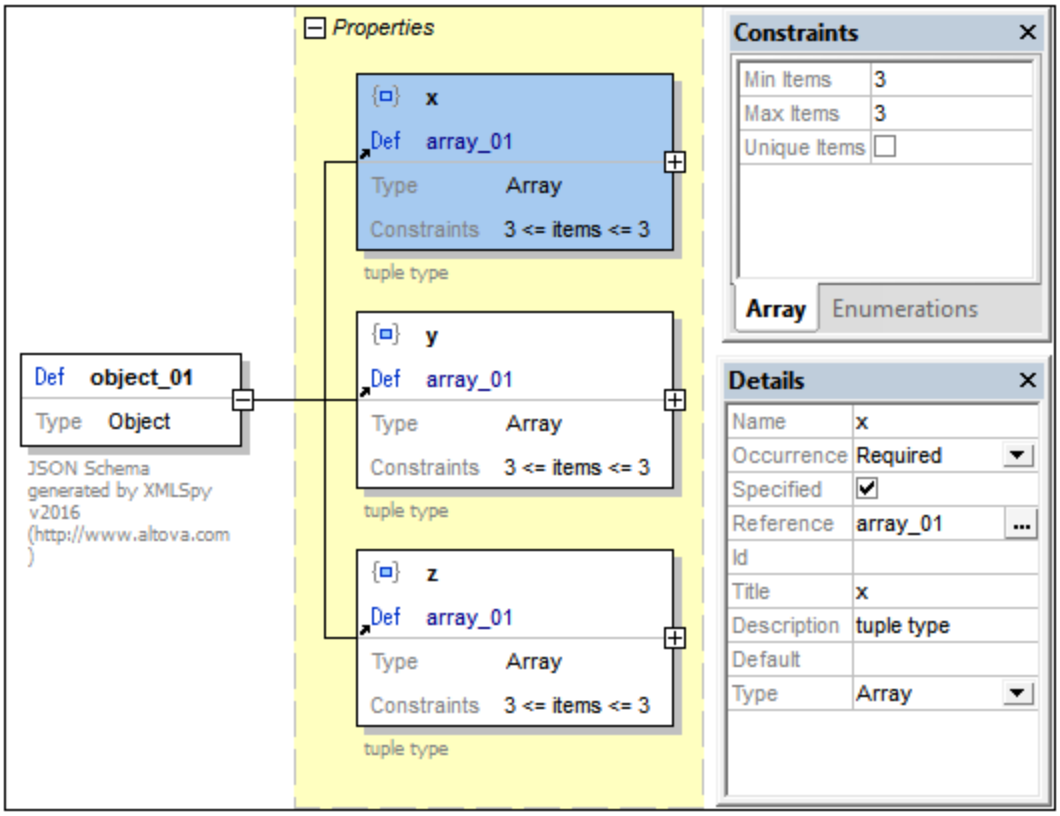
13.6.9 Arrays

Ein Array ist eine Liste von null oder mehr geordneten Elementen; ein Array steht innerhalb von eckigen Klammern. Jedem Element in der Liste wird ein Typ zugewiesen. Das Instanzcodefragment unten ist ein Objekt mit drei Eigenschaften. Der Wert jeder Eigenschaft ist ein Array (*die Trennzeichen sind gelb markiert*).

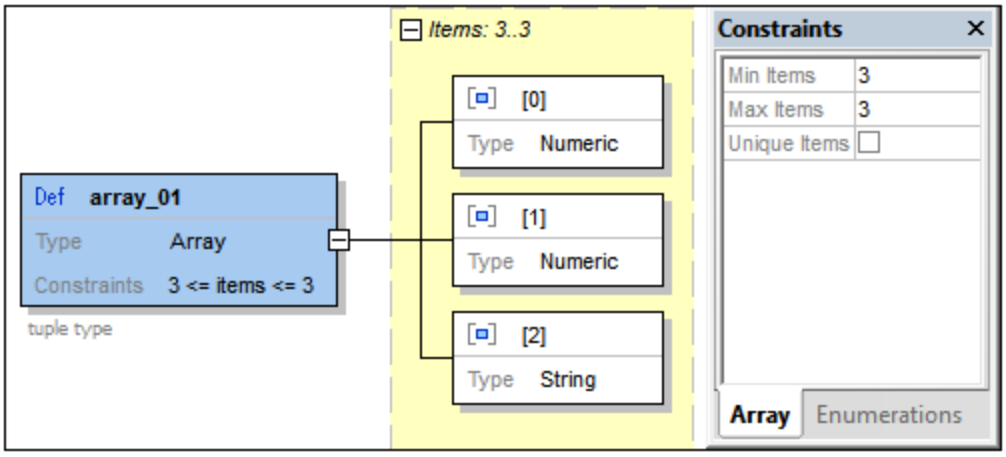
```
{
    "x": [ 1, 2, "abc" ],
    "y": [ 3, 4, "def" ],
    "z": [ 5, 6, "ghi" ]
}
```

Alle drei Arrays im Codefragment oben haben dieselbe Definition. Jedes enthält drei geordnete Elemente in der folgenden Reihenfolge: (i) ein Zahlelement, ein (ii) Zahlelement, (iii) ein String-Element. In der Abbildung unten sehen Sie dieses Objekt als Schemabeschreibung. Da die Definition für alle drei Arrays dieselbe ist, wurde sie in einem globalen Array namens `array_01` erstellt. Jedes der drei Arrays (`x`, `y` und `z`) referenziert das globale Array[733] `array_01`.

© 2016-2022 Altova GmbH



In der Abbildung oben ist Array **x** ausgewählt (blau markiert). Seine Details und Constraints werden in den jeweiligen Eingabehilfe angezeigt (*siehe Abbildung oben*). Beachten Sie den Constraint, also die Einschränkung, zur Anzahl der zulässigen Elemente. Die Zahl kann in der Eingabehilfe "Constraints" bearbeitet werden und wird im Diagramm angezeigt. Die Array-Elemente können in der Definition des Array selbst, in diesem Fall in der globalen Definition `array_01`, definiert werden (*Abbildung unten*).



Beachten Sie die folgenden Punkte:

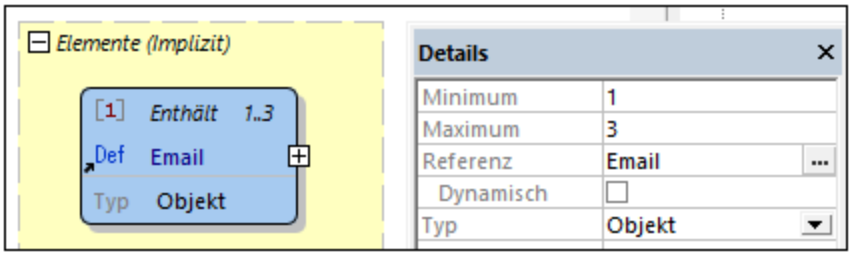
- Der Constraint `eindeutig` definiert, dass alle Elemente im Array eindeutig sein müssen.

- Die Nummerierung der Elemente beginnt mit `0`.
- Die Angaben *3* <= *Elemente* <= *3* und *Elemente: 3..3* (*siehe Abbildung oben*) geben die Mindest- und Maximalanzahl der zulässigen Elemente an. In diesem Fall müssen genau drei Elemente vorhanden sein.

Hinzufügen von Array-Elementen, Array-Elementplatzhaltern und dem Schlüsselwort "Enthält"

Array-Elemente, Array-Elementplatzhalter und das Schlüsselwort **`Enthält`** eines Array werden über das Kontextmenü einer Definition oder eines Array-Elements hinzugefügt.

- Mit Hilfe eines Array-Platzhalters kann eine breitere Palette an Objekten in das Array inkludiert werden.
- Mit dem Schlüsselwort **`Enthält`** wird definiert, dass der Wert des Schlüsselworts **`Enthält`** ein gültiges JSON-Schema sein muss und dass mindestens eines der Elemente des Array gemäß dem referenzierten Schema-Objekt gültig sein muss. Ab draft 2019-09 stehen die Schlüsselwörter **`minContains`** und **`maxContains`** zur Verfügung. In der Eingabehilfe "Details" werden diese als die *Min-* und *Max*-Eigenschaft des Enthält-Felds angezeigt (*siehe Abbildung unten*). Damit legen Sie fest, wie viele Einträge mit dem referenzierten Schema-Objekt übereinstimmen dürfen.



13.6.10 Atomare Typen

Es gibt fünf atomare JSON-Typen (auch einfache oder primitive Typen genannt) : (i) `String`, (ii) `Zahl`, (iii) `Ganzzahl`, (iv) `Boolesch` und (v) `Null`. Um für eine Definition einen dieser atomaren Typen zu definieren, wählen Sie eine der folgenden Methoden:

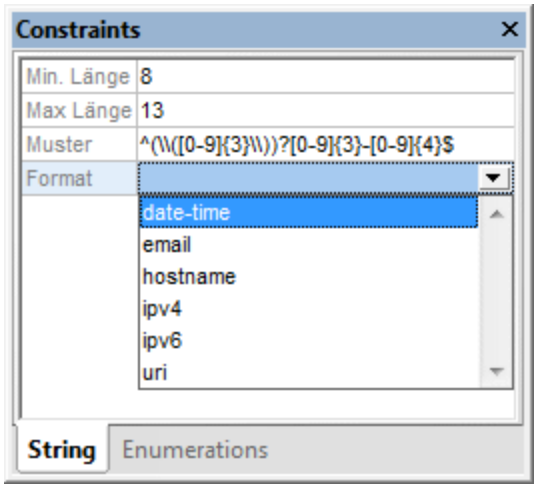
- Doppelklicken Sie im Definitionskasten auf das *Typ*-Wertefeld und wählen Sie den Typ aus.
- Wählen Sie den Typ in der Eingabehilfe "Details" aus der Dropdown-Liste im Feld *Typ* aus.

Im Folgenden finden Sie eine Beschreibung der Constraints der einzelnen atomaren Typen.

String

Für den Typ "String" können Sie die folgenden Constraints definieren: (i) die String-Länge, (ii) eine Regular Expression, die das Muster des String beschreibt, (iii) ein vordefiniertes Format aus der Spezifikation.

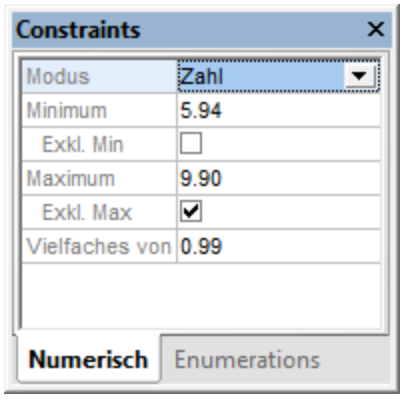
© 2016-2022 Altova GmbH



Anmerkung: Sie können in den JSON-Validierungseinstellungen[1615] von XMLSpy festlegen, ob das Format von Strings in JSON-Instanzdokumenten validiert werden soll odern nicht.

Numerisch

Der Typ "Numerisch" ist ein Überbegriff für zwei Typen (`Zahl` und `Ganzzahl`; *siehe Abbildung unten*). Der eigentliche Typ wird im Feld *Modus* definiert (die Standardeinstellung davon ist `Zahl`). Der Unterschied zwischen den beiden Typen ist, dass für den Typ `Zahl` Nachkommastellen erlaubt sind, während dies beim Typ `integer` nicht der Fall ist. Falls im Feld *Vielfaches von* ein Wert vorhanden ist, so muss der Instanzwert eine Ganzzahl sein, die ein Vielfaches des Werts im Feld *Vielfaches von* beträgt.



Gültige Werte für den in der Abbildung oben gezeigten Typ `Zahl` sind: `5,94`, `6,93`, `7,92` und `8,91`.

Boolesch und Null

Der Typ `Boolesch` erhält als Wert entweder `true` oder `false`. Der Typ `Null` erhält als Wert `Null`. Für keinen dieser beiden Typen gibt es Constraints.

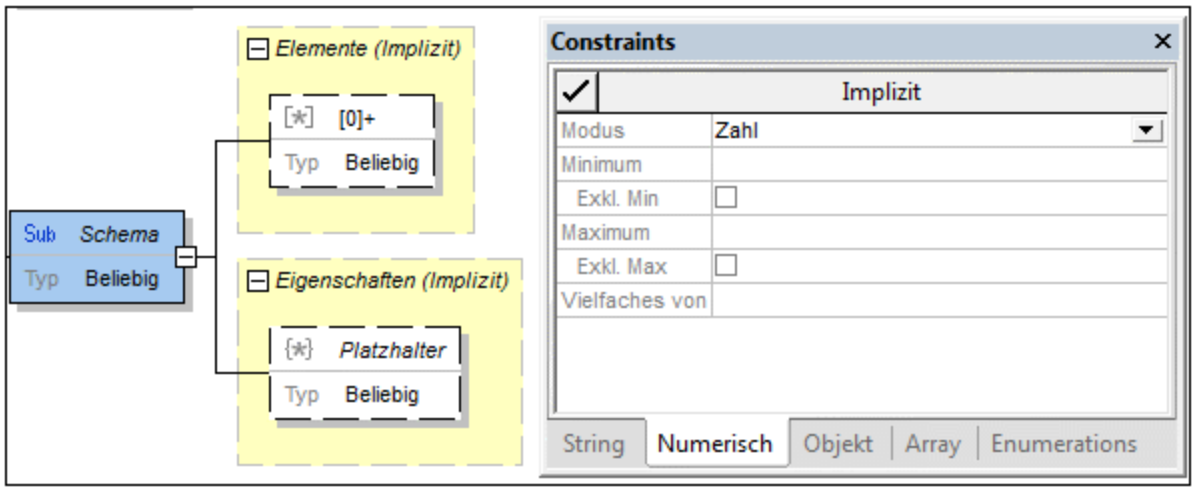
13.6.11 Typ-Selektoren (Beliebig, Mehrere, usw.)

Die Dropdown-Listen der Typ-Auswahllisten der JSON-Schema-Ansicht, enthält vier "Typen", die keine JSON-Typen sind: **`Beliebig`**, **`Mehrere`**, **`ohne Constraints`** und **`unzulässig`**. Dabei handelt es sich eigentlich um Typ-Selektoren.

- Mit dem Typ-Selektor **`Beliebig`** wird jeder beliebige JSON-Typ ausgewählt. Das bedeutet, dass jeder JSON-Typ in dieser Instanz für die jeweilige Definition gültig ist.
- Mit dem Typ-Selektor **`Mehrere`** werden einer oder mehrere JSON-Typen ausgewählt. Das bedeutet: Wenn es sich beim Instanztyp um einen der im Schema ausgewählten JSON-Typen handelt, so ist er für die jeweilige Definition gültig.
- Mit dem Typ-Selektor **`ohne Constraints`** (neu in draft-06 727) werden keine Constraings im JSON-Typ definiert, d.h. in der Instanz ist für eine Definition mit diesem Namen jeder JSON-Typ gültig.
- Bei Auswahl des Typ-Selektors **`unzulässig`** (neu in draft-06 727) ist kein JSON-Typ zulässig, d.h. es darf keine Definition mit diesem Namen vorhanden sein.

Der Typ-Selektor Beliebig

Der Typ-Selektor `Beliebig` kann überall, wo ein Typ ausgewählt werden kann, verwendet werden. Wenn eine Definition zum Schema hinzugefügt wird, so ist `Beliebig` der ausgewählte Standardtyp. Damit wird festgelegt, dass jeder der JSON-Typen gültig ist. Dadurch wäre der Instanztyp gültig, wenn es sich um ein Objekt, ein Array oder einen der atomaren Typen (`String`, `Zahl`, `Ganzzahl`, `Boolesch` und `Null`) handelt.



Das Subschema in der Abbildung oben hat den Typ `Beliebig`. Daher sind für diese Definition alle JSON-Typen gültig. Die folgenden Komponenten sind implizit vorhanden und daher auf der Benutzeroberfläche entsprechend implementiert:

- Da Objekte zulässig sind, wird automatisch ein Kasten "Eigenschaften" erstellt (*siehe Abbildung oben).* Laut Standarddefinition sind im Eigenschaftskasten beliebig viele Eigenschaften jedes beliebigen Typs zulässig (über einen Eigenschaftsplatzhalter mit dem Typ `Beliebig`). Sie können die Eigenschaftsdefinitionen nach Belieben ändern.
- Da Arrays zulässig sind, wird automatisch ein Kasten "Elemente" erstellt (*siehe Abbildung oben*). Für den Array-Elementkasten sind standardmäßig beliebig viele Array-Elemente jedes beliebigen Typs

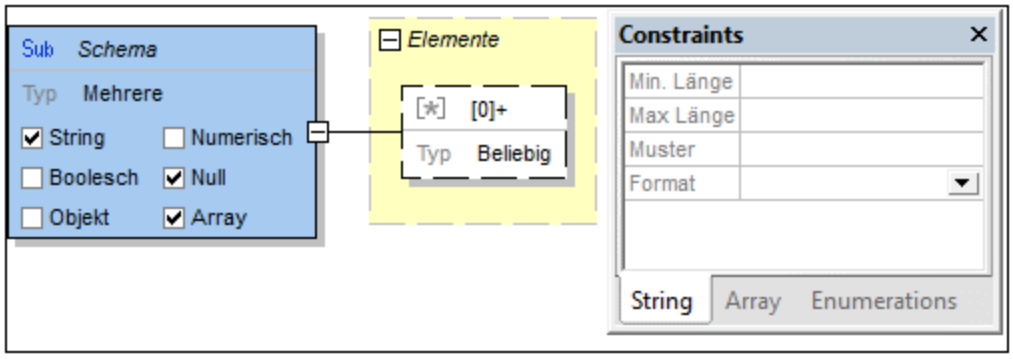
© 2016-2022 Altova GmbH

zulässig (über einen Array-Elementplatzhalter mit dem Typ `Beliebig`). Sie können die Elementdefinitionen nach Belieben bearbeiten.,
- Da String-Typen und numerische Typen (Zahl und Ganzzahl) zulässig sind, können in der Eingabehilfe "Constraints" Constraints für diese atomaren Typen definiert werden.

Alle diese Typen werden daher implizit mit dem Typ-Selektor `Beliebig` definiert. Um den Typ in einen bestimmten Typ zu ändern, wählen Sie diesen Typ aus. Um als Typ Objekte und Arrays zu definieren, gibt es eine alternative Methode: Klicken Sie mit der rechten Maustaste auf das Objekt oder Array und wählen Sie **Als explizit definieren**. Dadurch wird dieser Typ zum ausgewählten Typ und andere Typen werden entfernt bzw. werden definierte Objekt/Array-Typen inaktiv.

Der Typ-Selektor Mehrere

Der Typ-Selektor `Mehrere` kann überall dort, wo ein Typ ausgewählt werden kann, ausgewählt werden. Sie können damit durch Aktivieren der Kontrollkästchen für den/die gewünschten Typen einen oder mehrere JSON-Typen auswählen (*siehe Abbildung unten*). Anschließend können Sie in der Eingabehilfe "Constraints" Constraints für die ausgewählten Typen definieren.



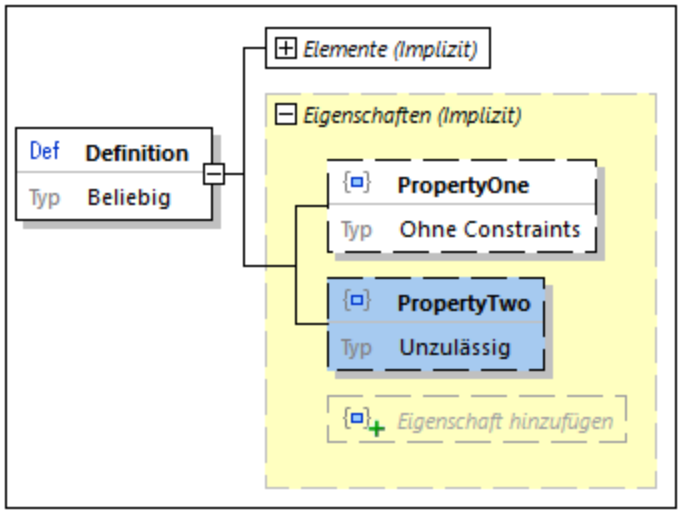
In der Abbildung oben sind gemäß dem Subschema die Typen `String`, `Null` und `Array` zulässig. Constraints für diese Typen können in der Eingabehilfe "Constraints" definiert werden (*siehe Abbildung*).

- String-Constraints werden in der Eingabehilfe "Constraints" definiert.
- Für den Typ "Null" können keine weiteren Constraints definiert werden.
- Es wird automatisch ein Array-Elementkasten erstellt. Sie können die Anzahl und Typen der zulässigen Array-Elemente definieren.

Die ausgewählten Typen sind in einem Instanzdokument an der Stelle, die dem des Subschemas entspricht, zulässig.

Die Typ-Selektoren "ohne Constraints" und "unzulässig"

Die Typ-Selektoren `ohne Constraints` und `unzulässig` können überall dort, wo ein Typ zur Auswahl steht, ausgewählt werden. Sie können damit festlegen, dass ein Objekt jeden zugelassenen Typ haben kann bzw. dass kein Objekt dieses Namens zulässig ist.



In der Abbildung oben hat eine Definition zwei Eigenschaften. **`PropertyOne`** kann einen Wert eines beliebigen Typs haben, während eine Eigenschaft mit dem Namen **`PropertyTwo`** nicht zulässig ist (*siehe Abbildung*). In Textform sieht dieses Konstrukt wie das Codefragment unten aus.

```
"Definition": {
	"properties": {
		"PropertyOne": true,
		"PropertyTwo": false
	}
}
```

13.6.12 BSON (Binary JSON) für MongoDB

In der MongoDB-Applikationsdatenplattform werden Daten als JSON-Strukturen, jedoch in einer binären Darstellung der Daten, gespeichert. Diese Darstellung wird als "Binary JSON" oder BSON bezeichnet. Die Verwendung von BSON für MongoDB hat die folgenden wichtigen Vorteile:

- Das Binärformat von BSON lässt sich schneller als der Text eines JSON-Dokuments parsen.
- Da in JSON Datentypen nur eingeschränkt unterstützt werden, wurde BSON mit mehr Datentypen (insbesondere mehr numerischen Datentypen) versehen.

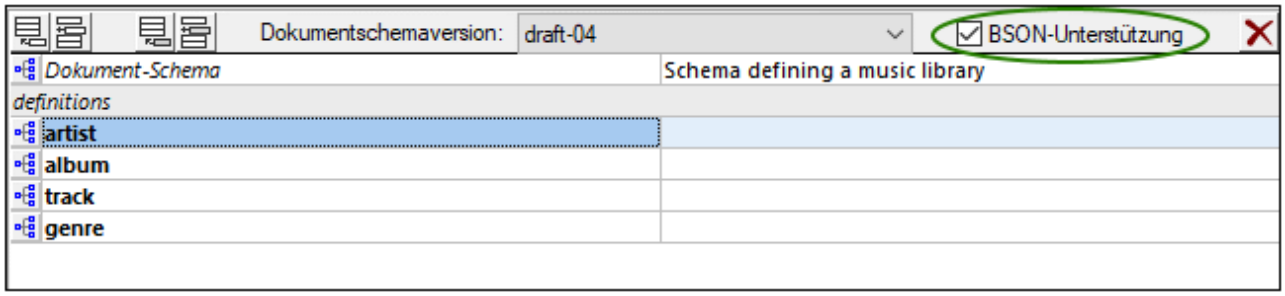
Nähere Informationen zu MongoDB und BSON finden Sie auf dieser Seite der MongoDB-Website.

Bearbeiten von JSON-Schemas für BSON-Daten

MongoDB bietet die Möglichkeit, Datenbankdaten bei Hinzufügung neuer Datenbankdaten oder Änderung vorhandener anhand eines JSON-Schema-Dokuments zu validieren. Da die zusätzlichen BSON-Datentypen aber nicht Teil der offiziellen JSON-Schema-Spezifikationen sind, sondern diese ergänzen, werden JSON-Schemas für BSON in XMLSpy über eine zusätzliche über den JSON-Schema-Editor darübergelegte Bearbeitungsebene für BSON bearbeitet. Über diese Bearbeitungsebene können Sie zusätzlich zu den JSON-Standardfunktionalitäten BSON-spezifische Schemafunktionalitäten hinzufügen und bearbeiten.

© 2016-2022 Altova GmbH

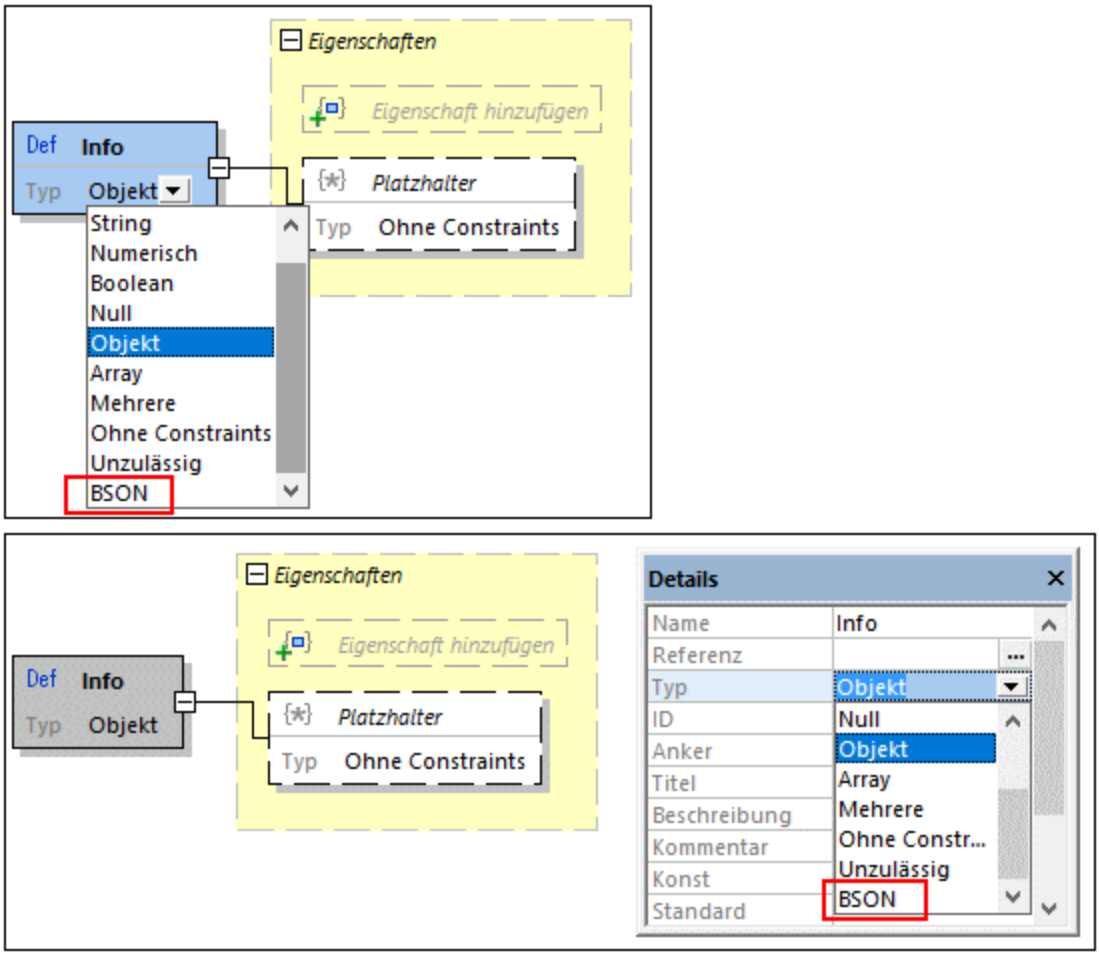
Sie können ein JSON-Schema-Dokument jeder beliebigen Version also mit oder ohne die BSON-Bearbeitungsebene bearbeiten. Wenn das Dokument ohne die BSON-Bearbeitungsebene bearbeitet wird, wird es als einfaches JSON-Schema-Dokument behandelt. Mit BSON-Unterstützung können im JSON-Schema-Dokument zusätzlich BSON-spezifische Funktionalitäten definiert werden. Um in der JSON-Schema-Ansicht BSON-Bearbeitungsfunktionalitäten *für das aktive Dokument* zu aktivieren, aktivieren Sie rechts oben im Hauptfenster das Kontrollkästchen *BSON-Unterstützung (in der Abbildung unten grün umrandet)*. Beachten Sie, dass die BSON-Unterstützung (i) unabhängig von der ausgewählten JSON-Schema-Version aktiviert werden kann und (ii) nur auf das aktuelle Dokument angewendet wird; sie kann für jedes Dokument separat aktiviert/deaktiviert werden.



BSON-Typen

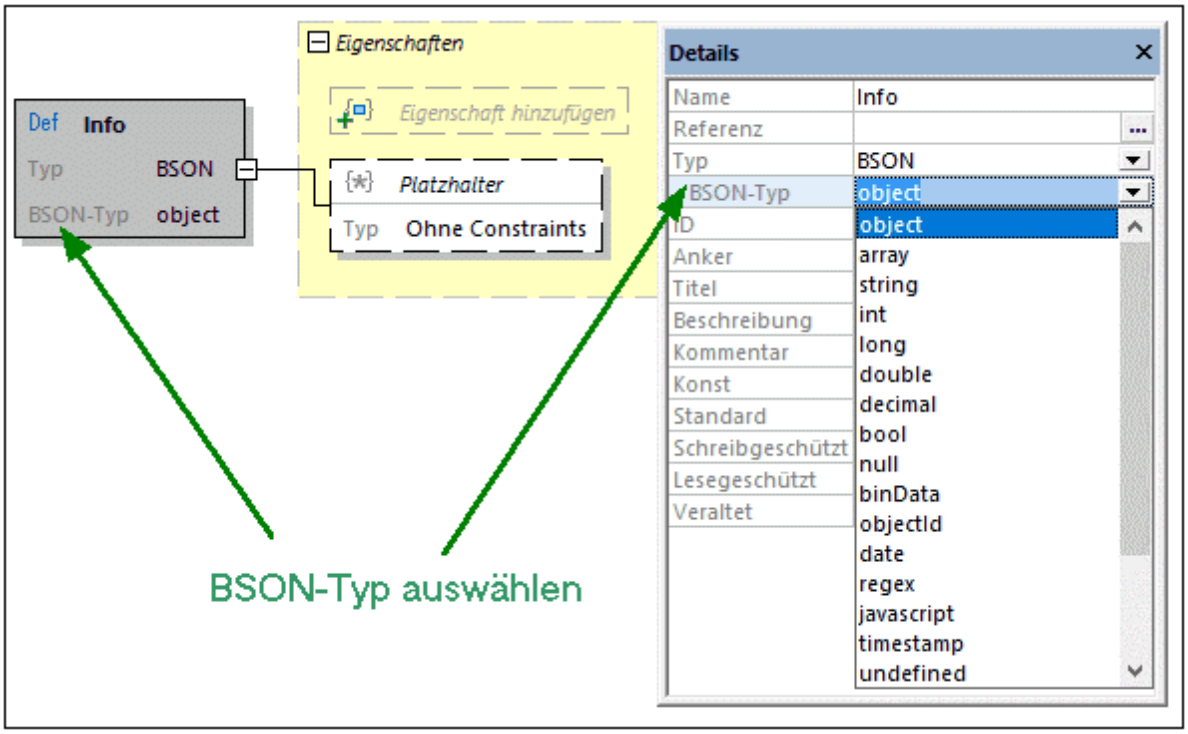
Nach Aktivierung der BSON-Bearbeitungsebene stehen für JSON-Objekte, -Eigenschaften und Array-Elemente BSON-Datentypen zur Verfügung. Die Definition, dass eine Komponente ein BSON-Typ ist, erfolgt in zwei Schritten:

1. Definieren Sie, dass es sich bei der Komponente um einen BSON-Datentyp (und nicht einen JSON-Datentyp) handelt, indem Sie BSON-als JSON-Basistyp auswählen. Dies kann entweder über die Datentyp-Auswahlliste der Komponente (durch Doppelklick auf den Typwert (*Abbildung links unten*) oder über die Eingabehilfe "Details" der Komponente (*Abbildung rechts unten*) erfolgen.



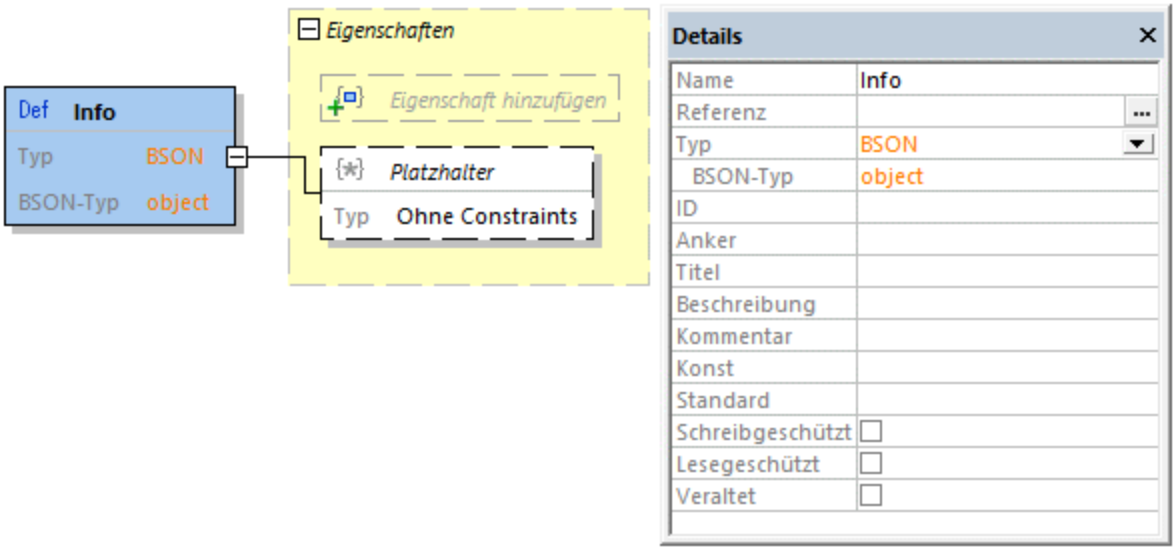
2. Nach Auswahl von BSON als Basistyp der Komponente, stehen die *BSON*-Typen zur Auswahl zur Verfügung. Wählen Sie den BSON-Typ entweder in der Datentypauswahl der Komponente (*linker Pfeil in der Abbildung unten*) oder in der Eingabehilfe "Details" (*rechter Pfeil in der Abbildung unten*) aus.

© 2016-2022 Altova GmbH



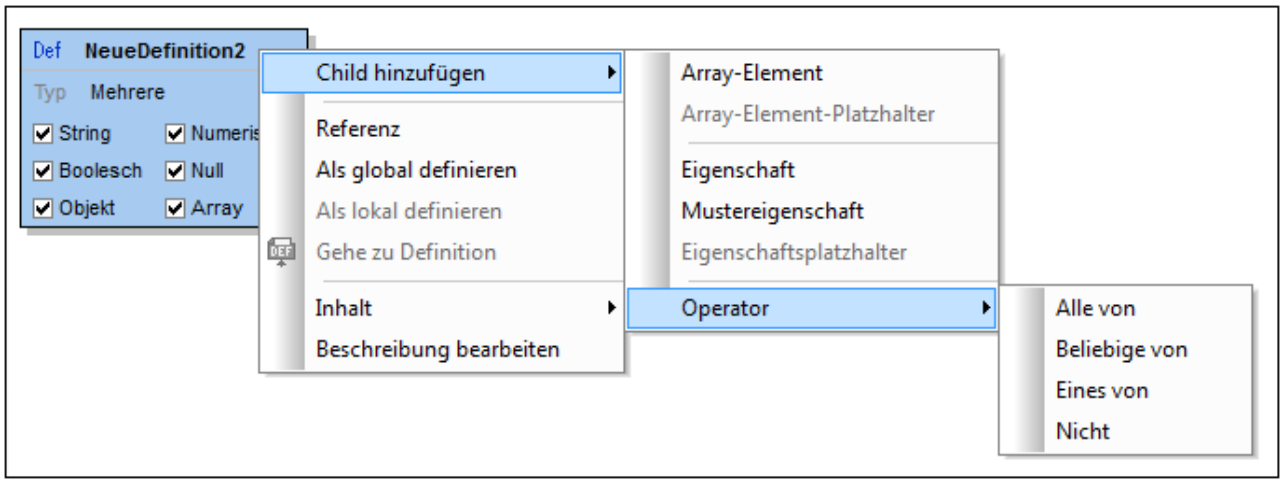
Deaktivieren der BSON-Unterstützung

Wenn Sie das Kontrollkästchen *BSON-Unterstützung* nach der oben beschriebenen Zuweisung eines BSON-Typs zu einer JSON-Schema-Komponente für das Dokument deaktivieren, erscheint ein Meldungsfeld. Darin werden Sie informiert, dass im Dokument BSON-Typen vorhanden sind und gefragt, ob Sie die BSON-Typen entfernen/konvertieren oder beibehalten möchten. Wenn Sie sich entschließen, die Typen zu entfernen/zu konvertieren, werden diejenigen BSON-Typen, die in JSON-Typen konvertiert werden können, konvertiert, während die anderen entfernt werden. Wenn Sie die BSON-Typen beibehalten, bleiben diese erhalten, werden in der Schema-Ansicht aber orange markiert, da die Unterstützung von BSON-Typen entfernt wurde (*siehe Abbildung unten*).



13.6.13 Operatoren

Es gibt vier Operatoren: (i) `Alle von`, (ii) `Beliebiges von`, (iii) `Eines von` und (iv) `Nicht`. Mit Hilfe von Operatoren können, wie unten beschrieben, Gültigkeitsbedingungen definiert werden. Ein Operator kann zu jeder beliebigen Definition hinzugefügt werden. Um das Untermenü "Operator" aufzurufen, klicken Sie mit der rechten Maustaste auf die Definition, zu der Sie einen Operator hinzufügen möchten und wählen Sie **Child hinzufügen | Operator** (*siehe Abbildung unten*).



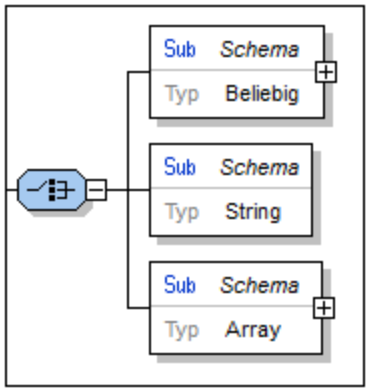
Mit diesen Operatoren können Bedingungen für eine erfolgreiche Validierung definiert werden:

Operator	Symbol	Beschreibung

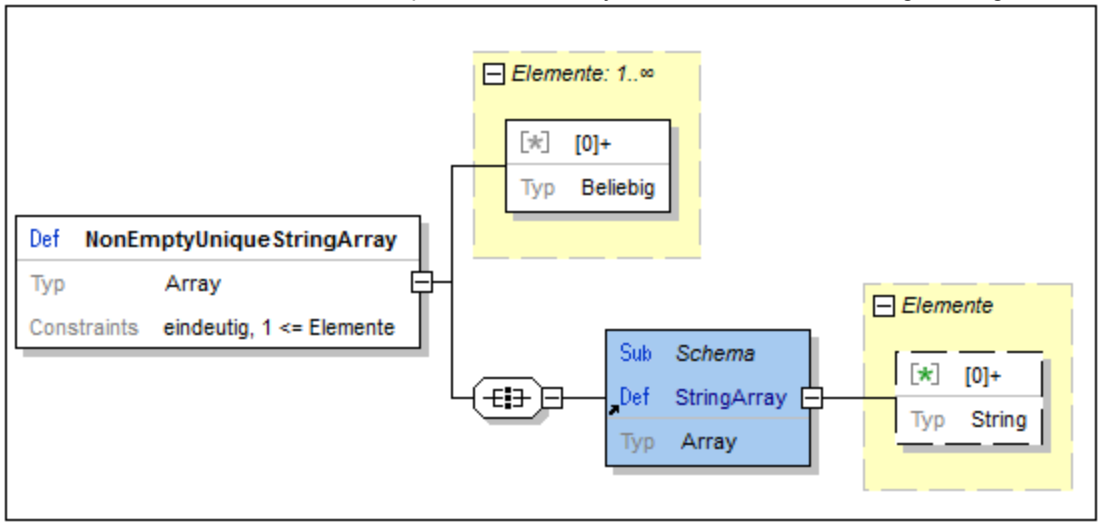
© 2016-2022 Altova GmbH

	Alle von	Enthält ein oder mehrere Subschemas (Definitionen), die als Children des Operators hinzugefügt werden bzw. wurden. Eine Instanz ist gültig, wenn sie gemäß allen diesen Subschemas gültig ist.
	Beliebige von	Enthält ein oder mehrere Subschemas (Definitionen), die als Children des Operators hinzugefügt werden bzw. wurden. Eine Instanz ist gültig, wenn sie gemäß mindestens einem dieser Subschemas gültig ist.
	Eines von	Enthält ein oder mehrere Subschemas (Definitionen), die als Children des Operators hinzugefügt werden bzw. wurden. Eine Instanz ist gültig, wenn sie gemäß genau einem dieser Subschemas gültig ist.
	Nicht	Enthält genau ein Subschema (Definition), das als Child des Operators hinzugefügt wird bzw. wurde. Eine Instanz ist gültig, wenn sie gemäß der angegebenen Definition ungültig ist.

In der Abbildung unten sehen Sie einen *Eines von*-Operator, der drei Subschemas (Definitionen) enthält. Damit die Instanz gültig ist, muss sie (an dieser Stelle der Dokumentstruktur) eine JSON-Datenstruktur haben, die gemäß einer der drei Subschema-Definitionen gültig ist.



Operatoren eignen sich gut, um Vererbung und Einschränkungen zu definieren. In der Abbildung unten sehen Sie, wie Sie mit Hilfe des *Alle von*-Operators ein Array, das nicht leere eindeutige Strings enthält, definieren.

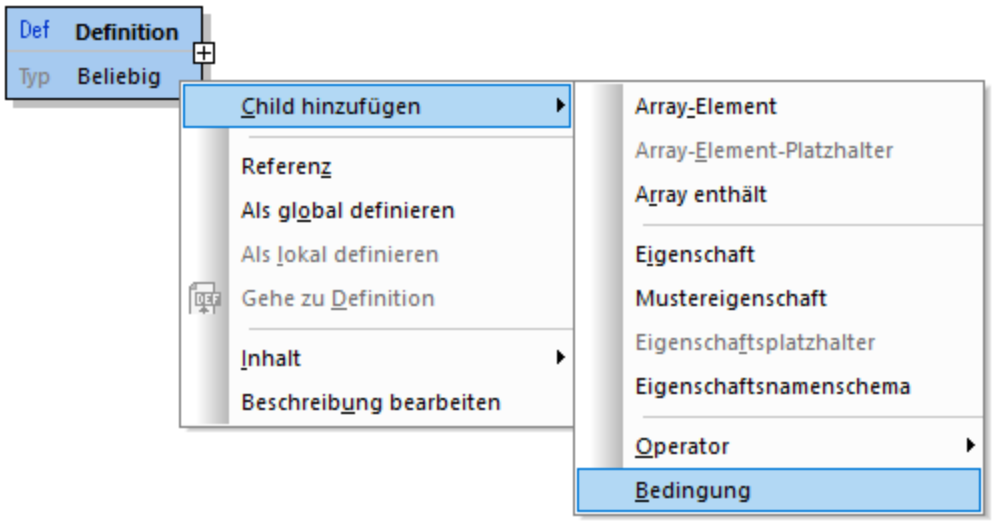


13.6.14 Bedingungen

Bedingungen sind eine neue Funktionalität in draft-07[727]. Sie können damit festlegen, dass die Validierungseinschränkungen je nach bestimmten Aspekten des Objekts, wie z.B. seinem Typ und/oder zusätzlichen typspezifischen Einschränkungen, unterschiedlich sein sollen.

Hinzufügen einer Bedingung

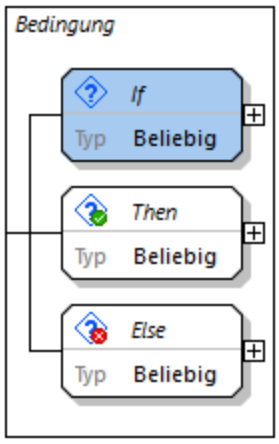
Sie können zu jeder Definition über das Kontextmenü der Definition (*siehe Abbildung unten*) eine Bedingung hinzufügen. Um das Untermenü der Bedingung aufzurufen, klicken Sie mit der rechten Maustaste auf die Definition, zu der Sie eine Bedingung hinzufügen möchten, und wählen Sie anschließend den Befehl **Child hinzufügen | Bedingung**.



Einrichten einer bedingten Validierung

Die Bedingung wird als Kasten mit den folgenden drei Elementen hinzugefügt: **`If-Then-Else`** *(siehe Abbildung unten)*.

© 2016-2022 Altova GmbH

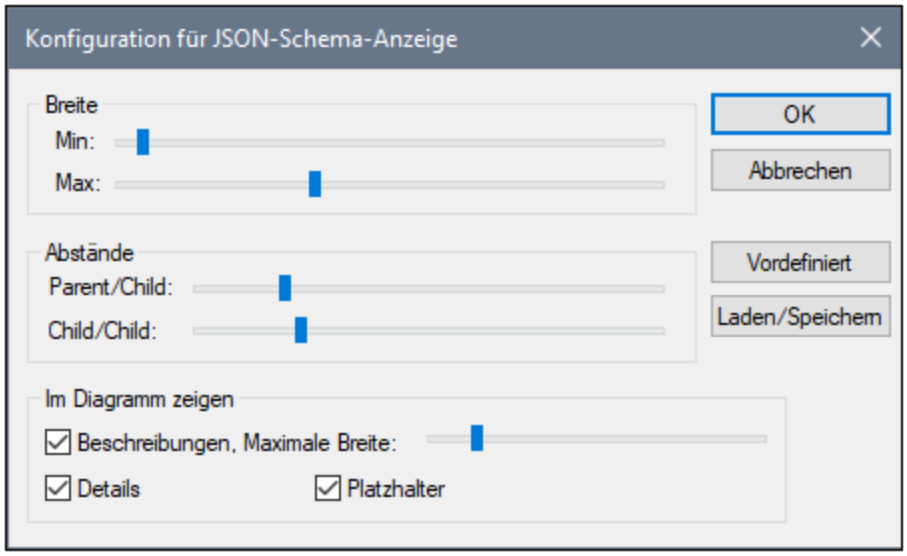


Um eine bedingte Validierung einzurichten, gehen Sie folgendermaßen vor:

1. Definieren Sie die Bedingung im Feld *If,* indem Sie zuerst in der Eingabehilfe "Details" einen Typ und anschließend in der Eingabehilfe "Constraints" einen typbasierten Constraint auswählen.
2. Definieren Sie im Feld *Then* die Validierungsanforderungen für den Fall, dass die (im Feld *If* definierte) Bedingung erfüllt wird.
3. Definieren Sie im Feld *Else* die Validierungsanforderungen für den Fall, dass die (im Feld *If* definierte) Bedingung nicht erfüllt wird.

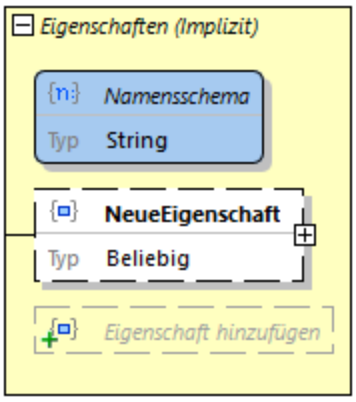
13.6.15 Konfigurieren der Design-Ansicht

Wenn das Hauptfenster im Modus der Design-Ansicht angezeigt wird, können Sie mit dem Menübefehl **Schema-Design | Ansicht konfigurieren** das Dialogfeld "Konfiguration anzeigen" aufrufen (*Abbildung unten*). Hier können Sie das Aussehen der Design-Ansicht konfigurieren.



Sie können die folgenden Aspekte der Design-Ansicht konfigurieren:

- *Breite:* Mit Hilfe von zwei Schiebereglern können Sie die minimale bzw. maximale Breite von Kästen in der Design-Ansicht festlegen. Zusammen definieren die beiden Schieberegler die zulässige Breite von Kästen.
- *Parent/Child-Abstände:* Definiert den horizontalen Abstand zwischen den einzelnen Ebenen in der Hierarchie.
- *Child/Child-Abstände:* Definiert den vertikalen Abstand zwischen den Kästen.
- *Breite von Beschreibungen:* Definiert die Breite von Beschreibungszeilen. Wenn die Länge des Texts diese Breite übersteigt, wird der Text auf die nächste Zeile umbrochen.
- *Detailanzeige:* Die Details von Beschreibungen können je nachdem, ob diese Option aktiviert ist oder nicht, angezeigt oder ausgeblendet werden.
- *Anzeige von Platzhaltern:* Platzhalter sind Elemente, die noch nicht definiert wurden, d.h. sie stehen für potenzielle Elemente. Mit dieser Option wird definiert, ob die Anzeige von Platzhaltern aktiviert oder deaktiviert ist. Es gibt auch eine Symbolleisten-Schaltfläche dafür. So ist z.B. das Element *Eigenschaft hinzufügen* in der Abbildung unten ein Platzhalter.



Anmerkung: Der Menübefehl **Ansicht konfigurieren** steht nur im Modus Design-Ansicht[726] zur Verfügung. In diesem Modus wird die genaue Definition eines Objekts angezeigt. Im Definitionsübersichtsraster[726] steht der Befehl nicht zur Verfügung.

13.6.16 Generieren von JSON-Schema-Dokumentation

Wenn ein JSON-Schema das aktive Dokument ist, können Sie durch Auswahl des Befehls **Schema-Design | Dokumentation generieren** Dokumentation zu diesem Schema generieren. Dokumentation kann in Form einer HTML-, MS Word- oder RTF-Datei ausgegeben werden. Sie können festlegen, welche Komponenten in der Dokumentation berücksichtigt werden sollen. Miteinander in Zusammenhang stehenden JSON-Komponenten werden in der generierten Dokumentation markiert, sodass Sie einfach zwischen diesen Komponenten navigieren können.

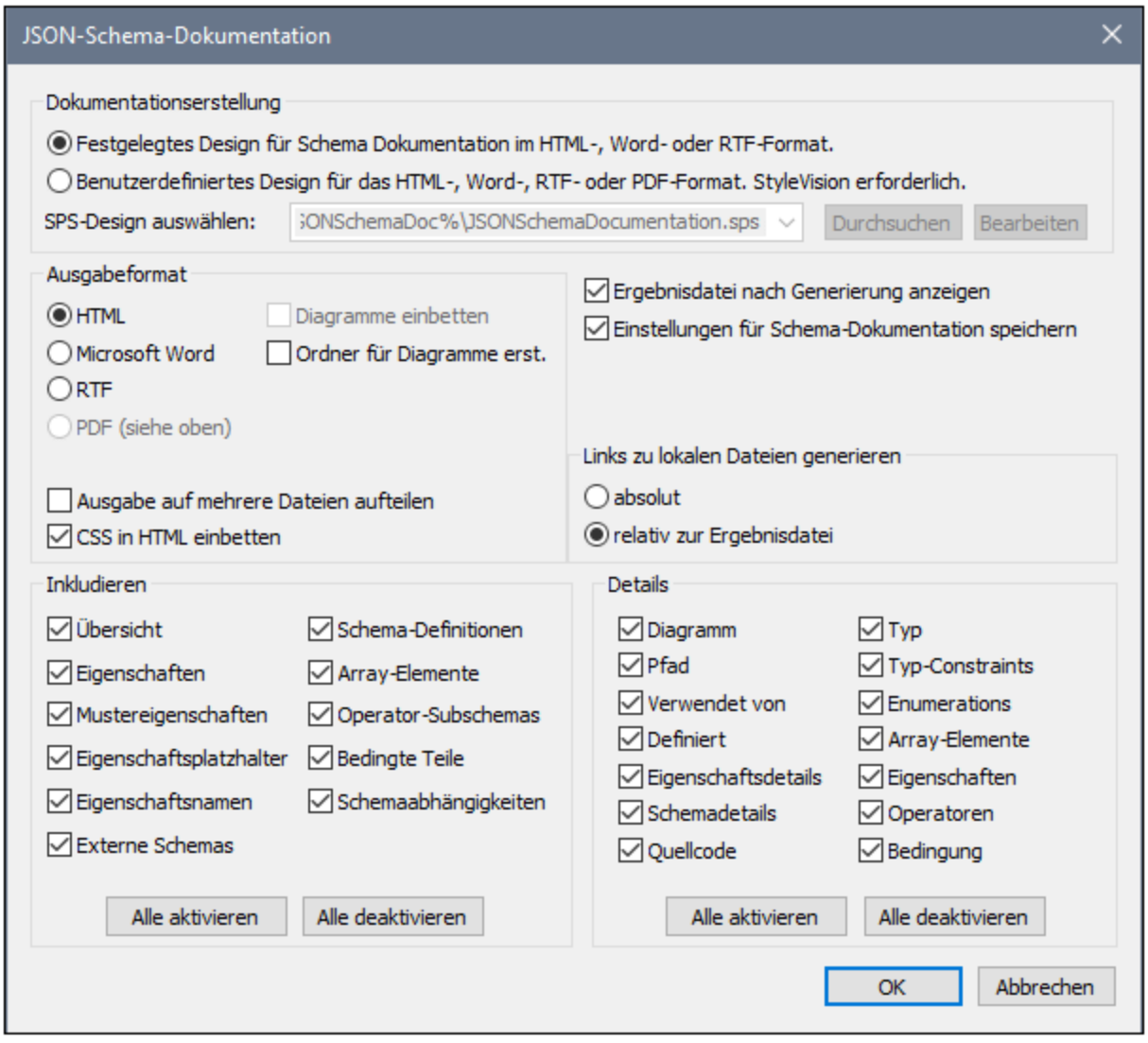
Anmerkung: Um Dokumentation im MS Word-Format generieren zu können, muss MS Word (Version 2000 oder höher) installiert sein.

Anleitung zum Generieren von JSON-Schema-Dokumentation

So generieren Sie Dokumentation zu einer JSON-Schema-Datei:

© 2016-2022 Altova GmbH

1. Machen Sie das JSON-Schema zum aktiven Dokument.
2. Wechseln Sie in die Schema-Ansicht.
3. Wählen Sie den Menübefehl **Schema-Design | Dokumentation generieren** aus. Daraufhin wird das Dialogfeld "JSON-Schema-Dokumentation" geöffnet (*Abbildung unten*).
4. Wählen Sie den gewünschten Ausgabetyp aus: HTML, MS Word oder RTF.
5. Wählen Sie die spezifischen Komponenten und Details aus, die in die Dokumentation inkludiert werden sollen und definieren Sie die anderen Optionen (*siehe weiter unten: Optionen zur JSON-Schema-Dokumentation*).



6. Klicken Sie auf **OK** und geben Sie den Namen der JSON-Schema-Dokumentationsdatei im Dialogfeld "Speichern unter" ein.

Optionen zur JSON-Schema-Dokumentation

Es stehen die folgenden Dokumentationsoptionen zur Auswahl:

- Als Designvorlage kann das vordefinierte (festgelegte) XMLSpy-Design oder ein in einer SPS-Datei gespeichertes benutzerdefiniertes Design verwendet werden. Eine Anleitung zur Verwendung eines benutzerdefinierten Designs finden Sie im Abschnitt Benutzerdefiniertes Design[1393].
- Das erforderliche Format wird im Bereich "Ausgabeformat" definiert: entweder HTML, Microsoft Word oder RTF. Die Dokumentation kann entweder als eine einzige Datei generiert oder auf mehrere Dateien aufgeteilt werden. Bei Generierung mehrerer Dateien entspricht jede Datei einer Komponente. Welche Komponenten in die Ausgabe inkludiert werden sollen, wird mit Hilfe der Kontrollkästchen im Bereich "*Inkludieren*" definiert.
- Die Option *Diagramme einbetten* ist für die Ausgabeoptionen MS Word und RTF aktiv. Wenn Sie diese Option aktivieren, werden Diagramme im PNG- oder EMF-Format in die Ergebnisdatei eingebettet. Andernfalls werden die Diagramme als PNG- oder EMF-Dateien erstellt und in der Ergebnisdatei über Objekt-Links angezeigt. Bei einer Ausgabe im HTML-Format werden alle Diagramme als dokumentexterne PNG-Dateien erstellt.
- Im Bereich *Inkludieren* wählen Sie aus, welche Elemente in die Dokumentation inkludiert werden sollen Mit der Option *Übersicht* werden alle Komponenten nach Typ geordnet im obersten Bereich der Datei aufgelistet. Wenn *Schema-Definitionen* deaktiviert ist, werden alle Child-Komponenten (d.h. alles mit Ausnahme von *Externen Schemas*) deaktiviert.
- Im Bereich *Details* werden die Einzelheiten aufgelistet, die für die einzelnen Komponenten inkludiert werden können. Wenn *Schema-Definitionen* deaktiviert ist, werden alle Details deaktiviert. Wählen Sie die Details aus, die in die Dokumentation inkludiert werden sollen.
- Die Option *Ergebnisdatei anzeigen* ist für alle drei Ausgabeoptionen aktiv. Wenn diese Option aktiviert ist, werden die Ergebnisdateien in der Browser-Ansicht (HTML-Ausgabe), der MS Word-Ansicht (MS Word-Ausgabe) und der Standardapplikation für `.rtf`-Dateien (RTF-Ausgabe) angezeigt.

© 2016-2022 Altova GmbH

13.7 Validieren von JSON-Dokumenten

XMLSpy enthält einen JSON-Validierungsprozessor, mit dem folgende Validierungen durchgeführt werden können:

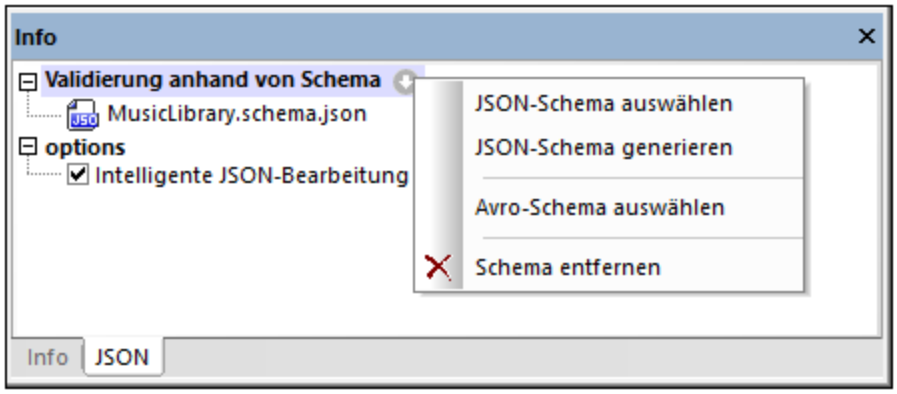
- *Wenn ein JSON-Schema das aktive Dokument ist:* Validiert das JSON-Schema anhand der entsprechenden JSON-Schema-Spezifikation (für die keine zusätzliche Schema-Zuweisung benötigt wird); die Schema-Version wird durch das Schlüsselwort `$schema` 682 angegeben; die Validierung kann in jeder der folgenden drei Ansichten durchgeführt werden: Text 685, Grid 690 und JSON-Schema 726).
- *Wenn ein JSON-Instanzdokument das aktive Dokument ist:* Validiert das JSON-Instanzdokument anhand eines JSON-Schemas; Das Schema wird der JSON-Instanz, wie unten beschrieben, zugewiesen. Die JSON-Instanzvalidierung kann in der Text 685- oder der Grid-Ansicht 690 erfolgen.
- *Wenn ein JSON5-Instanzdokument das aktive Dokument ist:* Validiert das JSON-Instanzdokument anhand eines JSON-Schemas; Das Schema wird der JSON5-Instanz, wie unten beschrieben, zugewiesen. Die JSON5-Instanzvalidierung kann in der Text- 685 oder der Grid-Ansicht 690 erfolgen.

Avro-Validierung (nur Enterprise Edition)
Avro-Daten und Avro-Schema-Dokumente können wie JSON-Dokumente in der Text- 685 und Grid-Ansicht 690 validiert werden:

- *Wenn eine Avro-Dateninstanz im JSON-Format das aktive Dokument ist:* Validiert die Avro-Instanz anhand eines Avro-Schemas; Das Schema wird der JSON5-Instanz, wie unten beschrieben, zugewiesen.
- *Wenn ein Avro-Schema das aktive Dokument ist:* Validiert das Avro-Schema-Dokument anhand der Avro-Schema-Spezifikation (es wird keine Schema-Zuweisung benötigt); Die Validierung kann in der Text 685- oder der Grid-Ansicht 690 erfolgen.

Zuweisen eines JSON- oder Avro-Schemas zu einer JSON-Instanz

Damit eine JSON-Instanz anhand eines JSON- oder Avro-Schemas validiert werden kann, muss dem aktiven Instanzdokument ein Schema zugewiesen werden. Diese Zuweisung erfolgt im Info-Fenster des aktiven JSON-Instanzdokuments (*Abbildung unten;*) oder über das Dialogfeld Projekteigenschaften 1344 (Option *Validieren mit*). Beachten Sie, dass JSON5-Instanzdokumente anhand von JSON-Schemas validiert werden.



Klicken Sie im Abschnitt "JSON" des Info-Fensters auf das Pfeilsymbol neben *Validierung anhand von Schema* und wählen Sie im daraufhin angezeigten Menü den Befehl **JSON-Schema auswählen** oder **Avro-Schema auswählen** (*siehe Abbildung oben*). Navigieren Sie zum gewünschten Schema und klicken Sie auf **OK**. Das Schema wird dem aktiven JSON-Instanzdokument nun zugewiesen und der Dateiname des Schemas wird im

Info-Fenster angezeigt. Wenn das JSON-Instanzdokument leer ist, wird die JSON-Instanz aufgrund der Zuweisung eines JSON-Schemas oder Avro-Schemas zur Instanz automatisch auf Basis dieses Schemas mit Beispieldaten befüllt.

Um die Zuweisung zu entfernen, wählen Sie im selben Menü den Befehl **Schema entfernen** (*siehe Abbildung oben*).

Informationen zum Generieren von JSON-Schemas anhand einer JSON-Instanz finden Sie im Abschnitt Generieren eines JSON-Schemas anhand einer JSON-Instanz[774].

Validieren von Instanz- und Schemadokumenten

Wählen Sie den Befehl **XML | XML validieren (F8)** oder klicken Sie auf die Symbolleisten-Schaltfläche **Validieren** , um das aktive JSON-Dokument (Instanz oder Schema) oder Avro-Schema zu validieren. Wenn ein Instanzdokument validiert wird, muss diesem ein Schema-Dokument zugewiesen sein (*siehe oben*). Die Validierungsergebnisse werden im Fenster "Meldungen"[122] angezeigt. Fehler werden in der Zeilennummernleiste markiert. Falls es zu einem Fehler einen Smart Fix-Vorschlag gibt, wird in der Zeile mit dem Fehler ein Glühbirnensymbol angezeigt. Wenn Sie den Mauszeiger über das Symbol platzieren, erscheint ein Popup-Fenster, in dem Smart Fix-Vorschläge aufgelistet werden. Wählen Sie einen Vorschlag aus, um ihn sofort anzuwenden.

Anmerkung: Die oben beschriebenen Validierungsfehlermarkierungen und Smart Fix-Vorschläge, werden erst bei Ausführung des Befehls **XML | Validieren (F8)** aktualisiert; sie werden nicht im Hintergrund aktualisiert. Sie müssen den Befehl **Validieren (F8)** erneut ausführen, um zu überprüfen, ob der Fehler wirklich behoben wurde.

Um vom Instanzdokument aus zum Schema-Dokument zu springen, doppelklicken Sie im Fenster "Info" auf das Schema (*siehe Abbildung oben*) oder wählen Sie den Befehl **DTD/Schema | Gehe zu Schema**. Um direkt zur Schema-Definition eins JSON-Schlüsselworts oder -Objekts zu gehen, wählen Sie das Schlüsselwort oder Objekt im Instanzdokument aus und wählen Sie **DTD/Schema | Gehe zu Definition**.

Sie können mit dem Befehl **Validieren** auch einen Projektordner[1080] mit JSON-Dateien validieren.

Bei Änderung validieren

Der *Bei Bearbeitung validieren*-Modus ist standardmäßig aktiviert. Wenn dieser Modus aktiv ist, werden, während Sie ein Dokument in der JSON-Grid-Ansicht bearbeiten, Wohlgeformtheits- und Validierungsüberprüfungen durchgeführt. Damit ein JSON-Dokument (nicht nur auf Wohlgeformtheit geprüft, sondern auch) validiert werden kann, muss dem JSON-Dokument ein JSON-Schema zugewiesen sein. Fehler im Text werden rot angezeigt. Die Stelle im Dokument wird durch ein rotes Ausrufezeichen markiert. Nähere Informationen dazu finden Sie unter Validieren von JSON-Dokumenten[765].

Der *Bei Bearbeitung validieren*-Modus kann entweder (i) über den Menübefehl **XML | Bei Bearbeitung validieren**[1359], (ii) die Symbolleisten-Schaltfläche **Bei Bearbeitung validieren** oder (iii) die Option *Bei Bearbeitung* der Validierungseinstellungen des Dialogfelds "Optionen"[1603] aktiviert bzw. deaktiviert werden.

© 2016-2022 Altova GmbH

13.8 Einfügen von JSON-Fragmenten

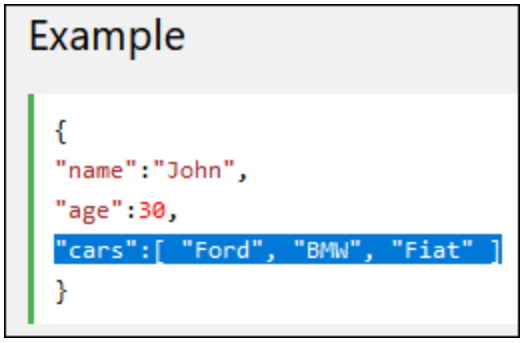
Sie können JSON-Fragmente aus anderen Applikationen oder Webseiten einfügen. Diese Fragmente können auf zwei verschiedene Arten eingefügt werden:

- Sie können diese mit der Maus (Drag-and-Drop) in die Text- oder Grid-Ansicht ziehen. Wenn Sie ein Fragment mittels Drag-and-Drop in die Grid-Ansicht ziehen, helfen Ihnen die intelligenten Informationen in den Drag Overlays[702] bei der Entscheidung, an welcher Stelle das Fragment eingefügt werden soll.
- Sie können diese in die Text- oder Grid-Ansicht kopieren (Kopieren und Einfügen).

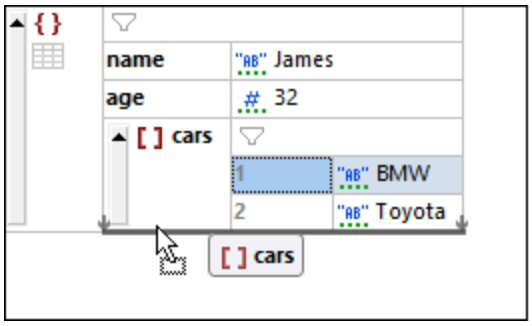
Beispiel

Im folgenden Beispiel wird gezeigt, wie ein Fragment einfach und schnell an der richtigen Stelle in einem JSON-Dokument hinzugefügt werden kann.

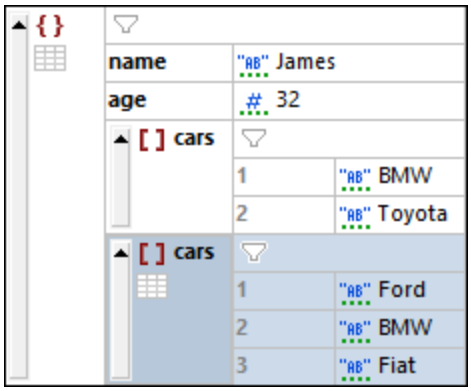
1. Das unten markierte Fragment (aus dem JSON-Tutorial unter w3schools.com) wird ausgewählt. Es handelt sich dabei um ein Array namens **cars**.



2. In der Abbildung unten sehen Sie die Grid-Ansicht eines JSON-Dokuments, das ein ähnliches **cars**-Array enthält. Wenn das Fragment aus der Webseite in das bereits bestehende **cars**-Array gezogen wird, erscheint ein Drag Overlay[702] mit der Information, dass das mit der Maus gezogene JSON-Fragment als neues Array namens **cars** unterhalb des vorhandenen Array eingefügt wird.



3. Wenn Sie die Maustaste loslassen, wird das Fragment genau an der gewünschten Stelle eingefügt (*Abbildung unten*).



© 2016-2022 Altova GmbH

13.9 JSON-Transformationen mit XSLT/XQuery

JSON-Zuordnungen, -Arrays und -Objekte können das Ziel von **XPath/XQuery 3.1**-Ausdrücken sein. JSON-Dokumente können daher mit Hilfe von **XSLT 3.0-, XQuery 3.1- und XQuery Update 3.0**-Dokumenten mittels der integrierten Prozessoren von XMLSpy transformiert werden.

Es stehen die folgenden Funktionalitäten zur Verfügung:

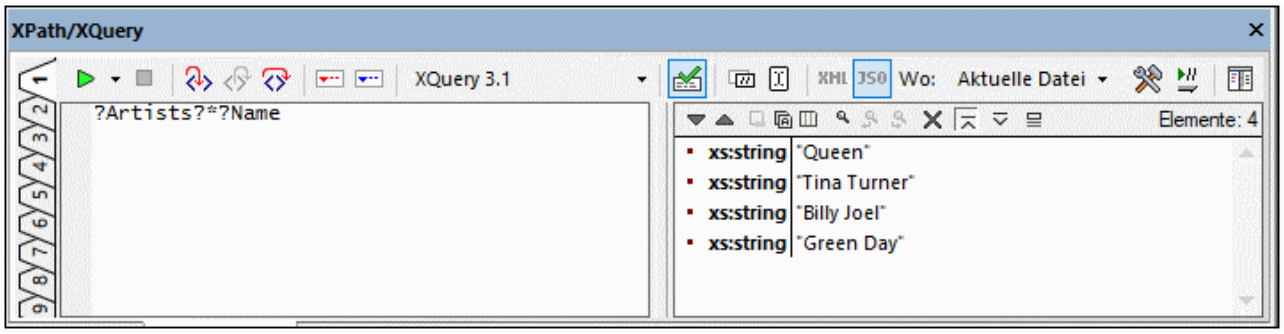
- Ein aktives JSON-Dokument kann mit Hilfe von XPath/XQuery 3.1-Ausdrücken[769] über das XPath/XQuery-Ausgabefenster[124] abgefragt werden.
- Ein aktives JSON-Dokument kann mit Hilfe einer vom Benutzer ausgewählten XSLT- oder XQuery-Datei transformiert werden[770].
- Ein aktives XSLT- oder XQuery-Dokument kann an einer vom Benutzer ausgewählten JSON-Quelldatei ausgeführt werden[770].

Diese Funktionen werden weiter unten ausführlicher beschrieben. Informationen zur Erstellung von XQuery-Ausdrücken für JSON-Dokumente finden Sie im Abschnitt XQuery-Ausdrücke für JSON[772].

Anmerkung: Sie können JSON-Transformationen mit Hilfe der JSON-, XSLT- und XQuery-Dateien aus dem Ordner *JSON Examples* des **Examples**-Projekts aus Ihrem Applikationsordner `<%SPYEX-DIR%>\Examples\Examples.spp` ausprobieren.

Abfragen eines JSON-Dokuments über das XPath/XQuery-Fenster

JSON-Dokumente können durch Eingabe eines XPath/XQuery 3.1-Ausdrucks im XPath/XQuery-Ausgabefenster[124] (*siehe Abbildung unten*) abgefragt werden. Wählen Sie entweder das **XPath 3.1-** oder das **XQuery 3.1**-Symbol aus und stellen Sie sicher, dass sich das Fenster im JSON-Auswertungsmodus befindet (*Erklärung siehe unten*).



Die Informationen unten beziehen sich auf die Auswertung von JSON-Dokumenten im JSON-Auswertungsmodus. (Eine Übersicht über das XPath/XQuery-Fenster und ausführliche Informationen zu seiner Verwendung finden Sie im Abschnitt Ausgabefenster: "XPath/XQuery"[124].)

JSON-Auswertungsmodus
Beim JSON-Auswertungsmodus gelten die folgenden Bedingungen:

- Das XPath/XQuery-Fenster befindet sich entweder im XML- oder im JSON-Auswertungsmodus. Welcher Modus gerade **aktiv** ist, sehen Sie daran, dass die Schaltfläche des aktiven Modus markiert

erscheint. Siehe dazu die XML/JSON-Auswertungsmodus-Schaltflächen in der Abbildung oben. In der Abbildung befindet sich das Fenster im JSON-Auswertungsmodus.

- Beachten Sie, dass die Schaltflächen "XML" und "JSON" in der Abbildung oben ausgegraut sind, d.h. sie sind **deaktiviert**. Wenn die Schaltflächen deaktiviert sind, kann ihr Status - ob aktiviert oder deaktiviert - nicht geändert werden. Wenn die Schaltflächen hingegeben **aktiviert** (nicht ausgegraut) sind, kann der Auswertungsmodus des Fensters geändert werden.
- Der aktivierte/deaktivierte Status der XML/JSON-Auswertungsmodus-Schaltflächen hängt vom Geltungsbereich der Auswertung ab (dem Wert des Felds "*Wo*", *siehe Abbildung oben*). Die Werte für den Geltungsbereich der Auswertung werden zur Ermittlung des *aktivierten/deaktivierten* Status in zwei Gruppen eingeteilt: (i) Einzelne Datei (*aktuelle Datei*) und (ii) Mehrere Dateien (*Offene Dateien, Projekt, Ordner*).
- Wenn als Geltungsbereich der Auswertung eine einzelne Datei (*aktuelle Datei*) ausgewählt ist (*wie in der Abbildung oben*), so wird der Modus des Fensters (JSON oder XML) anhand der Dateierweiterung[1606] ermittelt. Entweder die Datei ist JSON-konform[1606], dann ist der JSON-Auswertungsmodus **aktiviert**, oder die Datei ist nicht JSON-konform[1606] und der XML-Auswertungsmodus ist eingeschaltet. Da der Dateityp der einzelnen Datei bekannt ist, ist der entsprechende Auswertungsmodus aktiviert und beide Schaltflächen sind **deaktiviert**, sodass der Modus nicht geändert werden kann.
- Wenn die Option für mehrere Dateien (*Offene Dateien, Projekt, Ordner*) ausgewählt ist, so sind beide Auswertungsmodus-Schaltflächen **aktiviert** und der Benutzer kann auswählen, welchen Modus er aktivieren möchte (JSON oder XML). Der Standardauswertungsmodus für den Geltungsbereich für mehrere Dateien ist XML.
- Im XML-Auswertungsmodus werden XML-konforme Dateien[1606] verarbeitet und JSON-Dateien übersprungen.
- Im JSON-Auswertungsmodus werden JSON-konforme Dateien[1606] verarbeitet und XML-Dateien übersprungen.
- JSON-Ausdrücke können auch im Debug-Modus[591] abgefragt werden.

Transformieren eines JSON-Dokuments mit XSLT/XQuery

Um ein aktives JSON-Dokument[115] mit Hilfe eines XSLT 3.0-, XQuery 3.1- oder XQuery Update 3.0-Dokuments zu transformieren, gehen Sie folgendermaßen vor:

- *XSLT 3.0-Tansformation:* Klicken Sie auf **XSL/XQuery | XSL-Transformation**, navigieren Sie zur XSLT 3.0-Datei und klicken Sie auf **OK**.
- *XQuery 3.1- oder XQuery Update 3.0-Transformation:* Klicken Sie auf **XSL/XQuery | XQuery/Update-Ausführung**, navigieren Sie zur XQuery 3.1- oder XQuery Update 3.0-Datei und klicken Sie auf **OK**.

Die transformierten Dokumente werden generiert und können direkt in XMLSpy angezeigt werden.

Anmerkung: Der XSLT/XQuery Debugger[540] kann von einem JSON-Dokument aus gestartet werden, Breakpoints und Tracepoints können jedoch nur im XSLT- oder XQuery-Dokument gesetzt werden.

Bereitstellen eines JSON-Quelldokuments für ein XSLT/XQuery-Dokument

Um ein aktives XSLT- oder XQuery-Dokument[115] an einer JSON-Quelldatei auszuführen, gehen Sie folgendermaßen vor:

- *Aktives XSLT 3.0-Dokument:* Klicken Sie auf **XSL/XQuery | XSL-Transformation**, navigieren Sie zur JSON-Datei und klicken Sie auf **OK**.
- *Aktives XQuery 3.1- oder XQuery Update 3.0-Dokument:* Klicken Sie auf **XSL/XQuery | XQuery/Update-Ausführung**, navigieren Sie zur JSON-Datei und klicken Sie auf **OK**.

© 2016-2022 Altova GmbH

Die transformierten Dokumente werden generiert und können direkt in XMLSpy angezeigt werden.

Anmerkung: Der XSLT/XQuery Debugger[540] kann von einem XSLT- oder XQuery-Dokument aus gestartet werden und es kann ein JSON-Dokument als Input für die Debug-Sitzung zugewiesen werden. Breakpoints und Tracepoints können jedoch nur im XSLT- oder XQuery-Dokument gesetzt werden.

13.10 XQuery-Ausdrücke für JSON

XQuery 3.1-Ausdrücke für JSON

Da in JSON-Datenstrukturen oft Objekte und Arrays verwendet werden, wird zum Identifizieren von Nodes in JSON-Objekten (bei denen es sich aus Sicht von XQuery im Grunde um Zuordnungen handelt) und JSON-Arrays der XQuery 3.1 Lookup-Operator `?` verwendet. Diese Methode der Node-Suche unterscheidet sich von der Methode, mit der Pfadausdrücke zum Suchen von Nodes in XML-Dokumenten geschrieben werden. In diesen Dokumenten werden Schritte in einem Pfadausdruck (z.B: `items/*`) mit Hilfe des Schrägstrich-Operators `/` verbunden. In XQuery-Ausdrücken für JSON wird der Schrägstrich-Operator nicht zur Suche von Nodes verwendet.

Beispiele für XQuery-Ausdrücke für JSON

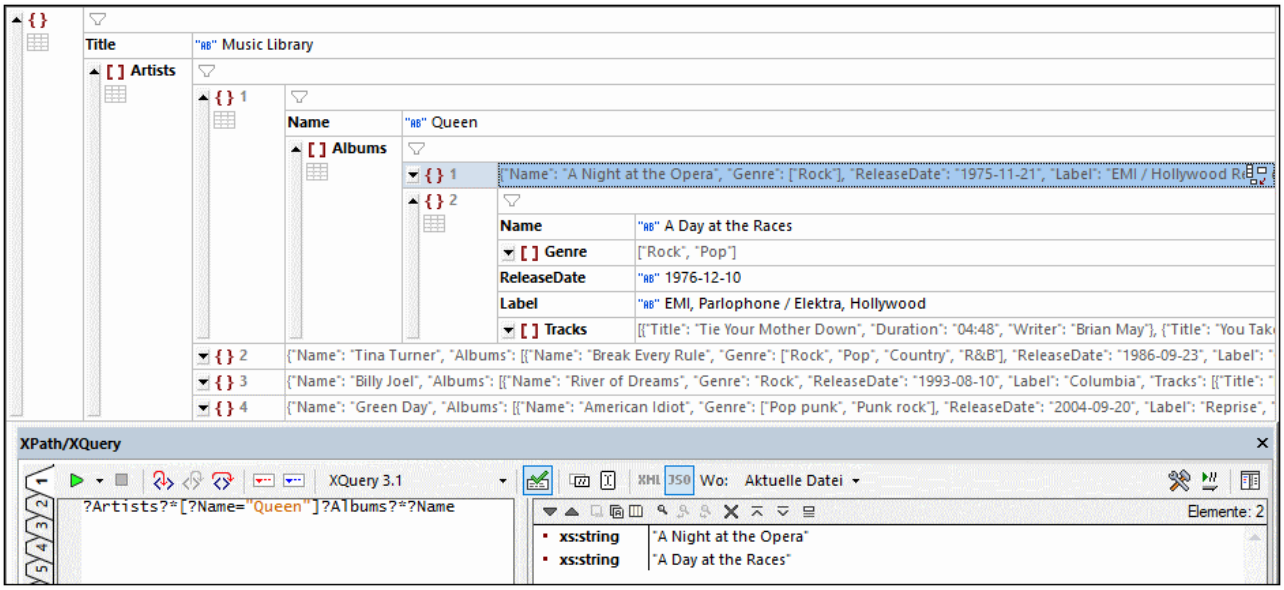
`?items?*`
Bedeutet: Suche den Child-Node `items` und anschließend alle diesem untergeordnete Nodes. Beachten Sie, dass erwartet wird, dass `items` ein Child-Node des Kontext-Node ist.

`?Artists?1?Albums?2?Name`
Bedeutet: Suche den Child-Node `Artists` und anschließend dessen ersten Child-Node. Suche innerhalb dieses Node den Child Node `Albums` und anschließend dessen zweiten Child Node. Gib nun den Node `Name` dieses zweiten Child Node zurück.

`?Tracks?*[contains(?Writer, 'Brian')]`
Bedeutet: Suche den Child Node `Tracks` und anschließend alle seine untergeordneten Nodes. Suche während der Suche der untergeordneten Nodes die einzelnen `Writer` Node Children und wähle nur diejenigen aus, die den String `'Brian'` enthalten. Beachte, dass dieser Ausdruck drei Lookup-Operatoren enthält. Jeder davon wird in einem neue Schritt, in dem ein Nodeset gesucht werden muss, verwendet.

`?Artists?*[?Name="Queen"]?Albums?*?Name`
Bedeutet: Suche im Root-Objekt den Child Node `Artists` und anschließend alle dessen Children, die einen `Name` Node mit dem Wert `"Queen"` haben. Suche in diesen Nodes alle untergeordneten Album-Nodes und anschließend deren untergeordnete Nodes. Suche in diesen untergeordneten Nodes die jeweiligen `Name` Nodes und gib diese zurück. In der Abbildung unten sehen Sie diesen Ausdruck im XPath/XQuery-Fenster[582] zusammen mit dem JSON-Zieldokument in der JSON Grid-Ansicht[690].

© 2016-2022 Altova GmbH


Title
"Music Library
Artists
Name
"Queen
Albums
{"Name": "A Night at the Opera", "Genre": ["Rock"], "ReleaseDate": "1975-11-21", "Label": "EMI / Hollywood Re
Name
"A Day at the Races
Genre
["Rock", "Pop"]
ReleaseDate
"1976-12-10
Label
"EMI, Parlophone / Elektra, Hollywood
Tracks
{"Name": "Tina Turner", "Albums": [{"Name": "Break Every Rule", "Genre": ["Rock", "Pop", "Country", "R&B"], "ReleaseDate": "1986-09-23", "Label":
{"Name": "Billy Joel", "Albums": [{"Name": "River of Dreams", "Genre": "Rock", "ReleaseDate": "1993-08-10", "Label": "Columbia", "Tracks": [{"Title":
{"Name": "Green Day", "Albums": [{"Name": "American Idiot", "Genre": ["Pop punk", "Punk rock"], "ReleaseDate": "2004-09-20", "Label": "Reprise",
XPath/XQuery
XQuery 3.1
Wo: Aktuelle Datei
?Artists?*[?Name="Queen"]?Albums?*?Name
Elemente: 2
xs:string
"A Night at the Opera"
xs:string
"A Day at the Races"

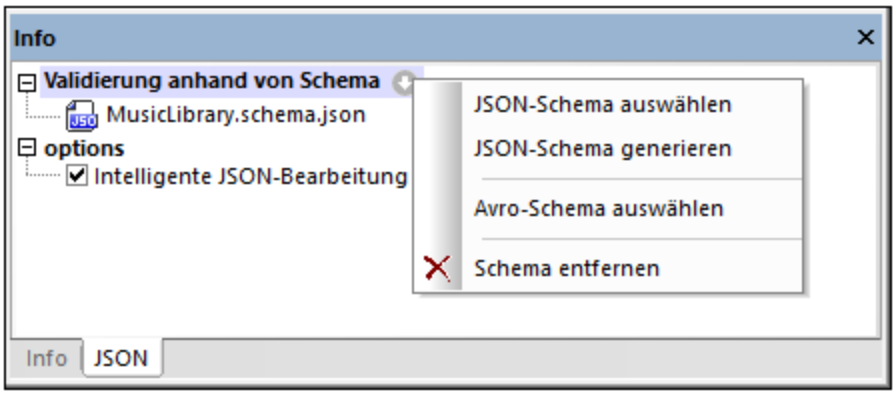
13.11 Generieren eines JSON-Schemas anhand einer JSON-Instanz

XMLSpy kann ein JSON-Schema anhand eines JSON-Instanzdokuments (bzw. eines JSON5-Instanzdokuments) generieren. Diese Funktion ist sehr nützlich, da Sie dadurch sofort ein Schema, das auf der vorhandenen JSON-Instanz basiert, zur Verfügung haben. Dadurch sparen Sie sich die Mühe, manuell ein Schema von Grund auf neu erstellen zu müssen. Sie können das generierte Schema anschließend Ihren Anforderungen entsprechend anpassen oder erweitern.

Generieren des Schemas

Um ein JSON-Schema anhand einer JSON-Instanz zu generieren, gehen Sie folgendermaßen vor:

1. Machen Sie das JSON-Instanzdokument zum aktiven Dokument.
2. Klicken Sie im Info-Fenster auf dem Register "JSON" auf das Pfeilsymbol neben *Validierung anhand von Schema* und wählen Sie im daraufhin angezeigten Menü den Befehl **JSON-Schema generieren** (*siehe Abbildung unten; Avro-Unterstützung steht nur in der Enterprise Edition zur Verfügung*). Alternativ dazu können Sie auch den Menübefehl **DTD/Schema | DTD/Schema generieren** auswählen.

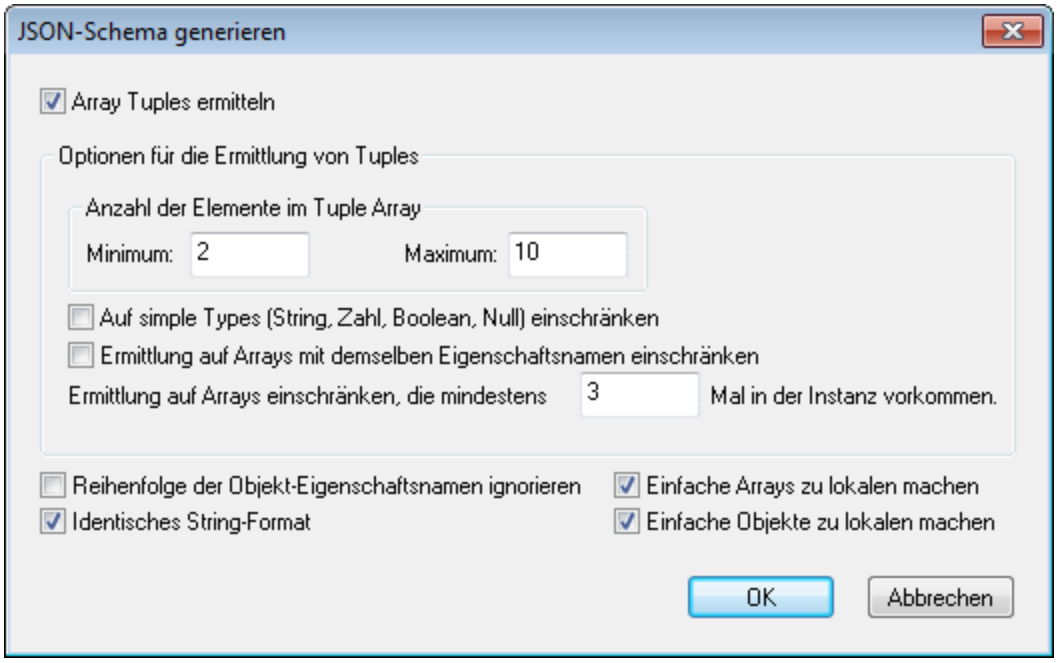


3. Bearbeiten Sie die Einstellungen nun Ihren Wünschen entsprechend (*nähere Informationen siehe unten*) im daraufhin angezeigten Dialogfeld "JSON-Schema generieren" (*siehe nächster Abschnitt unten*). Klicken Sie auf **OK**, sobald Sie damit fertig sind.
4. Sie werden aufgefordert, einen Pfad und Dateinamen für das generierte JSON-Schema anzugeben. Sobald Sie auf **Speichern** klicken, wird das Schema generiert und zum aktiven Dokument. Die generierte Schemadatei wird im Instanzdokument als das Schema zugewiesen, anhand dessen die Validierung durchgeführt werden soll; frühere Zuweisungen werden überschrieben. Um die Zuweisung zu ändern, verwenden Sie den Kontextmenübefehl **JSON-Schema auswählen** 765 (*siehe Abbildung oben*).

Einstellungen zum Generieren des Schemas

Im Dialogfeld "JSON-Schema generieren" (*Abbildung unten*) können Sie die Optionen für die Schemagenerierung definieren (*Abbildung unten*). Informationen dazu, wie Sie dieses Dialogfeld aufrufen, finden Sie im vorherigen Abschnitt 774.

© 2016-2022 Altova GmbH



Array Tuples ermitteln

Ein Array Tuple ist die Sequenz der Elemente in einem Array. Das folgende Array hat z.B. ein Tuple mit drei Elementen: `[ 1, 2, "abc" ]`. Sie können für die Validierung von Arrays im Schema definieren, ob die Reihenfolge und der Datentyp von Array (Tuple)-Elementen berücksichtigt werden soll oder nicht. Wenn die Option *Array Tuples ermitteln* aktiviert ist (*siehe Abbildung oben*), so werden die Reihenfolge und der Datentyp von Elementen ermittelt. Anhand der ermittelten Tuples wird im Schema eine entsprechende Definition erstellt. Zu dieser Einstellung gibt es die folgenden Optionen:

- *Anzahl der Tuple-Elemente:* Sie können eine Mindest- und Maximalanzahl von Tuple-Elementen definieren. Wenn ein Tuple in einer Instanz eine Anzahl von Elementen im angegebenen Bereich aufweist, so wird dieses Array erkannt und definiert.
- *Nur einfache Typen:* Bei der Erkennung werden nur Tuples mit Elementen vom einfachen Typ (die atomaren Typen `String`, `Zahl`, `Ganzzahl`, `Boolean` und `Null`) bei der Erkennung berücksichtigt.
- *Arrays mit demselben Namen:* Bei der Erkennung werden nur Arrays, die als Werte von Eigenschaften mit demselben Namen definiert sind, berücksichtigt. So sind z.B. im folgenden JSON-Datenfragment die durch rot hinterlegte eckigen Klammern markierten Arrays alle Werte von Eigenschaften mit dem Namen `a1` (blau angezeigt): `{"object1": [ { "a1": [ 1, 2, "abc" ] }, { "a1": [ 3, 4, "def" ] }, { "a1": [ 5, 6, "ghi" ] } ] }`.
- *Mindestanzahl von Arrays:* Sie können eine Mindestanzahl von Arrays definieren, bei der die Array-Ermittlung aktiviert wird.

Andere Einstellungen

- *Reihenfolge der Objekt-Eigenschaftsnamen ignorieren:* Wenn diese Option deaktiviert ist, wird die Reihenfolge der Eigenschaften eines Objekts überprüft und möglichst ähnlich wiedergegeben. Andernfalls wird die Reihenfolge nicht überprüft.
- *Identisches String-Format:* Im Schema kann festgelegt werden, dass String-Datentypen ein bestimmtes Format haben müssen. Wenn diese Option aktiv ist, so versucht XMLSpy das String-Format zu ermitteln und eine Formatdefinition für Strings hinzuzufügen, wo dies möglich ist.

- *Einfache Arrays zu lokalen machen:* Ein einfaches Array ist ein Array, in dem alle Elemente denselben einfachen Datentyp haben. Wenn die Option aktiv ist, werden alle einfachen Arrays lokal im Schema definiert, anstatt globale Definitionen zu verwenden, die lokal referenziert werden.
- *Einfache Objekte zu lokalen machen:* Ein einfaches Objekt ist ein Objekt, in dem alle Eigenschaftswerte denselben einfachen Datentyp haben. Wenn die Option aktiv ist, werden alle einfachen Objekte lokal im Schema definiert, anstatt globale Definitionen zu verwenden, die lokal referenziert werden.

Anmerkung: Sie können lokale Definitionen einzelner Objekte und Arrays nach Erstellung des JSON-Schemas zu globalen machen und umgekehrt. Nähere Informationen dazu finden Sie im Abschnitt Globale und lokale Definitionen 733.

 © 2016-2022 Altova GmbH

13.12 Generieren einer JSON-Instanz anhand eines JSON-Schemas

Sie können eine JSON-Instanz anhand eines JSON-Schemas generieren, wenn das JSON-Schema die aktive Datei in der Textansicht, Grid-Ansicht oder Schema-Ansicht ist. Klicken Sie auf **DTD/Schema | XML/JSON-Beispieldatei generieren** 1381. Beachten Sie, dass mit diesem Befehl ein JSON-Dokument, nicht aber ein JSON5-Dokument generiert wird.

13.13 Konvertierung zwischen JSON und XML

Es stehen die folgenden Konvertierungsoptionen zur Verfügung:

- XML-Instanz in JSON konvertieren[1493]: Wenn ein XML-Instanzdokument das aktive Dokument ist, können Sie auswählen, ob ein JSON- oder ein JSON5-Instanzdokument generiert werden soll. Verwenden Sie den Befehl **Konvertieren | XML-Instanz von/in JSON konvertieren**[1493].
- JSON-Instanz in XML konvertieren[1493]: Wenn ein JSON/JSON5-Instanzdokument das aktive Dokument ist, wird anhand der JSON-Instanz ein XML-Instanzdokument generiert, wenn Sie den Befehl **Konvertieren | XML-Instanz von/in JSON konvertieren**[1493] auswählen.
- XML-Schema in JSON-Schema konvertieren[1497]: Wenn ein XML-Schema-Dokument das aktive Dokument ist, wird anhand des XML-Schemas ein JSON-Schema-Dokument generiert, wenn Sie den Befehl **Konvertieren | XML-Schema von/in JSON-Schema konvertieren**[1497] auswählen.
- JSON-Schema in XML-Schema konvertieren[1497]: Wenn ein JSON-Schema-Dokument das aktive Dokument ist, wird anhand des JSON-Schemas ein XML-Schema-Dokument generiert, wenn Sie den Befehl **Konvertieren | XML-Schema von/in JSON-Schema konvertieren**[1497] auswählen.

Alle diese Konvertierungen sind sowohl in der Text- als auch der Grid-Ansicht aktiv. Klicken Sie auf die obigen Links, um zu den Beschreibungen der entsprechenden Funktion zu gelangen.

© 2016-2022 Altova GmbH

14 Avro, Avro-Schema

Apache Avro™ ist ein System zum Serialisieren von Daten in ein kompaktes Binärformat. Eine Avro-Datenstruktur ist in einem Avro-Schema definiert, welches im JSON-Format geschrieben ist. Ein Avro-Dokument wird in tatsächlichen Anwendungsszenarien normalerweise als Binärdatei, die nicht nur die Avro-Datenstruktur, sondern auch das zum Definieren dieser Strukturen verwendete Avro-Schema enthält, serialisiert. Die Avro-Binärdatei enthält somit sowohl die Daten als auch die Definition der Datenstruktur (das Avro-Schema). Avro-Daten können jedoch auch als JSON serialisiert werden. In diesem Fall referenzieren die Avro-Daten (in einer JSON-Datei) ein externes Avro-Schema.

XMLSpy unterstützt Apache Avro™ 1.8.1.

XMLSpy bietet die folgenden Avro-Unterstützung:

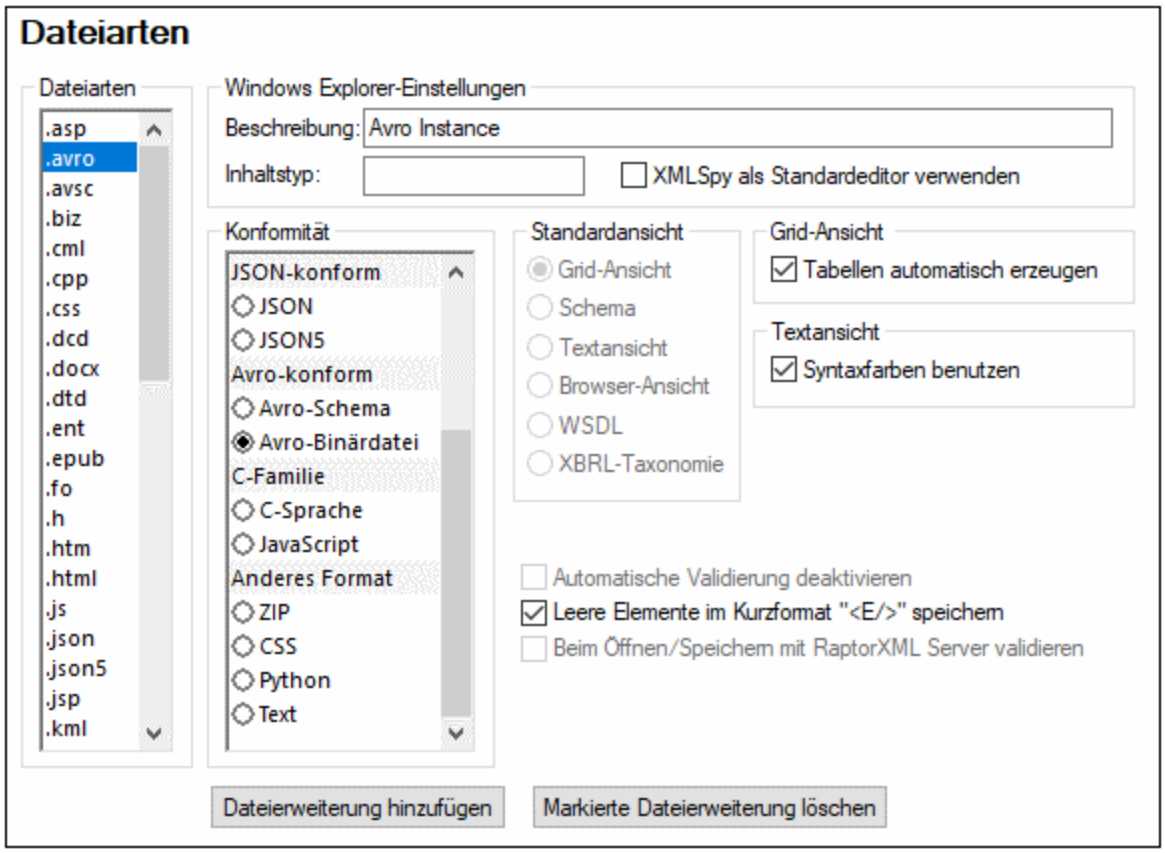
- Sie können Avro-Daten (als `.json` JSON-Dokumente) in der Text-[685] und der Grid-Ansicht[690] bearbeiten; beide Ansichten bieten intelligente Bearbeitungsfunktionen. Dem Datendokument kann ein Avro-Schema zugewiesen werden, anhand dessen es validiert werden kann.
- Sie können Avro-Schemas (als `.avsc` Avro-Schema-Dokumente) in der Text-[685] und der Grid-Ansicht[690] bearbeiten. Avro-Schemas können anhand der Avro-Schema-Spezifikation validiert werden. Die Ansichten bieten intelligente Bearbeitungsfunktionen.
- Sie können Avro-Binärdateiinstanzen (`.avro`-Dateien) in der Avro-Ansicht[967], in der Avro-Datenblöcke in einem tabellarischen Raster angezeigt werden, aufrufen.

Die RaptorXML Editionen von Altova bieten zusätzliche Avro-Unterstützung:

- Avro-Datenvalidierung (JSON-serialisiert; `.json`-Datei) (anhand eines Avro-Schemas)
- Avro-Datenvalidierung (Binärdatei-serialisiert; `.avro`-Datei)
- Avro-Schemavalidierung (normalerweise `.avsc`-Datei) (anhand der Avro-Schema-Spezifikation)
- Extraktion eines Avro-Schemas aus einer Avro-Binärdatei

Öffnen vorhandener Avro-Dokumente und Erstellen neuer Avro-Dokumente

Auf dem Register **Optionen | Dateiarten**[1606] (*Abbildung unten*) können Sie die Standardansicht festlegen, in der die verschiedenen Avro-Dokumente (JSON-Datenformat, Avro-Schema, Avro-Binärdatei) geöffnet werden sollen. Sie können jederzeit zwischen den verfügbaren Ansichten wechseln.



Dokumenttyp	Dateierweiterung	Konformität	Verfügbare Ansichten
Avro-Daten im JSON-Format	`.json`	JSON-konform | JSON	Textansicht, Grid-Ansicht
Avro-Schema	`.avsc`	Avro-konform | Avro-Schema	Textansicht, Grid-Ansicht
Avro-Daten in Binärdatei	`.avro`	Avro-konform | Avro-Binärdatei	Avro-Ansicht

Beachten Sie die folgenden Punkte:

- Vorhandene Dokumente und neue Dokumente eines ausgewählten Typs werden in der von Ihnen im Abschnitt *Dateiarten* ausgewählten Standardansicht geöffnet.
- Avro-Binärdateien können nur in der Avro-Ansicht[786], einer schreibgeschützten Ansicht, angezeigt werden. Wenn ein Dateityp als Avro-konform definiert ist, steht nur die Avro-Ansicht[786] zur Verfügung.
- Wenn XMLSpy Dateien mit einer bestimmten Dateierweiterung als einen der oben aufgelisteten Avro-Dokumenttypen lesen soll, fügen Sie diese neue Dateierweiterung hinzu und weisen Sie ihr die entsprechende Konformität zu.

© 2016-2022 Altova GmbH

- Um ein neues Dokument zu erstellen, klicken Sie auf **Datei | Neu** und wählen Sie den gewünschten Dokumenttyp aus. Da es sich bei Avro-Binärdateien um Binärdateien handelt, können diese natürlich nicht auf diese Art erstellt werden; sie haben in der Avro-Ansicht 786 nur Lesezugriff darauf.

14.1 Avro-Schema

In einem Avro-Schema ist die Struktur eines Avro-Datenblocks definiert. Es wird definiert, welche Datenfelder erwartet werden und wie die Werte repräsentiert werden. Informationen zu Avro-Schemas und der dazugehörigen Spezifikation finden Sie hier.

Beachten Sie die folgenden Punkte zu Avro-Schemas:

- Ein Avro-Schema ist im JSON-Format erstellt
- Bei einem Avro-Schema kann es sich um einen JSON-String, ein JSON-Objekt oder ein JSON-Array handeln
- Ein Avro-Schema kann vier Attribute enthalten: **`name`**, **`namespace`**, **`type`** und **`fields`**
- Es gibt acht primitive Datentypen: **`null`**, **`boolean`**, **`int`**, **`long`**, **`float`**, **`double`**, **`bytes`** und **`string`**
- Es gibt sechs komplexe Typen: **`records`**, **`enums`**, **`arrays`**, **`maps`**, **`unions`** und **`fixed`**
- Primitive Typen haben keine Attribute; jeder komplexe Typ hat seine eigene Gruppe von Attributen

Nähere Informationen zu Avro-Schemas finden Sie in der Avro-Schema-Spezifikation.

Beispiele

Weiter unten finden Sie einige einfache Beispiele für Avro-Schemas, jedes davon mit dem dazugehörigen Avro-Datenfragment im JSON-Format. Beachten Sie, dass im Schema eine bestimmte Struktur definiert ist. In einigen Fällen ist die erzeugte Ausgabe eventuell kein gültiges JSON-Dokument, wenn die definierte Struktur mehrmals instantiiert wird. So kann etwas die Struktur eines JSON-Objekts in einem Schema definiert sein. Wenn das JSON-Objekt mehrmals instantiiert wird, könnte zwar jedes einzelne Objekt gemäß dem Avro-Schema gültig sein, doch wäre das gesamte Dokument kein gültiges JSON-Dokument, da kein Container-Objekt vorhanden ist. Wenn ein gültiges JSON-Dokument erforderlich ist, müssen Sie das Avro-Schema eventuell ändern, damit es als Array von JSON-Objekten validiert wird. Vergleichen Sie dazu Beispiel 4 mit Beispiel 5 weiter unten.

01: Avro-Schema als JSON-String

Dieses Schema ist ein einziger String und definiert, dass das Instanzdokument einen Wert enthalten muss, der den primitiven Avro (int)-Datentyp: **`"int"`** hat.

Gültiges Avro: **`2016`**
Ungültiges Avro: **`"2016"`**

02: Avro-Schema als JSON-Objekt

In diesem Schema ist genau dasgleiche wie im vorherigen Schema definiert, doch ist es ein JSON-Objekt. Das Instanzdokument muss ein Element enthalten, der ein Wert des primitiven Avro-Datentyps (int) ist:

```
{
   "type": "int"
}
```

Gültiges Avro: 2016
Ungültiges Avro: "2016"

© 2016-2022 Altova GmbH

- *03: Avro-Schema als JSON-Objekt: Array von Ganzzahlen*

  Dieses Schema ist ein JSON-Objekt, in dem ein Array von Ganzzahlen definiert ist:

```
{
   "type": "array",
   "items": "int"
}
```

  Gültiges Avro: `[2016, 2017]`
  Gültiges Avro: `[2016]`
  Gültiges Avro: `[2016]`
  Ungültiges Avro: `2016, 2017`

- *04: Avro-Schema als JSON-Objekt: Datensätze*

  Dieses Schema ist ein JSON-Objekt, in dem ein einziger Datensatz (record) definiert ist:

```
{
   "type": "record",
   "name": "ages",
   "fields" : [
   {"name": "name", "type": "string"},
   {"name": "age", "type": "int"}
   ]
}
```

  Gültiges Avro: `{"name":"John", "age":35}`

- *05: Avro-Schema als JSON-Objekt: Mehrere Datensätze*

  Dieses Schema ist ein JSON-Objekt, in dem ein Array von Elementen definiert sind, von denen jeder ein JSON-Objekt sein muss:

```
{
   "type": "array",
   "items": {
   "type": "record",
   "name": "ages",
   "fields" : [
   {"name": "name", "type": "string"},
   {"name": "age", "type": "int"}
   ]
    }
}
```

  Gültiges Avro: `[{"name":"Mary", "age":34}, {"name":"John", "age":35}]`

Avro-Schemadateitypen

Wenn Sie die Funktionen von XMLSpy für die Bearbeitung und Validierung von Dateien im Zusammenhang mit Avro verwenden möchten, muss XMLSpy in der Lage sein, eine Datei als Avro-Schema zu erkennen. Eine Datei wird als Avro-Schema erkannt, wenn die Dateierweiterung im XMLSpy-Dialogfeld "Optionen" (**Extras | Optionen | Dateiarten**[1606]) als solche erkannt wird. In den Standardeinstellungen von XMLSpy ist eine einzige Dateierweiterung[779], nämlich die Erweiterung `.avsc`, als die einer Avro-Schema-Datei definiert. Wenn Sie

andere Dateierweiterungen für Avro-Schema-Dokumente erstellen möchten[779], fügen Sie diese Dateierweiterungen als Avro-Schemaerweiterungen zur Liste im Dialogfeld "Optionen"[1606] hinzu.

Erstellen und Bearbeiten von Avro-Schemas

Sie können eine neue Datei[1276] in XMLSpy als Avro-Schema erstellen, indem Sie eine Avro-Schemadatei als ihren Dateityp definieren. XMLSpy stellt während der Eingabe intelligente Bearbeitungshilfen zur Verfügung. Dazu gehören kontextsensitive Schlüsselwortvorschläge, automatische Eingabe von eckigen und geschweiften Klammerpaaren, von Anführungszeichenpaaren, Syntaxfärbung und Autokomplettierung von Schlüsselwörtern. Zusätzlich dazu gibt es drei Eingabehilfen: JSON-Eigenschaften, JSON-Werte und JSON-Entities. Die darin verfügbaren Entities sind kontextsensitiv. Doppelklicken Sie auf einen Eintrag, um diesen an der aktuellen Cursorposition einzufügen. Sie können die Datei mit dem Menübefehl **Validieren | XML validieren (F8)** anhand der Avro-Schema-Spezifikation validieren.

© 2016-2022 Altova GmbH

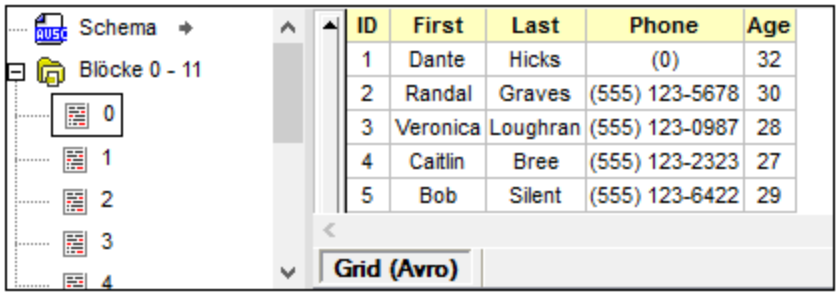
14.2 Avro-Daten im JSON-Format

Avro-Daten können ins Binärformat oder ins JSON-Format serialisiert werden. In den folgenden Punkten wird die Unterstützung von XMLSpy für dieses Avro-Format erläutert.

- Avro-Daten im JSON-Format werden normalerweise in einer `.json` -Datei gespeichert. Sie können festlegen, dass XMLSpy zusätzliche Dateierweiterungen als JSON-konform erkennt.[779]
- Avro-JSON-Dateien können in der Text-[685] und der Grid-Ansicht[690] geöffnet und bearbeitet werden.
- Der Avro-JSON-Datei kann eine Avro-Schema-Datei zugewiesen werden[765] und die Datendatei kann anschließend anhand des Avro-Schemas validiert werden[765].
- Sowohl in der Text-[685] als auch in der Grid-Ansicht[690] stehen intelligente Bearbeitungsfunktionen für JSON-Dokumente zur Verfügung. Wenn einem Avro-Datendokument im JSON-Format außerdem ein Avro-Schema zugewiesen ist, so steht in der Avro-Instanz die Autokomplettierung von schemadefinierten Schlüsselwörtern[685] zur Verfügung.

14.3 Avro-Ansicht: eine Grid-Ansicht von Avro-Binärdateien

Avro-Daten können ins Binärformat oder ins JSON-Format serialisiert werden. Das Binärformat enthält sowohl die Avro-Datenstrukturen als auch ihre Schemas und wird normalerweise mittels automatisierter Datenverarbeitungsprozeduren generiert. Eine Avro-Binärdatei kann in der Avro-Ansicht, einer Grid-Ansicht, in der die Avro-Datenstruktur in einem übersichtlichen Tabellenformat (*siehe Abbildung unten*) angezeigt wird, geöffnet werden. Die Avro-Ansicht dient also als benutzerfreundliches Ansichtsprogramm für Avro-Binärdateien.



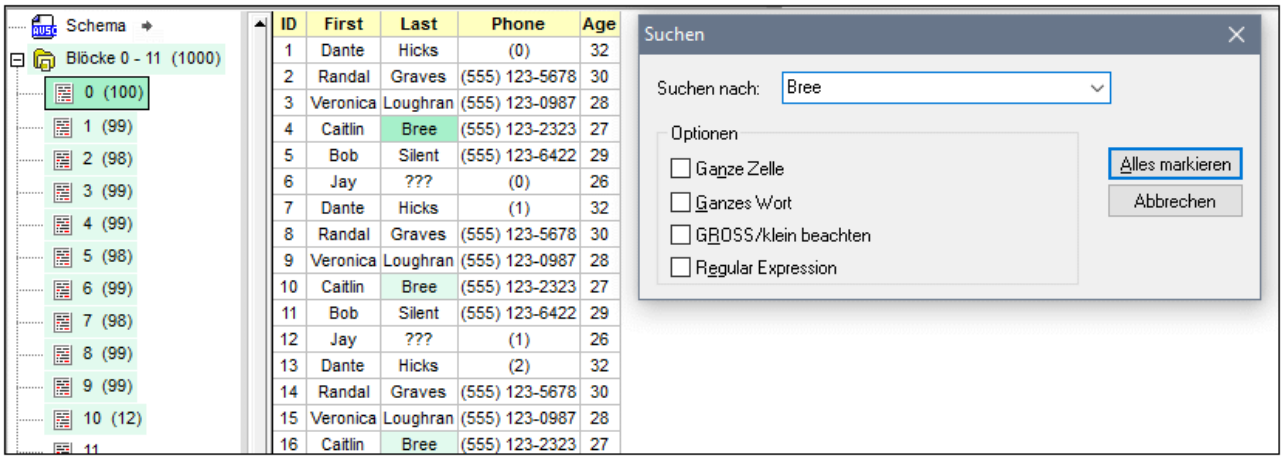
Beachten Sie die folgenden Punkte:

- Die Avro-Binärdatei muss für XMLSpy als solche erkennbar sein. Setzen Sie dazu die Dateierweiterung der Avro-Binärdatei[779] im Abschnitt **Optionen | Dateiarten**[1606] auf Avro-konform. Standardmäßig wurde die Dateierweiterung `.avro` als Avro-konform definiert. Sie können weitere Dateitypen hinzufügen und als Avro-konform definieren[779]. Diese Dateien werden in der Avro-Ansicht geöffnet.
- Die Avro-Ansicht besteht aus zwei Bereichen: (i) einem Bereich *Blöcke* für die Navigation und (ii) einem Bereich *Daten*, in dem die Datenstruktur des im Bereich *Blöcke* ausgewählten Blocks angezeigt wird.
- Im Bereich *Blöcke* sind die Datenblöcke in Gruppen von `1000` gegliedert. Jede Gruppe kann erweitert/reduziert angezeigt werden. Die Datenblöcke werden nach Indexnummer angezeigt.
- Um einen bestimmten Datenblock anzuzeigen, suchen Sie ihn im Bereich *Blöcke* und doppelklicken Sie darauf.
- Der Bereich *Blöcke* enthält auch einen Eintrag namens *Schema*. Wenn Sie auf die Schaltfläche rechts davon klicken, wird das Avro-Schema aus der Avro-Binärdatei extrahiert und auf einem neuen Textansichtsregister geöffnet. Sie können das Avro-Schema anschließend bei Bedarf speichern.

Textsuche

Um nach einem Textstring zu suchen, wählen Sie den Menübefehl **Bearbeiten | Suchen (Strg+F)**. Geben Sie den Suchbegriff im daraufhin angezeigten Dialogfeld (*siehe Abbildung unten*) als Textstring oder Regular Expression ein. Aktivieren Sie alle gewünschten Optionen (Beschreibung hier)[1307]. Klicken Sie auf **Alle markieren**.

© 2016-2022 Altova GmbH



- Die gefundenen Übereinstimmungen werden sowohl im Bereich *Blöcke* als auch im Bereich *Daten* markiert: der aktuell ausgewählte Treffer wird dunkelgrün angezeigt, alle anderen hellgrün.
- Im Bereich *Blöcke* wird die Anzahl der Treffer neben dem Eintrag angezeigt.
- Sie können durch die Treffer navigieren, indem Sie zu einem Block gehen, ein Feld im Block auswählen und anschließend **F3** (**Bearbeiten | Weitersuchen**) und **Umschalt +F3** (Vorheriges suchen) drücken.
- Beachten Sie, dass die Avro-Ansicht schreibgeschützt ist; Sie können Daten in der Avro-Binärdatei nicht bearbeiten.

15 WSDL und SOAP

In diesem Abschnitt werden die WSDL- und SOAP-Funktionen von XMLSpy beschrieben.

Altova Website: WSDL-Editor

WSDL

Ein WSDL-Dokument ist ein XML-Dokument, in dem ein Webservice beschrieben wird. XMLSpy unterstützt WSDL 1.1 und WSDL 2.0. Sie können sowohl WSDL 1.1- als auch WSDL 2.0-Dokumente in der WSDL-Ansicht von XMLSpy erstellen und bearbeiten. Die WSDL-Ansicht enthält die richtigen Bearbeitungshilfen für alle WSDL-Versionen.

In der WSDL-Ansicht von XMLSpy kann ein WSDL-Dokument mittels grafischer Bausteine erstellt werden, wodurch die Erstellung enorm vereinfacht wird. Die WSDL-Ansicht[253] wird im Abschnitt Bearbeitungsansichten[140] näher beschrieben. Eine Anleitung zur Erstellung eines WSDL-Dokuments finden Sie in dieser Dokumentation im WSDL Tutorial[789]. Außerdem können Sie WSDL-Dokumente in der Text-[144] und der Grid-Ansicht[169] anzeigen und bearbeiten. In diesen beiden Ansichten werden WSDL-Dokumente wie ganz normale XML-Dokumente[288] bearbeitet.

XML-Signaturen[417] für WSDL-Dateien in der WSDL-Ansicht können als externe Dateien erstellt werden und auch in die WSDL-Datei verpackt (enveloped) werden. Eine Anleitung zur Arbeit mit Signaturen finden Sie im Abschnitt XML-Signaturen[417].

SOAP

SOAP ist eine Spezifikation zur Übertragung von XML-Nachrichten (Messages) zwischen Applikationen. In XMLSpy können Sie mit Hilfe der intelligenten Bearbeitungsfunktionen für XML-Dokumente[288] nicht nur ein SOAP-Dokument erstellen und bearbeiten (in der Textansicht[144] und der Grid-Ansicht[169]), sondern anhand einer WSDL-Datei auch eine SOAP Request-Datei generieren. Eine Anleitung, wie man anhand einer WSDL-Datei einen SOAP Request generiert, finden Sie im WSDL Tutorial[789]. XMLSpy kann mit Hilfe der Befehle im Menü SOAP[1522] SOAP Requests auch senden und empfangen. Außerdem können Sie SOAP Requests mit dem SOAP Debugger[801] von XMLSpy debuggen. Eine Beschreibung dazu finden Sie in einem Unterabschnitt dieses Abschnitts.

© 2016-2022 Altova GmbH

15.1 WSDL Tutorial

Dieses Tutorial ist in zwei Teile gegliedert:

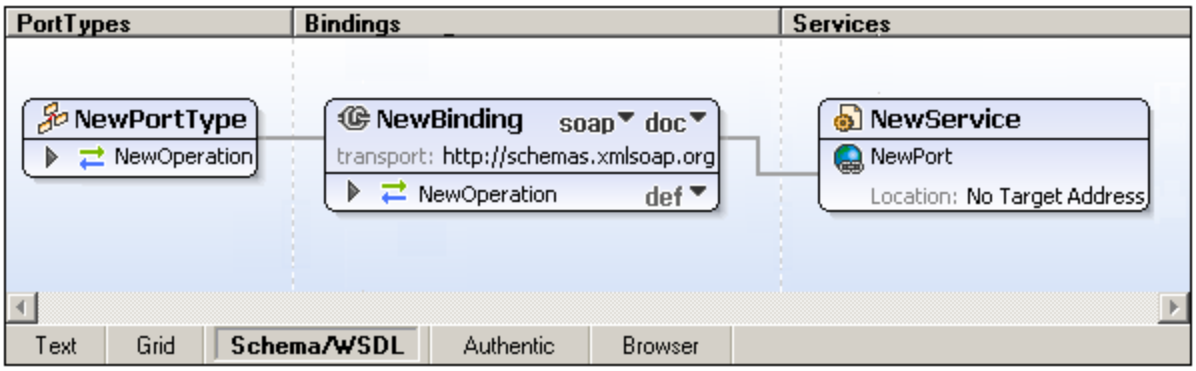
- Der erste Teile enthält eine Beschreibung, wie ein WSDL-Dokument in der grafischen WSDL-Ansicht von XMLSpy erstellt wird. In diesem Teil lernen Sie: (i) wie man mit Hilfe der Menüoption **Datei | Neu** ein rudimentäres WSDL-Dokument erstellt, (ii) wie man einen PortType erstellt (iii) wie man ein Binding erstellt; (iv) wie man einen Service und einen Port erstellt; (v) und das Dokument validiert und speichert.
- Der zweite Teil enthält eine Beschreibung, wie man eine Verbindung zu einem Webservice herstellt, die WSDL-Datei lokal speichert und einen SOAP Request an den Webservice sendet

Sie können das alles in der graphischen WSDL-Ansicht[253] tun und benötigen die Textansicht hierfür nicht. Sie können die WSDL-Komponenten direkt mit Hilfe von Drag & Drop verschieben und Eigenschaftswerte in den Eingabehilfen der WSDL-Ansicht eingeben.

Siehe auch: Weitere Informationen zum Arbeiten mit WSDL-Dokumenten finden Sie in den Abschnitten WSDL-Ansicht[253] und Menübefehle | Menü "WSDL"[1509].

15.1.1 Erstellen eines neuen Dokuments

Um ein neues WSDL-Dokument zu erstellen, wählen Sie den Befehl **Datei | Neu**. Wählen Sie im Dialogfeld "Neues Dokument erstellen", das daraufhin angezeigt wird, `WSDL` (WSDL Web Service Description v1.1) als Dokumenttyp aus und klicken Sie auf **OK**. Daraufhin wird das Gerüst für ein neues Dokument erstellt (*Abbildung unten*), welches in der grafischen WSDL-Ansicht (in diesem Tutorial als WSDL-Ansicht bezeichnet) geöffnet wird.



Zuweisen eines Target Namespace

Wechseln Sie in die Text-Ansicht. Der Start Tag des Elements `wsdl:definitions` sieht ungefähr folgendermaßen aus:

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
3       xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
4       xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
5       xmlns:xs="http://www.w3.org/2001/XMLSchema"
6       xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
7       xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
8       xmlns:tns="http://new.webservice.namespace"
9       targetNamespace="http://new.webservice.namespace">
10
```

Ändern Sie das Target Namespace (`targetNamespace`) Attribut in `http://mywebservice.namespace` oder einen beliebigen anderen Namen (Dieses Tutorial dient schließlich nur dazu, Ihnen zu zeigen, wie ein WSDL-Dokument erstellt wird und stellt keinen tatsächlichen Service bereit). Anschließend sollten Sie auch den Namespace-Wert des `tns` Attributs in `http::://mywebservice.namespace` ändern (oder den als Target Namespace gewählten Namespace).

Anmerkung: In den WSDL-Elementen, die im gerade erstellten Dokumentgerüst erstellt wurden, befinden sich WSDL-Elemente im Target Namespace, während Referenzen zu WSDL-Elementen über das Präfix `tns` definiert werden. Beispiel: `<wsdl:binding name="NewBinding" type="tns:NewPortType">`. Damit das `tns` Präfix mit dem Target Namespace übereinstimmt, muss sein Namespace-Wert mit dem Target Namespace übereinstimmen.

15.1.2 Erstellen eines PortType

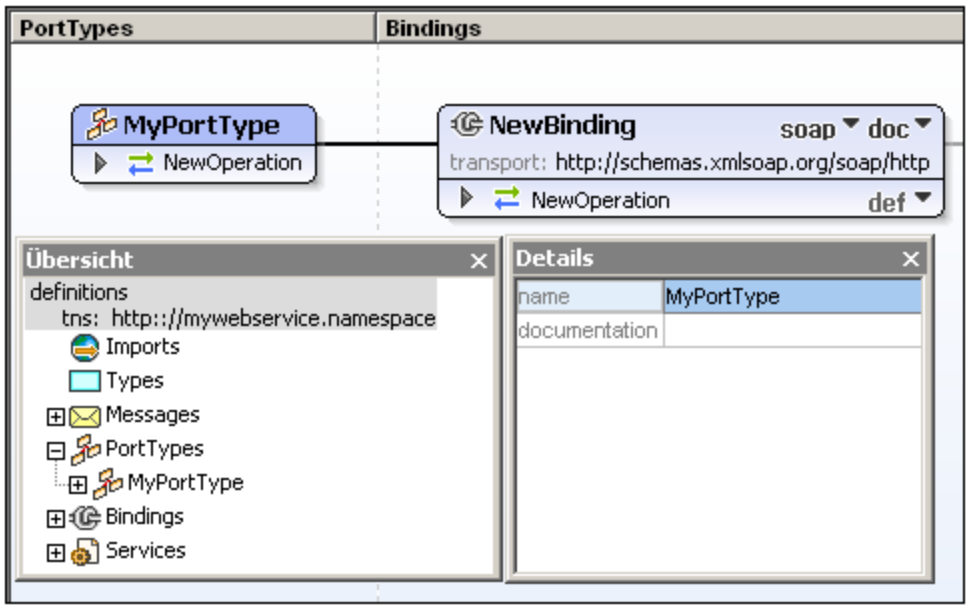
Um einen PortType zu erstellen, sind folgende Schritte erforderlich:

- Benennung des PortType
- Einfügen einer Operation
- Hinzufügen von Input- und Output Messages
- Hinzufügen von Parametern zu Messages

Benennung des PortType

Benennen Sie den PortType `NewPortType` in `MyPortType` um, indem Sie im Design in die Titelleiste des Felds `NewPortType` doppelklicken, den Namen bearbeiten und anschließend die **Eingabetaste** drücken. Beachten Sie, dass sich der Name der PortType auch im Übersichtsfenster und in der Eingabehilfe "Details" ändert (*Abbildung unten*).

© 2016-2022 Altova GmbH



Einfügen einer Operation

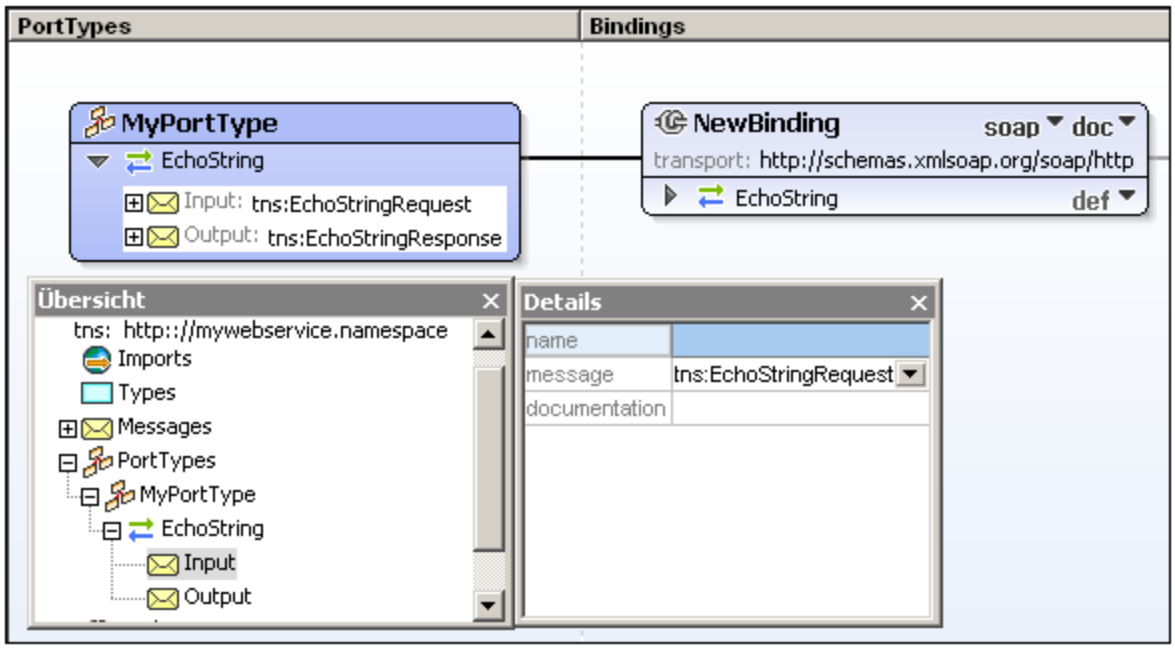
Im Fall von `MyPortType` ist eine Operation, `NewOperation`, bereits vorhanden. Daher werden wir mit dieser Operation arbeiten. Zunächst benennen wir `NewOperation` in z.B. `EchoString` um (doppelklicken Sie auf den Namen, bearbeiten Sie ihn und drücken Sie die **Eingabetaste**). (Um zusätzliche Operationen für einen PortType einzufügen, klicken Sie mit der rechten Maustaste auf den Kasten PortType und wählen Sie den Befehl **Operation anhängen** und klicken Sie anschließend auf die gewünschte Art von Operation)

Hinzufügen von Input- und Output Messages

Wenn eine Operation an einen PortType angehängt wird, können Sie auswählen, welchem von fünf Typen die Operation angehören soll.

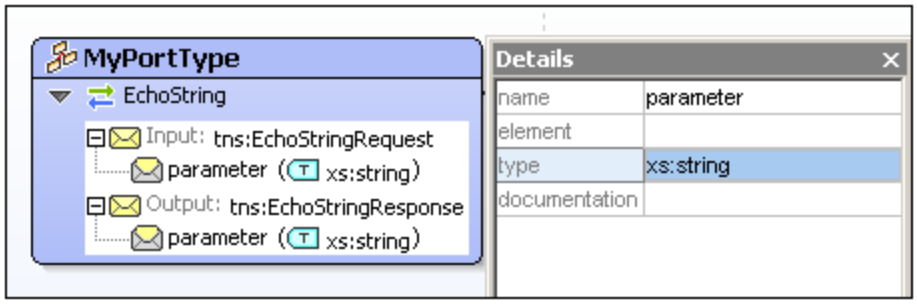
- Request response
- Solicit response
- One-way
- Notification
- Leere Operation

Für jeden Typ werden je nach Operationstyp automatisch Input- und Output Messages hinzugefügt. Bei Auswahl des Befehls **Leere Operation** können Sie mit einem Rechtsklick auf die Operation den einzufügenden Message-Typ auswählen. Eine Message kann durch Rechtsklick und Auswahl des Befehls **Input/Output/Fault-Element löschen** gelöscht werden. Im Fall der Operation `EchoString` benennen Sie die Input und Output Message in `EchoStringRequest` bzw. `EchoStringResponse` um.



Hinzufügen von Parametern zu Messages

Jede Input oder Output Message wird mit einem einzigen Default Message Part (oder Parameter) vom Typ `xs:string` erstellt (*siehe Abbildung unten*). Um einen weiteren Parameter hinzuzufügen, klicken Sie mit der rechten Maustaste entweder auf die Message oder auf einen ihrer Parameter und wählen Sie den Befehl **Message Part (Parameter) hinzufügen**.



Um einen Parameter zu bearbeiten, (i) doppelklicken Sie entweder auf den Text und bearbeiten Sie ihn; (ii) oder rechtsklicken Sie auf den Parameter und wählen Sie "Bearbeiten" oder (iii) verwenden Sie dien Eingabehilfe "Details" (*siehe Abbildung unten*).

15.1.3 Erstellen eines Binding

Ein Binding ist ein konkretes Protokoll und eine Datenformatspezifikation für einen bestimmten PortType. Bei Erstellung eines Binding sind daher die folgenden Schritte erforderlich:

© 2016-2022 Altova GmbH

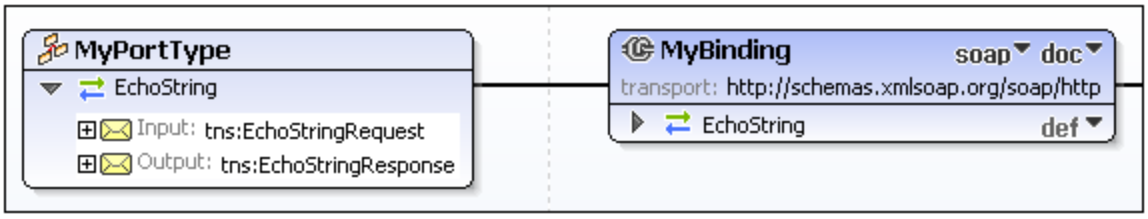
- Verknüpfung eines Binding mit einem PortType.
- Definition des Binding-Protokolls und der Datenformatsspezifikation.
- Verknüpfung des Binding mit einem Port.

Verknüpfung des Binding mit einem PortType

Bei Erstellung eines neuen Binding (rechtsklicken Sie dazu an eine beliebige Stelle in einem leeren Design-Bereich und wählen Sie den Befehl **Binding einfügen**) ist kein PortType mit diesem Binding verknüpft (*Abbildung unten*). (Wenn Sie ein neues WSDL-Dokument erstellt haben, wird das Binding standardmäßig mit dem standardmäßig erstellten PortType verknüpft. Die Verknüpfung wird als Verbindungslinie zwischen zwei Kästchen dargestellt).

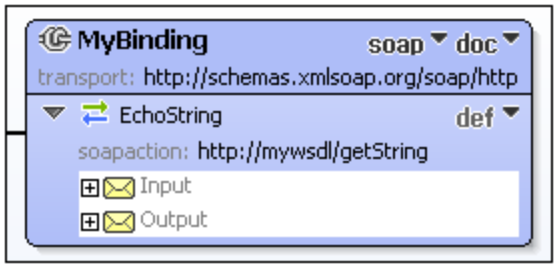


Um einen PortType mit einem Binding zu verknüpfen, klicken Sie im neuen Binding-Kästchen auf den Nach-unten-Pfeil des PortType-Eintrags (*siehe Abbildung oben*). Daraufhin wird eine Liste der in dem Dokument definierten PortTypes angezeigt. Wählen Sie den PortType, mit dem das Binding verknüpft werden soll. Wenn ein PortType mit dem Binding verknüpft wurde, wird die Verknüpfung in Form einer Linie dargestellt, die das Kästchen des ausgewählten PortType mit dem Binding-Kästchen verbindet:



Auswahl des Protokolls und Datenformats

Das Protokoll des Binding wird durch Klicken auf den Nach-unten-Pfeil in der Titelleiste des Binding-Felds (dem mit dem Eintrag `soap/http`) und Auswahl eines der vier verfügbaren Protokolle ausgewählt: `SOAP`, `SOAP 1.2`, `HTTP-GET` und `HTTP-POST` (*Abbildung unten*). Wenn Sie das SOAP 1.1- oder 2.1-Protokoll auswählen, können Sie als Datenformat `document` oder `rpc` auswählen (Diese Listenoptionen erscheinen bei Klick auf den Drowdown-Pfeil rechts von der Protokollauswahl).



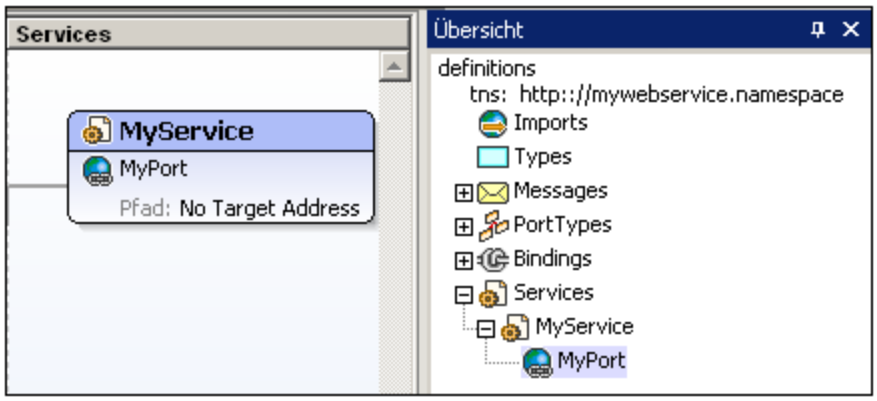
Die `soapAction` für die einzelnen Operationen im Binding können im Design (*siehe Abbildung oben*) oder in der Eingabehilfe "Details" definiert werden, wenn diese Operation ausgewählt ist.

Verknüpfung des Binding mit einem Port

Um das Binding mit einem Port zu verknüpfen, muss zuerst der Port definiert werden. Eine Beschreibung zur Erstellung eines Port in einem Service und Verknüpfen des Port mit einem Binding finden Sie im Abschnitt Erstellen eines Service und von Ports 794.

15.1.4 Erstellen eines Service und von Ports

Um einen neuen Service hinzuzufügen, rechtsklicken Sie auf eine leere Stelle im Design und wählen Sie im Kontextmenü den Befehl "Service einfügen". Wenn Sie ein neues WSDL-Dokument angelegt haben, enthält das Design bereits einen Service. Sie können den Service umbenennen, indem Sie auf seinen Namen klicken, ihn bearbeiten und anschließend die **Eingabetaste** drücken. Beachten Sie, dass der Name des Service sich auch in der Eingabehilfe "Übersicht" ändert (*Abbildung unten*).



Doppelklicken Sie in der Eingabehilfe "Übersicht" auf den Eintrag `NewPort`, ändern Sie ihn in `MyPort`. und drücken Sie die **Eingabetaste**. Beachten Sie, dass sich der Name des Port auch im Design im Feld `MyService` ändert (*Abbildung oben*). Um zusätzliche Ports hinzuzufügen, rechtsklicken Sie entweder auf den Service oder den Port und wählen Sie im Kontextmenü den Befehl **Port einfügen**.

Eingabe der Adresse eines Port

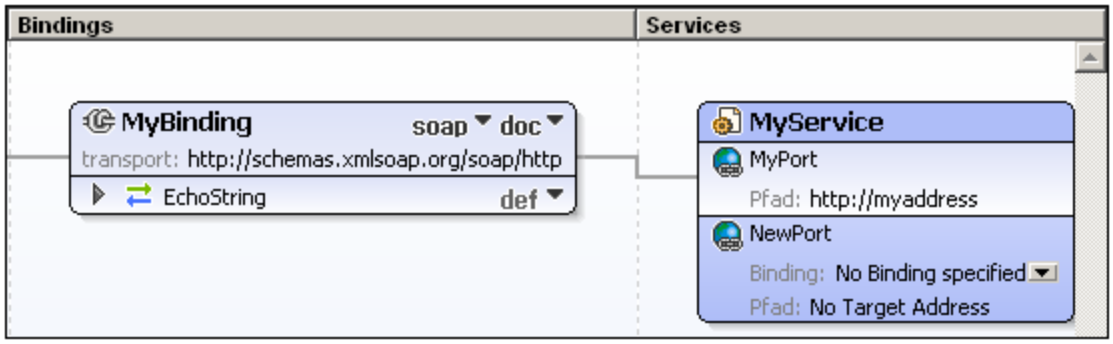
Die Adresse eines Port kann folgendermaßen eingegeben werden: (i) entweder direkt im Design als Wert des Eintrags `Pfad` (*siehe Abbildung oben*) oder (ii) in der Eingabehilfe "Detail" (durch Doppelklick auf das Feld `Location` und Eingabe der Adresse (*Abbildung unten*).



 © 2016-2022 Altova GmbH

Verknüpfen eines Binding mit einem Port

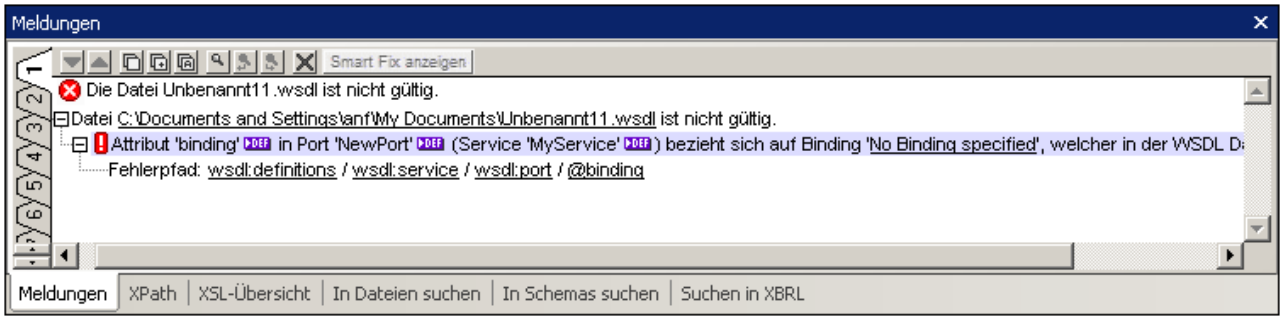
Ein Port ist der Endpunkt, der ein Binding mit einer Netzwerk-Adresse kombiniert. Sobald die Adresse der Port definiert wurde, müssen Sie nur noch ein Binding mit dem Port verknüpfen. Klicken Sie dazu auf den Nach-unten-Pfeil des Binding-Eintrags eines Port und wählen Sie eines aus der Liste der für dieses Dokument definierten Bindings aus.



Anmerkung: Wenn ein Binding bereits mit einem Port verknüpft wurde und Sie ein anderes Binding verknüpfen möchten, müssen Sie die Referenz des Binding entfernen (über das Kontextmenü des Port) und anschließend die neue Binding-Referenz einfügen.

15.1.5 Validieren des WSDL-Dokuments

Nach Fertigstellung des WSDL-Dokuments können Sie es durch Auswahl des Befehls **XML | XML validieren (F8)** validieren. Die Ergebnisse der Validierung werden unterhalb des Hauptfensters im Fenster "Meldungen" angezeigt (*Abbildung unten*).



Es werden detaillierte Informationen über jeden gefundenen Fehler angezeigt, sodass Sie diesen schnell finden und beheben können.

15.1.6 Herstellen einer Verbindung zu einem Webservice und Öffnen von Dateien

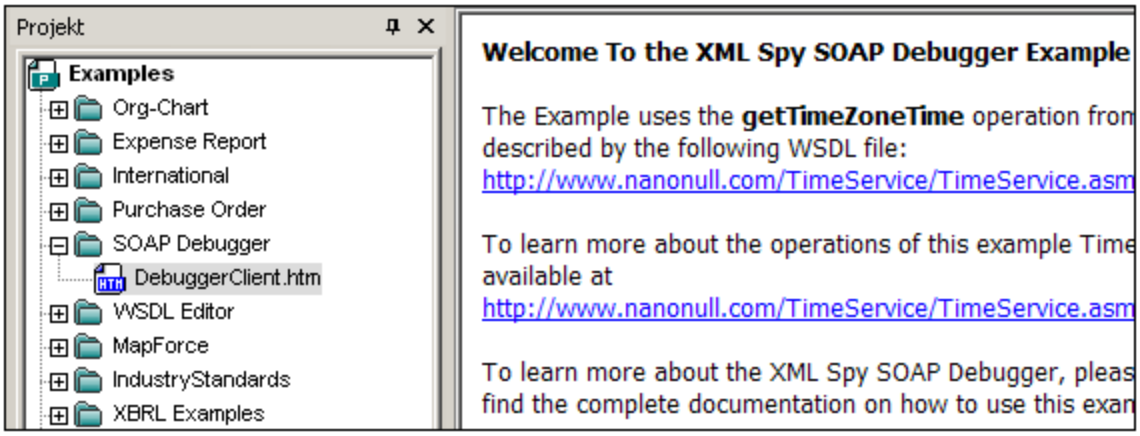
In diesem Abschnitt wird Folgendes beschrieben:

- Zugriff auf einen Webservice mittels XMLSpy
- Öffnen einer WSDL-Datei im Web in XMLSpy
- Öffnen eines mit dem WSDL-Dokument verknüpften XML-Schemas in XMLSpy

Zugriff auf einen Webservice

Ein Webservice wird normalerweise über eine HTML-Seite aufgerufen. Ein Beispiel dafür ist `DebuggerClient.htm`, welches Sie im Ordner Examples sowie im XMLSpy-Projekt[1080] unter `Examples/SoapDebugger` (Öffnen Sie `Examples.spp`, um mit dem Projekt zu arbeiten) finden. Um den auf dieser Seiten angezeigten Webservice aufzurufen, gehen Sie folgendermaßen vor:

1. Aktivieren Sie das Projektfenster, falls es nicht zu sehen ist (über die Menüoption **Fenster | Projekt-Fenster**).
2. Klicken Sie auf das Erweiterungssymbol neben dem Ordner "SOAP Debugger" und doppelklicken Sie auf die Datei `DebuggerClient.htm`. Daraufhin wird im Hauptfenster der SOAP Debugger Beispiel Client geöffnet.

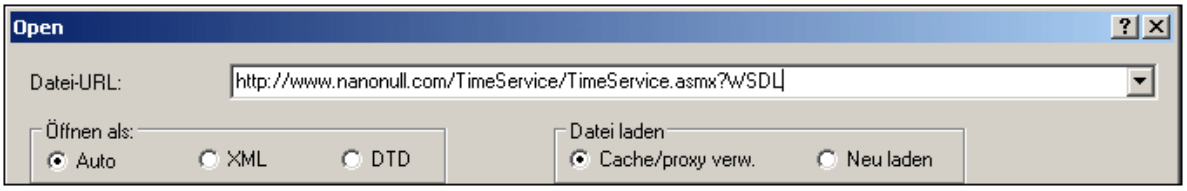


Öffnen einer WSDL-Datei in XMLSpy

Um eine webbasierte WSDL-Datei in XMLSpy zu öffnen, gehen Sie folgendermaßen vor:

1. Wählen Sie die Menüoption **Datei | Öffnen** und klicken Sie in im Dialogfeld "Öffnen" auf die Schaltfläche **Zu URL wechseln.** Geben Sie dann die Adresse `http://www.nanonull.com/TimeService/TimeService.asmx?WSDL` in das Feld "Datei-URL" ein oder kopieren Sie sie hinein.

© 2016-2022 Altova GmbH



2. Klicken Sie auf **Öffnen,** um die WSDL-Datei zu laden. Die WSDL-Datei wird in der Text-Ansicht angezeigt.

3. Wählen Sie die Menüoption **Datei | Speichern unter** und speichern Sie die Datei lokal z.B. unter dem Namen `timeservice.wsdl`.
4. Klicken Sie auf das WSDL-Ansichtsregister, um die Datei im grafischen WSDL-Editor anzuzeigen.

Anzeigen der mit der aktiven WSDL-Datei verknüpften Schema-Datei

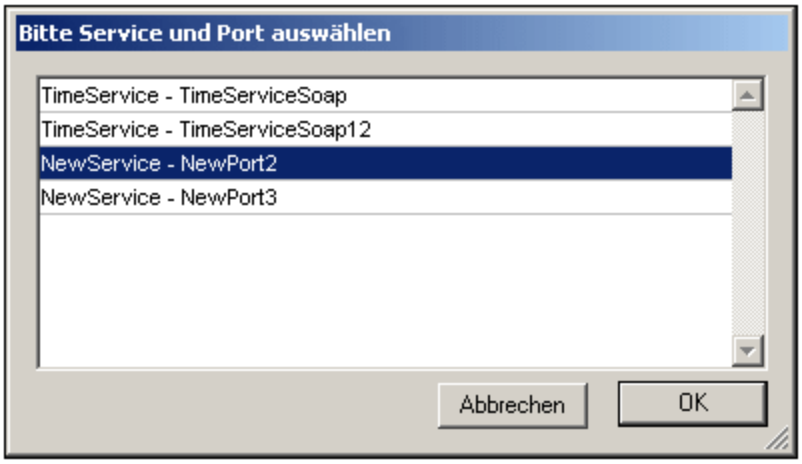
Während im Hauptfenster die Datei `timeservice.wsdl` das aktive Dokument ist, wählen Sie die Menüoption **WSDL | Types | Schema(s) in Schema-Ansicht bearbeiten**. Daraufhin wird die Schema-Datei geöffnet, in der alle in der Datei `timeservice.wsdl` verwendeten Datentypen definiert sind. Sie können das Schema ändern und die Änderungen speichern. Diese Änderungen werden erst dann in der WSDL-Datei übernommen, sobald die Datei erneut geparst wird.

Anmerkung: Es wird empfohlen, die WSDL-Datei über ihren Dateinamen (z.B: **`timeservice.wsdl`**) anstatt mit Hilfe der **`?wsdl`**-Abfragemethode aufzurufen, da als Ergebnis der Abfragemethdoe eventuell eine WSDL-Datei zurückgegeben wird, der einige Funktionalitäten der WSDL-Originaldatei fehlen oder die nicht korrekt funktioniert.

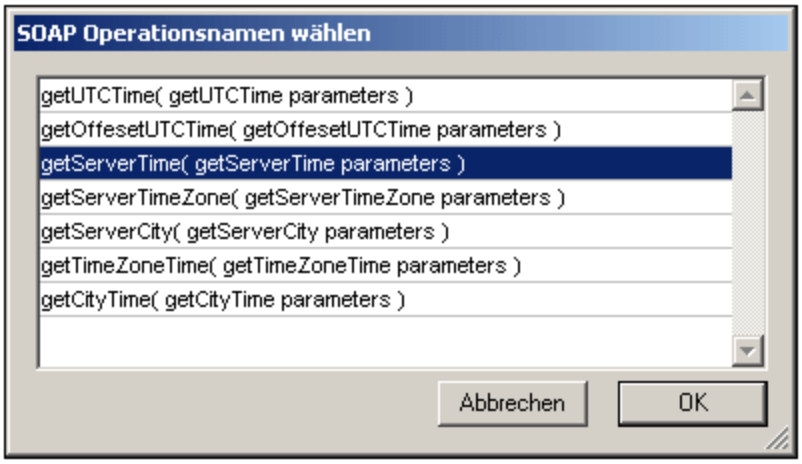
15.1.7 Senden eines SOAP Request von der WSDL-Datei

Um einen SOAP Request von der Datei `timeservice.wsdl` zu senden, gehen Sie folgendermaßen vor:

1. Machen Sie `timeservice.wsdl` im Hauptfenster zur aktiven Datei.
2. Wählen Sie die Menüoption **SOAP | Neuen SOAP Request erzeugen**.
3. Suchen Sie die Datei `timeservice.wsdl` und bestätigen Sie die Auswahl mit **OK**.
4. Wenn es unter den verschiedenen im Dokument definierten Services mehr als einen Port gibt, der ein SOAP 1.1 oder 1.2-Binding referenziert, erscheint ein Popup-Fenster (*Abbildung unten*), in dem Sie den gewünschten Service und Port auswählen müssen. Klicken Sie anschließend auf OK.



5. Wählen Sie im Dialogfeld, das daraufhin angezeigt wird (*Abbildung unten*), eine SOAP-Operation, z.B. `getServerTime` und klicken Sie auf **OK**.



Daraufhin wird ein neues SOAP Request-Dokument erstellt, das die Operation `getServerTime` enthält und das bei Bedarf gespeichert werden kann.

5. Machen Sie das request-Dokument zum aktiven Dokument und wählen Sie die Menüoption **SOAP | Request an Server senden**. Das SOAP Response-Dokument wird im Hauptfenster angezeigt. Es enthält das Element `getServerTimeResult`, welches die aktuelle Server-Uhrzeit des Nanonull.com Time Service anzeigt.

15.1.8 Erstellen von WSDL-Dokumentation

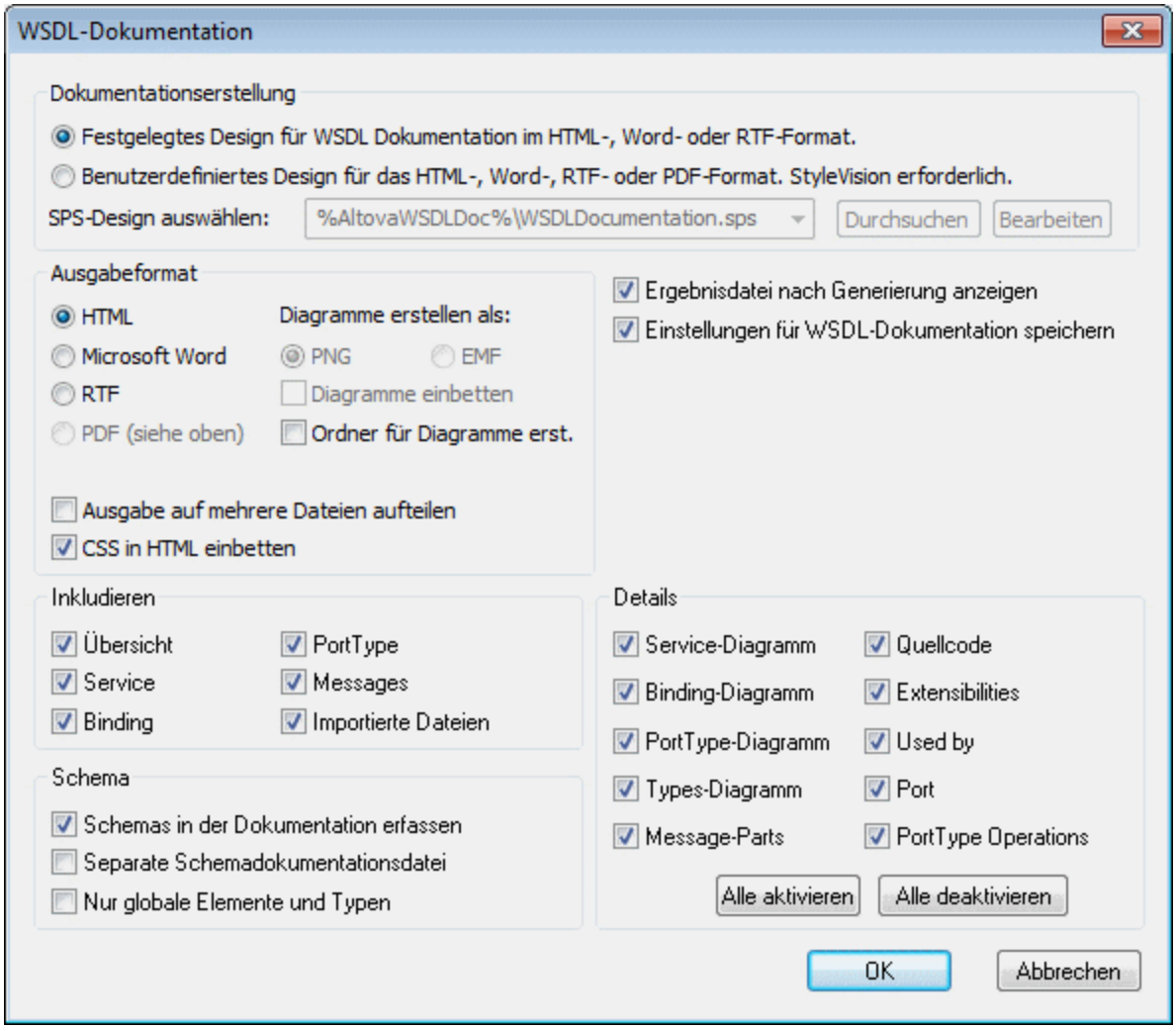
Mit Hilfe der Option **WSDL | Dokumentation generieren** können Sie eine detaillierte Dokumentation des aktuellen WSDL-Dokuments generieren. Sie können die Dokumentation als MS Word-, RTF- oder HTML-Datei ausgeben und angeben, welche Komponenten inkludiert werden sollen. Miteinander in Zusammenhang stehende WSDL-Elemente werden in der generierten Dokumentation durch Hyperlinks verbunden, um die Navigation zu vereinfachen.

Anmerkung: Um Dokumentation im MS Word-Format generieren zu können, muss MS Word (Version 2000 oder höher) installiert sein.

© 2016-2022 Altova GmbH

Um Dokumentation für die WSDL-Datei zu generieren, gehen Sie folgendermaßen vor:

1. Machen Sie `timeservice.wsdl` zum aktiven Dokument.
2. Wechseln Sie in die **WSDL-**Ansicht.
3. Wählen Sie die Menüoption **WSDL | Dokumentation generieren**.
   Daraufhin wird das Dialogfeld "WSDL-Dokumentation" (*siehe Abbildung unten*)geöffnet.
4. Wählen Sie den gewünschten Ausgabetyp: **HTML, MS Word** oder **RTF** und
5. Wählen Sie die spezifischen WSDL-Komponenten, die Sie in der Dokumentation berücksichtigen möchten und legen Sie weitere Optionen fest (siehe WSDL-Dokumentationsoptionen unten).



6. Klicken Sie auf OK und geben Sie den Namen der WSDL-Dokumentationsdatei im Dialogfeld "Speichern unter" ein.

WSDL-Dokumentationsoptionen

Sie können zwischen den folgenden Dokumentationsoptionen wählen:

- Das gewünschte Format wird im Bereich "Output-Format" definiert: entweder HTML, MS Word oder RTF. Die Dokumentation kann entweder als eine einzige Datei generiert oder in mehrere Dateien aufgeteilt werden. Wenn mehrere Dateien generiert werden, entspricht jede Datei einer Komponente. Welche Komponenten in der Ausgabe enthalten sein sollen, wird mittels der Kotrollkästchen im Bereich "Inkludieren" festgelegt.
- Die Option "Diagramme einbetten" ist für die Ausgabeoptionen MS Word und RTF aktiv. Wenn diese Option aktiviert ist, werden Diagramme entweder im PNG- oder im EMF-Format in die Ergebnisdatei eingebettet. Andernfalls werden die Diagramme als PNG- oder EMF-Dateien generiert und in der Ergebnisdatei über Objektlinks angezeigt. Wenn es sich bei der Ausgabedatei um eine HTML-Datei handelt, werden alle Diagramme als dokumentexterne PNG-Dateien generiert.
- Im Bereich **Inkludieren** können Sie auswählen, welche Datenelemente in der Dokumentation inkludiert werden sollen. Bei Auswahl der Option "Index" werden alle verwandten Schemas mit ihren globalen Komponenten nach Komponententyp geordnet am Anfang der Datei aufgelistet. Wenn die Option *Importierte Dateien* aktiviert ist, werden Komponenten in importierten Dateien in die Schema-Dokumentation inkludiert.
- Im Bereich "Schema" können Sie auswählen, ob Schemas in der Datei in die Dokumentation inkludiert werden sollen oder nicht. Soll die Dokumentation auch Schemas erfassen, haben Sie die Wahl zwischen folgenden Optionen: (i) ob die Schema-Dokumentation in einer eigenen Datei oder in der Hauptdokumentation ausgegeben werden soll und (ii) ob das gesamte Schema oder nur globale Elemente, SimpleTypes und ComplexTypes dokumentiert werden sollen.
- Im Bereich **Details** werden die Einzelheiten aufgelistet, die für die einzelnen Komponenten inkludiert werden können. Wählen Sie die gewünschten Informationen aus.
- Die Option "Ergebnisdatei nach der Generierung anzeigen" ist für alle drei Ausgabeoptionen aktiv. Wenn diese Option aktiviert ist, wird die erzeugte Datei in der Browser-Ansicht (HTML-Ausgabe), MS Word (MS Word Ausgabe) und der Standardapplikation für .rtf-Dateien (RTF-Ausgabe) angezeigt.

15.1.9 Konvertieren in WSDL 2.0

Sie können ein bestehendes WSDL 1.1-Dokument in XMLSpy jederzeit in das WSDL 2.0-Format konvertieren. Versuchen Sie dies mit der Beispieldatei `TimeService.wsdl`:

1. Machen Sie die Datei `TimeService.wsdl` zur aktiven Datei. (Diese Datei befindet sich im `Examples`-Projekt von XMLSpy im Ordner `WSDL Editor`.)
2. Klicken Sie auf den Befehl **WSDL | In WSDL 2.0 konvertieren**.
3. Geben Sie nun im Dialogfeld "Speichern unter", das jetzt angezeigt wird, den Namen für die WSDL 2.0 Datei ein - z.B. `TimeService20.wsdl`..
4. Die neue Datei wird generiert, automatisch validiert und in der WSDL-Ansicht angezeigt.

© 2016-2022 Altova GmbH

15.2 SOAP

In diesem Abschnitt wird beschrieben wie man:

- SOAP-Nachrichten anhand von WSDL validiert. SOAP-Nachrichten können nicht nur anhand der SOAP-Spezifikation, sondern auch anhand aller in der entsprechenden WSDL-Definition referenzierten XML-Schemas validiert werden.
- SOAP Requests mit Hilfe des SOAP Debuggers sendet und empfängt
- Breakpoints zum Senden und Empfangen von SOAP Requests setzt
- einen fehlerhaften SOAP Request vor dem Absenden an einen Webservice bearbeitet

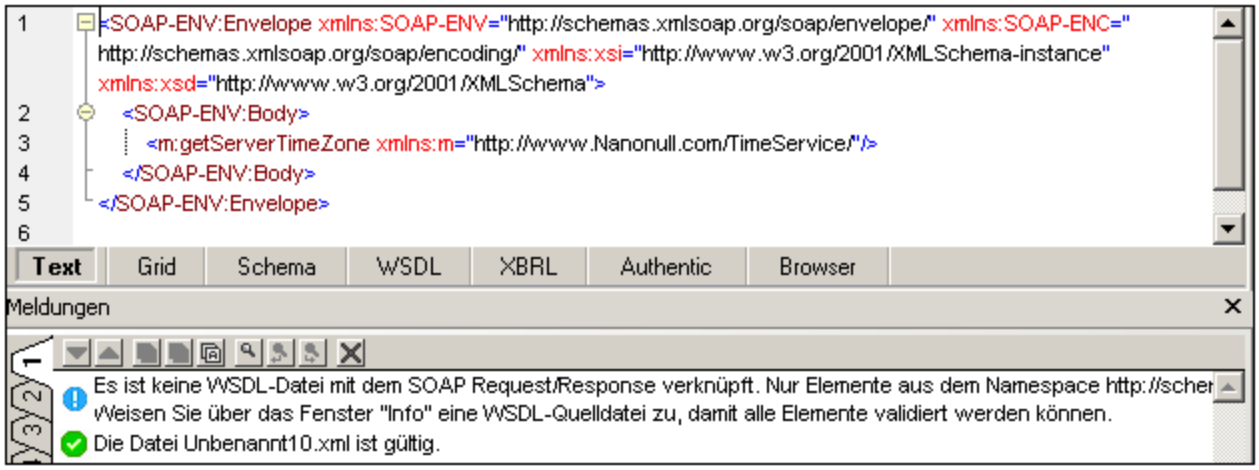
Altova Website: SOAP Debugger

15.2.1 SOAP-Validierung

SOAP-Nachrichten können nicht nur anhand der SOAP-Spezifikation, sondern auch anhand jeden in der entsprechenden WSDL-Definition referenzierten XML-Schemas validiert werden.

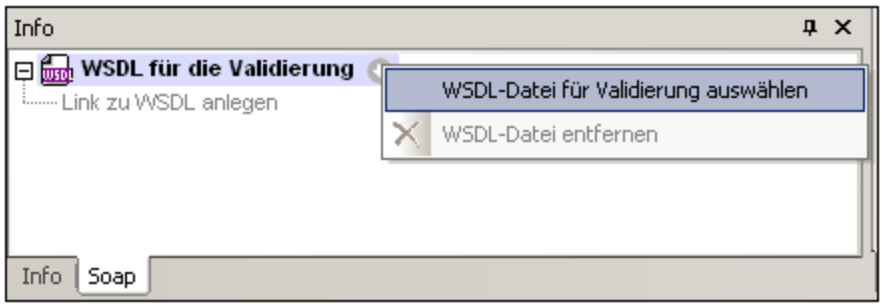
Validierung nur anhand der SOAP-Regeln

Um eine SOAP-Nachricht zu validieren, öffnen Sie die SOAP-Nachrichtendatei (*Abbildung unten*) und drücken Sie **F8** (oder rufen Sie den Menübefehl **XML | Validieren** auf). Da keine WSDL-Datei mit der SOAP-Nachrichtendatei verknüpft wurde, wird die SOAP-Nachricht anhand der Regeln für SOAP-Nachrichten validiert. Die Datei gilt als gültig, wenn sie diesen Regeln entspricht (*siehe Fenster "Meldungen" in der Abbildung unten*).

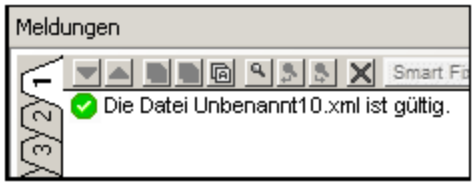


Validierung anhand der SOAP-Regeln und der verknüpften WSDL

Um eine SOAP-Nachricht zusätzlich auch anhand der verknüpften WSDL zu validieren, muss die WSDL-Datei mit der SOAP-Datei verknüpft werden. Dies geschieht im Fenster "Info" auf dem Register "SOAP" (*Abbildung unten*). Klicken Sie auf die Schaltfläche rechts vom Eintrag *WSDL für Validierung* und wählen Sie den Befehl **WSDL-Datei für Validierung auswählen**. Navigieren Sie im Dialogfeld, das daraufhin angezeigt wird, zur gewünschten WSDL-Datei und klicken Sie auf **OK**. Die WSDL-Datei erscheint im Fenster "Info" und die SOAP-Nachricht wird damit verknüpft.



Wenn Sie die Taste **F8** drücken (oder den Menübefehl **XML | Validieren** auswählen), wird die SOAP-Nachricht nicht nur anhand der Regeln für SOAP-Nachrichten, sondern auch anhand der damit verknüpften WSDL-Datei validiert.



Die Datei gilt als gültig, wenn sie beiden Regelwerken entspricht (*siehe Abbildung oben*).

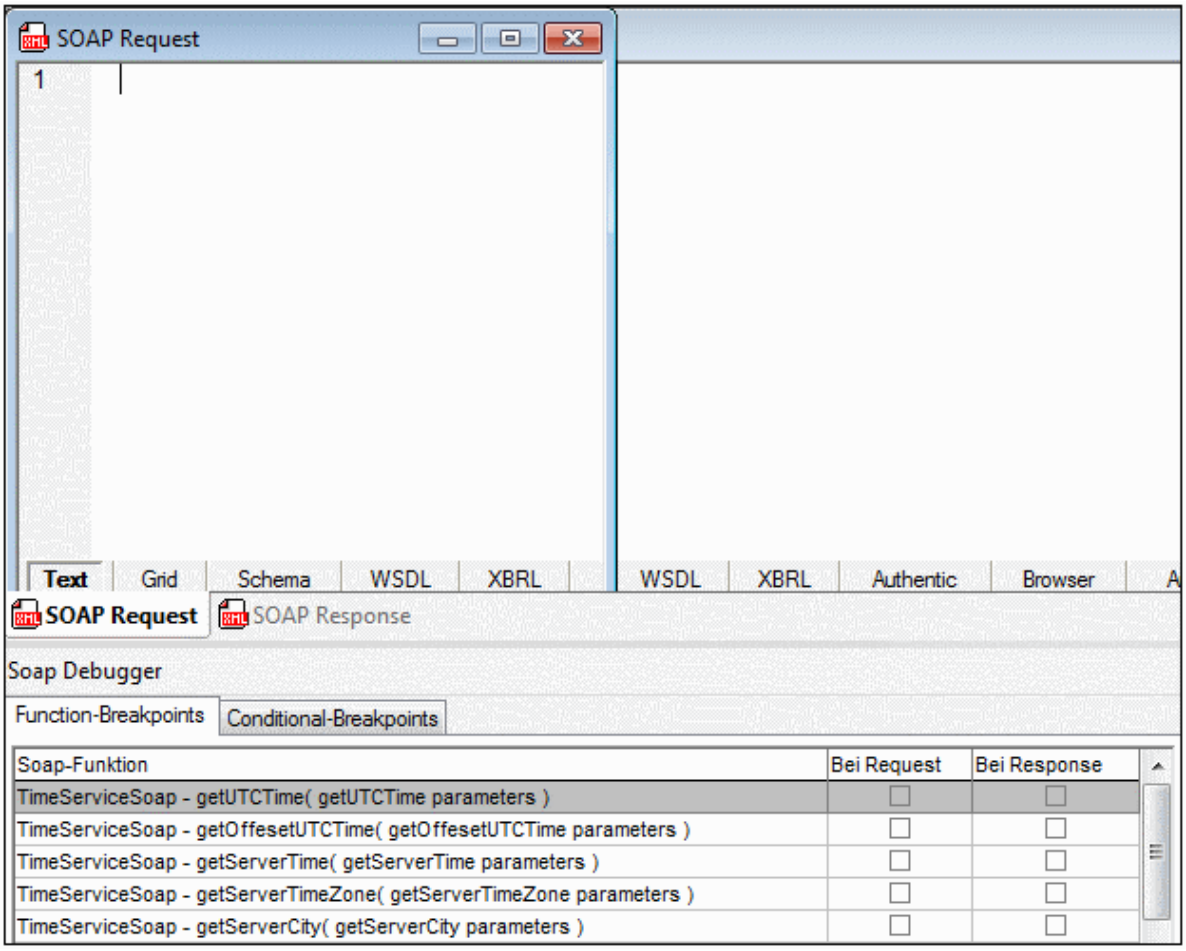
Anmerkung: Das Register "SOAP" wird im Info-Fenster angezeigt, wenn der SOAP Request mit Hilfe der XMLSpy-Funktion zur Erstellung von SOAP Requests anhand einer WSDL-Datei erstellt wurde (**SOAP | Neue SOAP Request erstellen**). Wenn das Register "SOAP" im Info-Fenster nicht sichtbar ist (weil der SOAP Request nicht mit XMLSpy erstellt wurde), so wird das Register sichtbar, sobald Sie die SOAP-Request-Datei gespeichert haben.

15.2.2 SOAP Debugger

Der SOAP Debugger (*Abbildung unten*) dient zum Anzeigen und Analysieren von SOAP Requests und SOAP Responses. Er fungiert als **Proxy Server** zwischen Ihrem Client und dem Webservice.
Der SOAP Debugger unterstützt folgende Funktionen:

- das schrittweise Durchgehen von SOAP Requests und Responses
- die Änderungen von SOAP Requests und Responses
- das Weitersenden geänderter Requests an den Client oder Server
- Für jede Request und Response Message können Breakpoints gesetzt werden. Dies beinhaltet auch bedingte Breakpoints über XPath-Ausdrücke

© 2016-2022 Altova GmbH



Der SOAP Debugger funktioniert folgendermaßen:

- Vor dem Start einer SOAP-Debugger-Sitzung sollten die SOAP Debugger Optionen[804] definiert werden. Dazu gehören die IP-Adresse des Computers und die Layout- und Timeout-Optionen für den SOAP Debugger.
- Um den SOAP Debugger zu öffnen (eine Sitzung zu starten)[805], aktivieren Sie den Ein/Aus-Befehl **SOAP | SOAP Debugger-Sitzung**. Sie müssen nun (i) angeben, wo sich die WSDL-Datei aus der die relevanten SOAP-Informationen stammen, befindet und (ii) welche Quell- und Ziel-Ports verwendet werden sollen.
- Setzen Sie im Fenster "SOAP Debugger Breakpoints" die erforderlichen Breakpoints[810].
- Nun können Sie die Datei, die den SOAP Request ausführt, öffnen und den SOAP Debugger ausführen[811].
- Anschließend können Sie die Ergebnisse analysieren[812] und etwaige Fehler beheben.
- Um den SOAP Debugger zu schließen, klicken Sie den Ein/Aus-Befehl **SOAP | SOAP Debugger-Sitzung**.

In den Unterabschnitten dieses Abschnitts wird beschrieben, wie man den SOAP Debugger verwendet. Als Beispieldatei dient die Datei `DebuggerClient.htm` aus dem Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples`. Bei dieser Beispieldatei fungiert das Browser-Fenster als Client-Applikation, die SOAP-Nachrichten sendet und empfängt. Der

Webservice-Server ist der Nanonull Time Service-Dienst unter der Adresse:
`http://www.nanonull.com/TimeService/TimeService.asmx?WSDL`.

15.2.2.1 SOAP-Kommunikationsvorgang

Nach dem Starten des Proxy Servers (SOAP Debuggers) läuft der SOAP-Kommunikationsvorgang folgendermaßen ab:

Der Proxy Server überwacht einen Eingang/Port ständig auf eingehende Client Requests

- Die Client-Anwendung sendet einen Request an den Proxy Server
- Client Requests können geändert werden, wenn/falls Breakpoints gefunden wurden.
- Die Request-Daten des Proxy Servers werden an den Webservice-Server weitergeleitet.

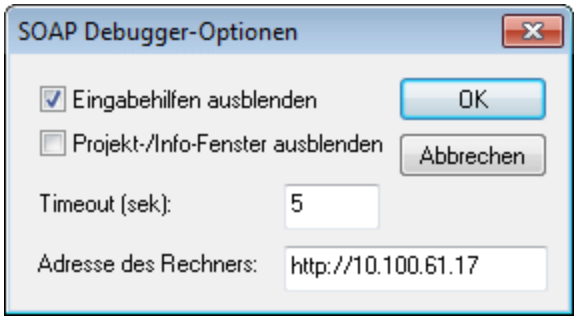
Der Websevice Server antwortet auf den Proxy Server Request und sendet die Response-Daten zurück an den Proxy Server

- Die Server Response-Daten können geändert werden, wenn/falls Breakpoints gefunden werden.
- Die Proxy Server Response-Daten werden an die Client-Applikation weitergeleitet.
- Die Client Applikation empfängt Response-Daten vom Proxy Server.

Port-Einstellungen
Der SOAP Debugger verwendet zum Überwachen der Client Request den Port 8080. Der Port kann nur beim Start einer neuen SOAP Debugger-Sitzung geändert werden. Wenn dieser Port durch Firewalls oder Werbefilter ist, müssen Sie diese Programme zuerst deaktivieren oder eine andere Port-Adresse auswählen.

15.2.2.2 Debugger-Optionen

Über das Dialogfeld "SOAP Debugger-Optionen" (*Abbildung unten*) können Sie die IP-Adresse des Computers und andere unten aufgelistete Debugger-Optionen konfigurieren. Das Dialogfeld wird über den Menübefehl **SOAP | SOAP Debugger-Optionen** aufgerufen.



- *Timeout:* Dieser Wert ist die Zeitspanne, die der SOAP Debugger an einem Breakpoint stehen bleibt. Die Standardeinstellung ist 5 Sekunden.
- *Eingabehilfen ausblenden; Projekt-/Info-Fenster ausblenden:* Diese Optionen sind nützlich, um mehr Platz für das SOAP Debugger-Fenster zu haben.

© 2016-2022 Altova GmbH

- *Adresse des Rechners:* Die Adresse des Proxy-Servers, über den der Debugger ausgeführt wird. Der Debugger auf dem Proxy Server erhält Requests von Rechnern im Netzwerk und sendet diese an den Webservice. Da der Debugger in XMLSpy ausgeführt wird, dient der Rechner, auf dem XMLSpy installiert ist, auch als Proxy Server. Die IP-Adresse des Rechners wird automatisch ermittelt und in dieses Feld eingetragen. Sie müssen die IP-Adresse nur dann (als `http`-Adresse) in dieses Feld eingeben, wenn die IP-Adresse nicht automatisch ermittelt werden kann. Um die IP-Adresse Ihres Rechners herauszufinden, öffnen Sie ein Eingabeaufforderungsfenster, geben Sie den Befehl `ipconfig /all` ein und drücken Sie die **Eingabetaste**.

15.2.2.3 Starten einer Debugger-Sitzung

Eine SOAP Debugger-Sitzung kann jederzeit, wenn eine beliebige Datei in XMLSpy aktiv ist, gestartet werden. Beim Starten des SOAP Debuggers muss weder einer SOAP-Datei noch eine SOAP-Request-Eintrittspunktdatei[808] aktiv sein. Beim Starten des SOAP Debuggers werden Sie nach folgenden Dingen gefragt:

1. dem Pfad der WSDL-Datei, aus der die relevanten SOAP-Informationen stammen, und
2. den Verbindungseinstellungen.

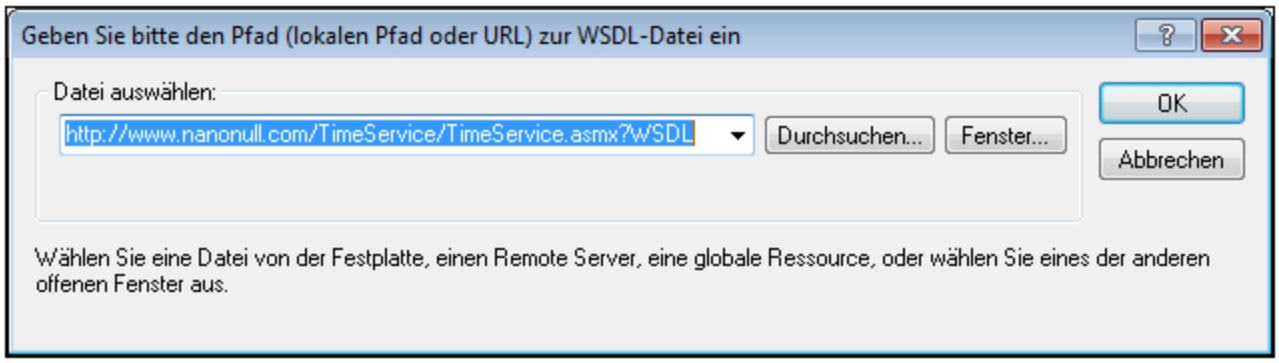
Diese Schritte sind weiter unten beschrieben.

WSDL-Dateipfad

Wenn eine Debugger-Sitzung gestartet wird, werden Sie nach der URL der WSDL-Datei, aus der die gewünschten SOAP-Informationen für unser Beispiel stammen, gefragt. In unserer Beispieldatei[808], `DebuggerClient.html`, wird die folgende WSDL-Datei-URL verwendet:

```
http://www.nanonull.com/TimeService/TimeService.asmx?WSDL
```

Starten Sie den SOAP Debugger durch Auswahl des Menübefehls **SOAP | SOAP Debugger-Sitzung**. Daraufhin wird das Dialogfeld "Geben Sie bitte den Pfad....ein" (*Abbildung unten*) geöffnet.

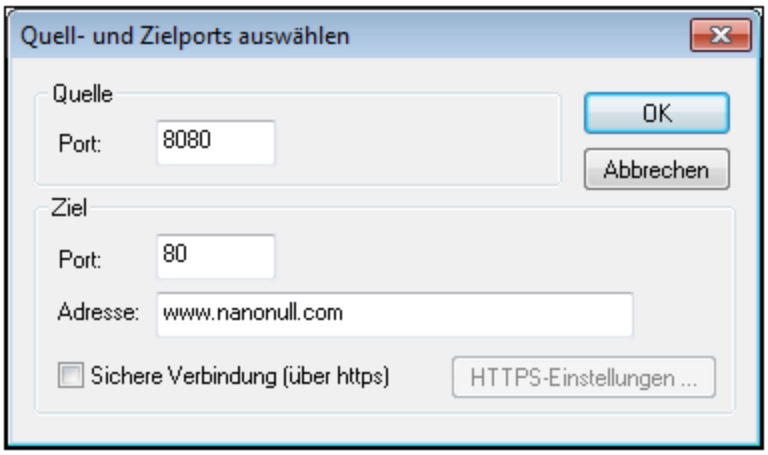


Geben Sie die URL der WSDL-Datei ein und klicken Sie auf **OK.** Daraufhin erscheint ein Dialogfeld zur Eingabe von Quell- und Ziel-Port (*Abbildung unten*).

Verbindungseinstellungen

Das Dialogfeld " Verbindungseinstellungen für Quelle und Ziel" (*Abbildung unten*) bietet die folgenden Einstellungen:

- *Quell-Port:* Der Port auf einem Proxy Server (der auch Ihr Computer sein kann), der für die Kommunikation verwendet wird. Die Standardeinstellung ist `8080`. Sie können diese Einstellung jedes Mal, wenn Sie den Debugger starten, ändern.
- *Ziel-Port und Adresse*: Die Einstellung für den Ziel-Port stammt aus der WSDL-Datei und wird ebenfalls automatisch ausgefüllt. Der Standard-Port ist der HTTP-Standard-Port `80`. Sie können ein Timeout für die Verbindung definieren oder die Option *Unendlich* aktivieren, wenn Sie kein Timeout wünschen. Um die HTTP-Sicherheitseinstellungen zu konfigurieren, klicken Sie auf die Schaltfläche **Bearbeiten** des Bereichs "HTTP-Sicherheitseinstellungen" und geben Sie die Sicherheitseinstellungen ein. Informationen zu diesen Einstellungen finden Sie im Abschnitt SOAP Request-Einstellungen[1525].



Klicken Sie nun auf die Schaltfläche **OK,** um die SOAP Debugger-Sitzung zu starten. Daraufhin wird der SOAP Debugger Proxy Server im **deaktivierten** Status geöffnet. Sie sehen den Status daran, dass das Proxy Server-Symbol deaktiviert ist. Um den Proxy Server (das Debuggen) zu starten, klicken Sie in der XMLSpy-Symbolleiste auf die Schaltfläche "**Start**" oder wählen Sie den Menübefehl **SOAP | Start**. Nähere Informationen zum eigentlichen Debuggen finden Sie im Abschnitt Debuggen[811].

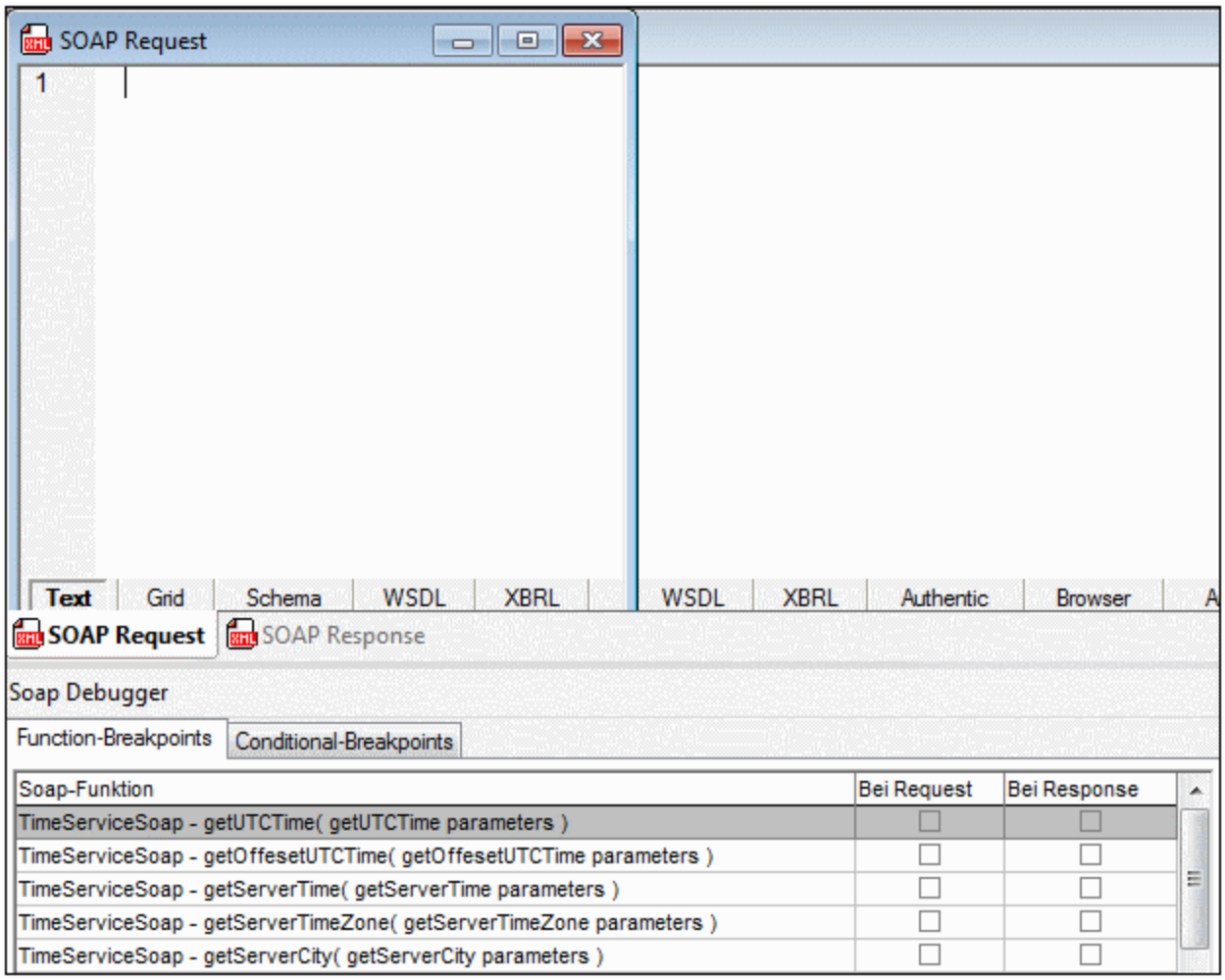
SOAP Debugger Layout

Der SOAP Debugger besteht aus drei Fenstern (*siehe Abbildung unten*):

- einem SOAP Request-Fenster,
- einem SOAP Response-Fenster und
- einem Fenster für die SOAP Debugger Breakpoint-Einstellungen.

In der Standardeinstellung werden das Request- und das Response-Fenster im oberen Bereich von XMLSpy geöffnet und das Fenster für die Breakpoint-Einstellungen wird im unteren Bereich über die ganze Breite angezeigt. In der Abbildung unten sehen Sie das Standard-Layout.

© 2016-2022 Altova GmbH



Durch Ausblenden der XMLSpy Seitenleisten (Projekt, Info und Eingabehilfen) erhalten Sie mehr Platz für die Debugger-Fenster. Die Einstellungen zum Ein-/Ausblenden dieser Fenster finden Sie im Dialogfeld SOAP Debugger-Optionen[804] (Aufruf über den Menübefehl **SOAP | SOAP Debugger-Optionen**).

Informationen zu vertrauenswürdigen Zertifikaten

Altova Produkte verwenden zum Aufrufen und Verwalten vertrauenswürdiger Zertifikate sicherer Webserver Internet Explorer (IE). Wenn Sie das Zertifikat eines Webservers in IE installieren, greift IE auf den Webserver zu, ohne eine Warnmeldung auszugeben oder den Vorgang abzubrechen. Um das Zertifikat eines sicheren Webservers zu installieren, gehen Sie folgendermaßen vor. Je nach verwendetem Browser können die Schritte geringfügig unterschiedlich sein.

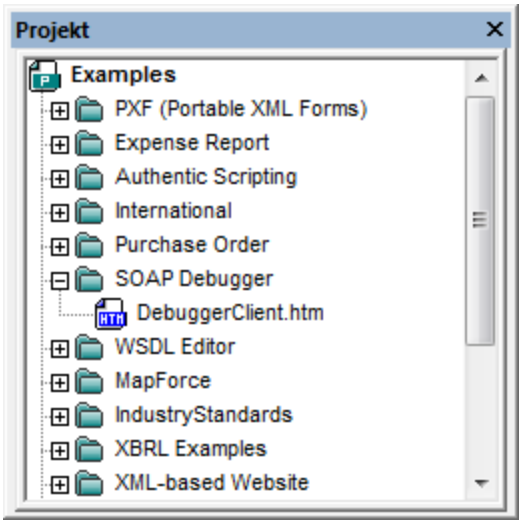
1. Öffnen Sie die sichere Website in Internet Explorer 9 (oder einer höheren Version).
2. Wählen Sie den Befehl **Datei | Eigenschaften** und klicken Sie auf die Schaltfläche "Zertifikate".
3. Klicken Sie auf **Zertifikat installieren** und starten Sie den Assistenten zum Import eines Zertifikats. (Dieser Assistent kann auch über **Extras | Internet Optionen| Inhalt | Zertifikate | Importieren** aufgerufen werden.)
4. Das Zertifikat sollte in den Speicher vertrauenswürdiger Stammzertifizierungsstellen platziert werden, zu dem Sie manuell navigieren können.

5. Wenn Sie die einzelnen Schritte im Assistenten fertig gestellt haben, schließen Sie die Dialogfelder "Zertifikate" bzw. "Eigenschaften" durch Klicken auf OK. Eventuell müssen Sie Internet Explorer neu starten.

15.2.2.4 SOAP-Request-Eintrittspunkt

Das Projekt "Examples" enthält eine HTML-Datei (`DebuggerClient.htm`), die ein Skript enthält, anhand dessen gezeigt wird, wie der SOAP Debugger verwendet wird. Mit Hilfe dieser Datei kann der Benutzer einen SOAP Request an einen Webservice senden und anschließend die Response des Webservice anzeigen. Öffnen Sie diese Datei folgendermaßen in XMLSpy:

1. Wählen Sie die Menüoption **Projekt | Öffnen**.
2. Wählen Sie die Datei "`Examples.spp`" im Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples`. Daraufhin wird das Projekt "Examples" im Projektfenster (*Abbildung unten*) angezeigt.



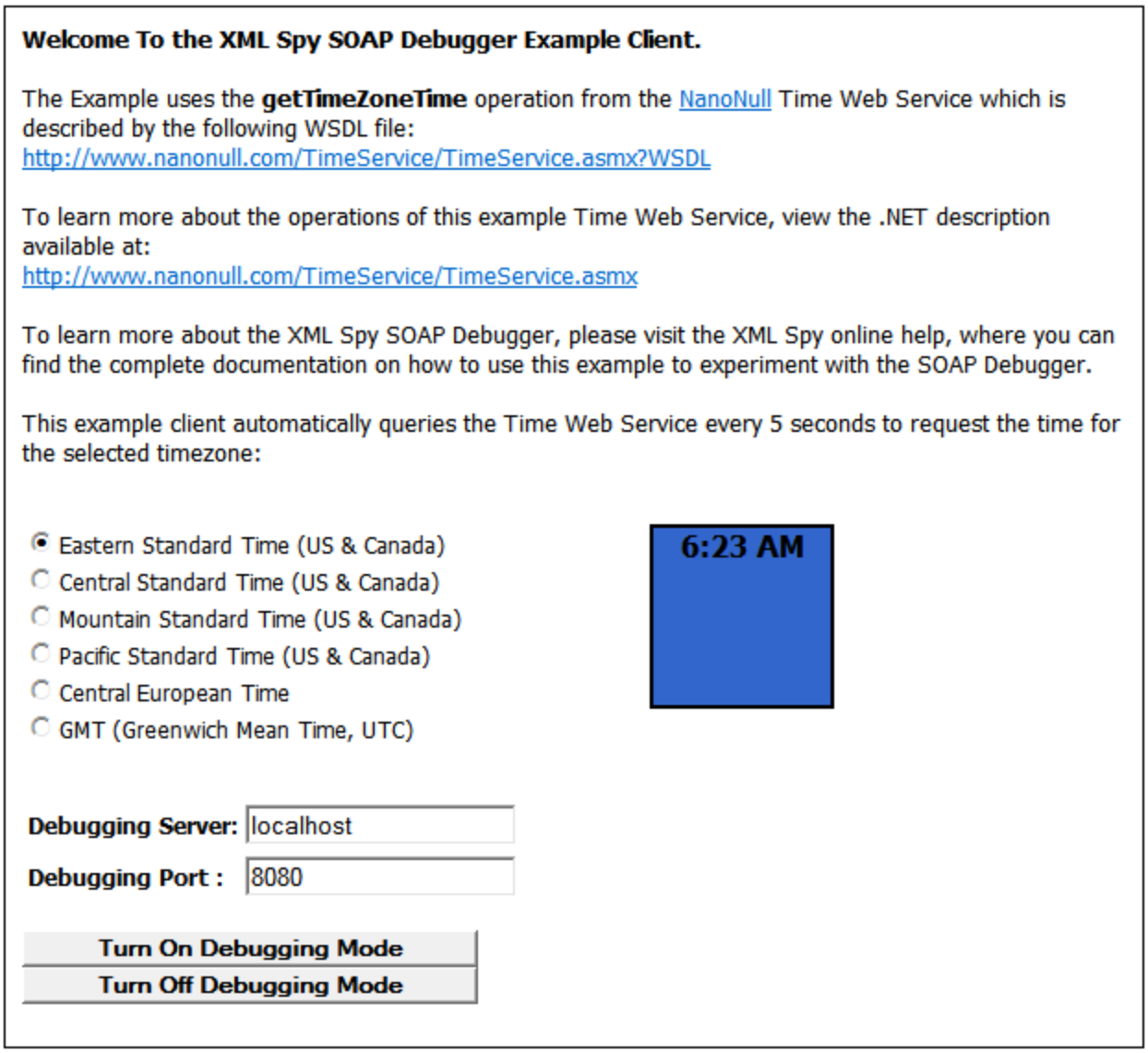
3. Klicken Sie auf das +-Zeichen des Ordners SOAP Debugger, um den Inhalt des Ordners anzuzeigen. Doppelklicken Sie auf die Datei `DebuggerClient.htm`, um sie in XMLSpy zu öffnen.

Anmerkung: Sie können diese Datei (`DebuggerClient.htm`) auch mit dem Befehl **Datei | Öffnen** öffnen. Die Datei befindet sich im Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples`.

Die Beispieldatei

Die Datei `DebuggerClient.htm` sieht in der Browser-Ansicht von XMLSpy in etwa folgendermaßen aus. Wenn eines der Optionsfelder ausgewählt ist, wird ein SOAP Request an den Nanonull Time Webservice gesendet. Die Response des Webservice wird im färbigen Kästchen rechts von den Optionsfeldern angezeigt.

© 2016-2022 Altova GmbH



Beachten Sie, dass die Response auf eine Eastern Standard Time-Anfrage (`6:23`) in der Abbildung oben in einem blauen Uhrzeitkästchen angezeigt wird. Klicken Sie auf das Optionsfeld GMT. Anstatt den GMT-Wert anzuzeigen, wird eine Fehlermeldung angezeigt und das Kästchen wird rot angezeigt (*siehe Abbildung unten*).



Um die Ursache des Problems zu finden und beheben zu können, müssen wir den SOAP Debugger aktivieren, um die SOAP Messages zu analysieren. In den folgenden drei Abschnitten wird beschrieben, wie man (i) Breakpoints setzt[810], (ii) wie man den SOAP Debugger[811] mit `DebuggerClient.htm` ausführt und (iii) wie man die SOAP Debugger-Ausgabe analysiert, um Fehler zu finden[812].

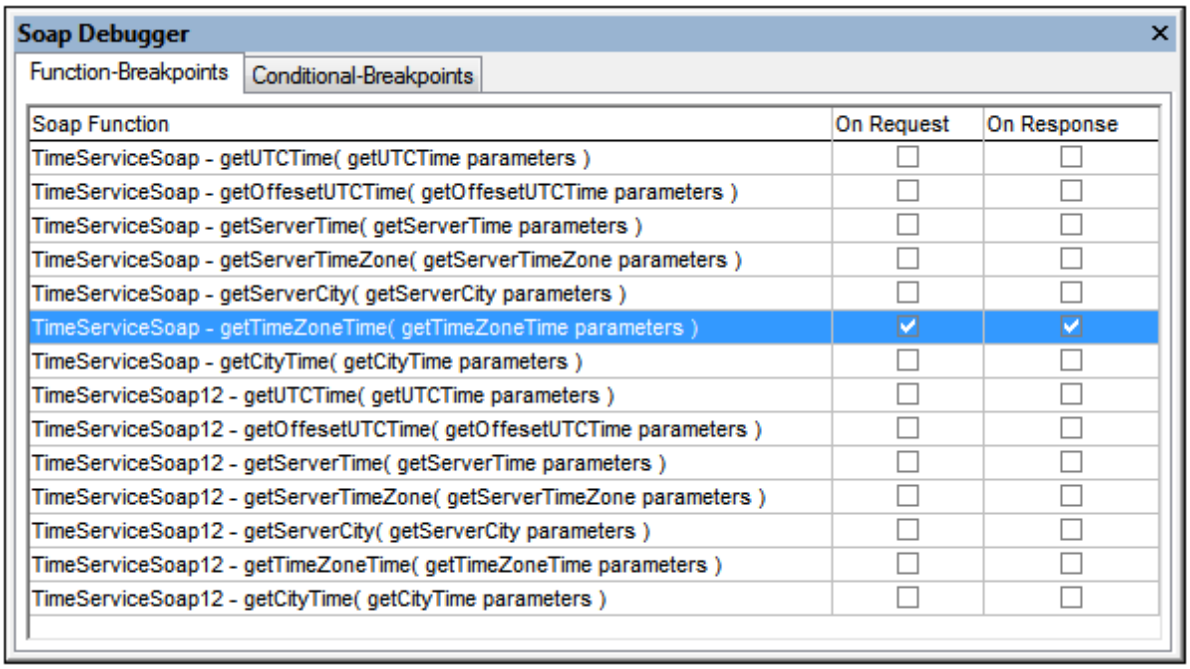
15.2.2.5 Setzen von Breakpoints

Bevor Sie mit dem Debuggen beginnen, müssen Sie im SOAP Debugger Breakpoints setzen. Sobald er mit dem Debuggen startet, zeigt der Debugger die Requests und Responses an den gesetzten Breakpoints an.

Der SOAP Debugger listet die Breakpoints (in den Bereichen *Function-Breakpoints* und *Conditional-Breakpoints*) gemäß den Informationen aus der WSDL-Datei, die zum Zeitpunkt des SOAP Debugger-Starts[805] ausgewählt war, auf. Diese Breakpoints beziehen sich auf die SOAP Requests, die von der WSDL-Datei generiert werden können. Für jeden SOAP Request kann durch Aktivieren der entsprechenden Kontrollkästchen in der jeweiligen Spalte (*siehe Abbildung unten*) ein Breakpoint beim Request und einer bei der Response ausgewählt werden.

In unserem Beispiel werden folgende Dateien verwendet:

- `DebuggerClient.htm` als der SOAP-Request Eintrittspunkt[808] und
- die WSDL-Datei `http://www.nanonull.com/TimeService/TimeService.asmx?WSDL`, die zum Zeitpunkt des SOAP Debugger-Starts ausgewählt war[805].



Der von `DebuggerClient.htm` gefragte Webservice ermittelt mit Hilfe der Methode `getTimeZoneTime` die Uhrzeit in der ausgewählten Zeitzone. SOAP Requests, die im SOAP Debugger anhand der ausgewählten WSDL-Datei generiert werden können, werden als Breakpoints aufgelistet. Wir setzen nun einen Breakpoint bei `getTimeZoneTime` sowohl für *Bei Request* als auch für *Bei Response* (*siehe Abbildung oben*). Auf diese Art können Sie sowohl die SOAP Request als auch die Webservice Responses auf Fehler analysieren.

© 2016-2022 Altova GmbH

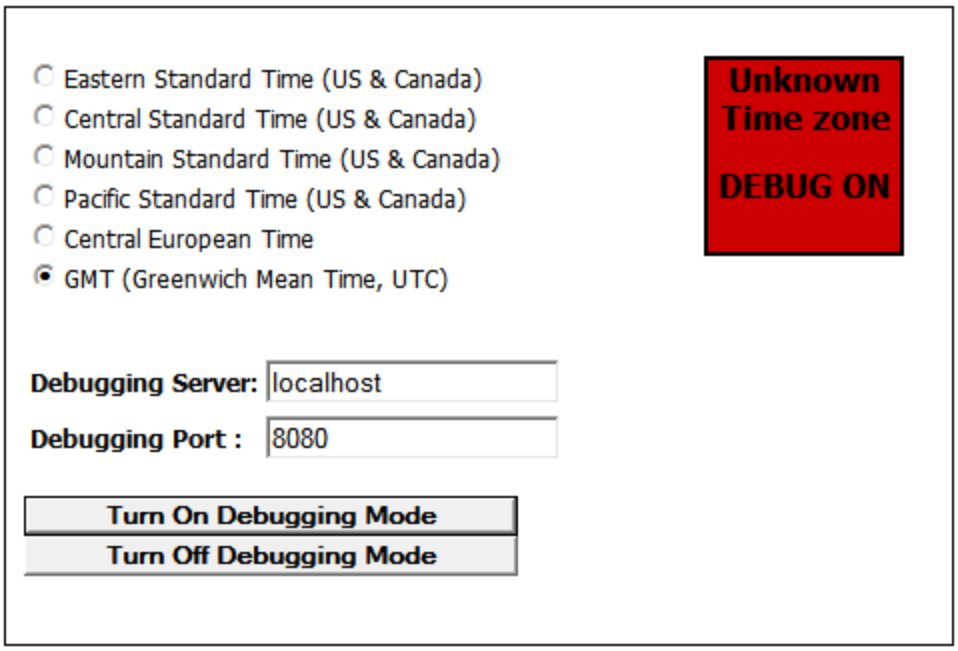
Nähere Informationen zum Setzen von Breakpoints finden Sie im Abschnitt Mehr über Breakpoints[814].

15.2.2.6 Debuggen

In unserem Beispiel werden folgende Dateien verwendet:

- `DebuggerClient.htm` als der SOAP-Request Eintrittspunkt[808] und
- die WSDL-Datei `http://www.nanonull.com/TimeService/TimeService.asmx?WSDL`, die zum Zeitpunkt des SOAP Debugger-Starts ausgewählt war[805].

Nachdem Sie Breakpoints gesetzt haben, klicken Sie auf das **Start**-Symbol  (oder wählen Sie den Menübefehl **SOAP | Start**). Klicken Sie auf das Register *DebuggerClient.htm,* um zur SOAP-Eintrittspunktdatei zu wechseln. Stellen Sie sicher, dass die Option GMT ausgewählt ist und klicken Sie auf die Schaltfläche "Turn On Debugging Mode" (*siehe Abbildung unten).* Daraufhin wird die Meldung "Debug on" angezeigt und der SOAP Request wird an den Debugger gesendet. Die Ergebnisse des Debugging werden in den Fenstern "SOAP Request" und "SOAP Response" angezeigt und sind im nächsten Abschnitt Analysieren der Ergebnisse und Fehlerbehebung[812] erläutert.



SOAP Debugger-Steuerelemente

Die Symbolleiste "SOAP Debugger" (*Abbildung unten*) enthält Schaltflächen für die Bedienung des SOAP Debuggers.



Diese Schaltflächen sind von links nach rechts:

- *Start:* Startet das Debuggen.
- *Einzelschritt:* Geht den Request-Response-Vorgang schrittweise durch und hält bei Breakpoints an.
- *Bei nächstem Request anhalten:* Hält beim nächsten SOAP Request an.
- *Bei nächstem Response anhalten:* Hält beim nächsten Response des Webservice an.
- *Proxyserver anhalten:* Stoppt das Debugging. Beachten Sie, dass dies nicht dasselbe ist, wie das Beenden der SOAP Debugger-Sitzung. Um die SOAP Debugger-Sitzung zu beenden/zu starten, wählen Sie den Menübefehl **SOAP | SOAP Debugger-Sitzung**.

15.2.2.7 Analysieren der Ergebnisse und Fehlerbehebung

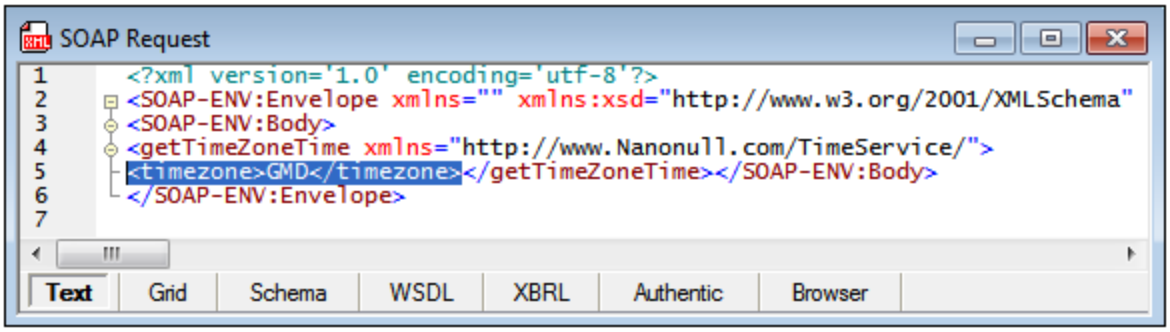
Die SOAP Debugger-Ergebnisse werden in zwei Fenstern angezeigt: SOAP Request und SOAP Response. Die Breakpoints werden in den SOAP Debugger Breakpoints-Bereichen unterhalb des SOAP Debugger-Fensters gesetzt.

Entsprechend den gesetzten Breakpoints werden die Ergebnisse im entsprechenden Ergebnisfenster, nämlich SOAP Request oder SOAP Response, angezeigt:
In unserem Beispiel werden folgende Dateien verwendet:

- `DebuggerClient.htm` als der SOAP-Request Eintrittspunkt[808] und
- die WSDL-Datei `http://www.nanonull.com/TimeService/TimeService.asmx?WSDL`, die zum Zeitpunkt des SOAP Debugger-Starts ausgewählt war[805].

Suchen des Fehlers und Testen, ob der Fehler behoben ist

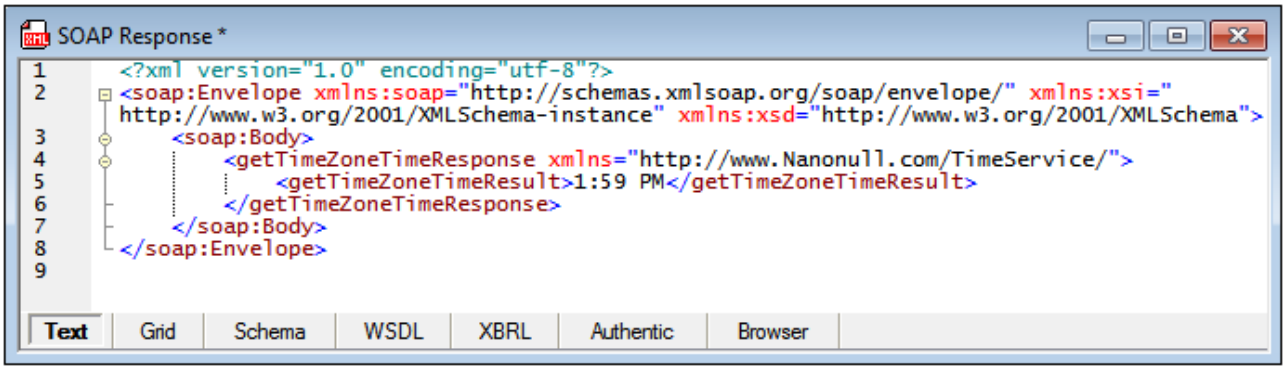
Das Debuggen wurde, wie im vorigen Abschnitt Debuggen[811] beschrieben, gestartet. Der SOAP Request für die Auswahl "GMT" wird automatisch im SOAP Request-Fenster des Debuggers in der Textansicht angezeigt. Wir wollen uns den Request nun ansehen und etwaige Fehler beheben.



Beim Element **timezone** stellen wir fest, dass der Wert `GMD` ist. Dies ist falsch und muss in `GMT`geändert werden. Doppelklicken Sie in das Element `timezone`, ändern Sie den Eintrag in `GMT`.

Klicken Sie auf das **Start**-Symbol (oder wählen Sie den Menübefehl **SOAP | Starten**), um den korrigierten Request an den Webservice zu senden. Nach wenigen Sekunden erscheint der Webservice Response auf den SOAP Request im SOAP Response-Fenster. Wählen Sie **Ansicht | Zeilenumbruch**, um den gesamten SOAP Response zu sehen (*siehe Abbildung unten*).

© 2016-2022 Altova GmbH



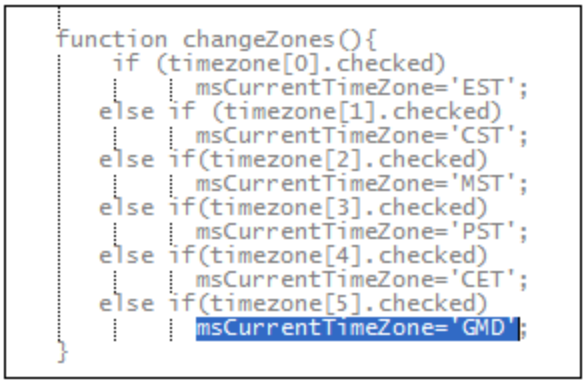
Wechseln Sie auf das Register `DebuggerClient.htm`, klicken Sie auf das Debugger **Start**-Symbol. Die Fehlermeldung wird nicht mehr angezeigt, stattdessen sehen Sie die korrekte GMT-Zeit. (*Abbildung unten*).



Wählen Sie den Menübefehl **SOAP | SOAP Debugger-Sitzung**, um die Debugger-Sitzung zu beenden.

Fehlerbehebung

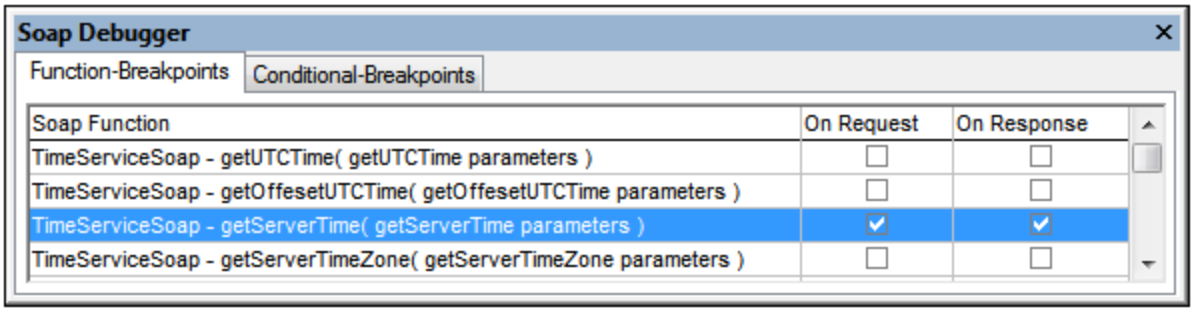
Wir wissen nun, dass der unzulässige Wert "`GMD`" anstatt "`GMT`" im SOAP Request generiert wird. Wenn wir die SOAP Request-Eintrittspunktdatei nach `GMD` durchsuchen (über das Dialogfeld "Suchen", mit **Strg+F** oder mit **Bearbeiten | Suchen**), finden wir den Tippfehler im unten gezeigten Codefragment.



Wenn Sie diesen Fehler korrigieren und anschließend das Optionsfeld GMT auswählen, wird der Fehler "*Unknown Timezone*" nicht mehr angezeigt. Stattdessen erscheint im Fenster die richtige Uhrzeit.

15.2.2.8 Mehr über Breakpoints

Das Setzen und Löschen von Breakpoints erfolgt im SOAP Debugger-Fenster. Es besteht aus den folgenden zwei Registerkarten (*Abbildung unten*):



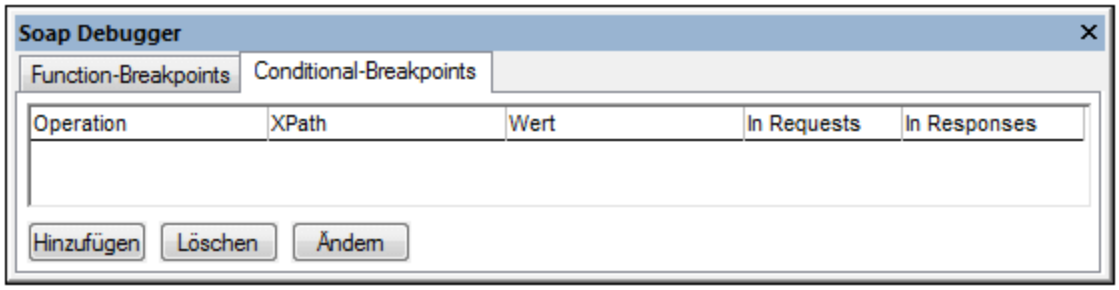
Function-Breakpoints

Das Register "Function breakpoints" gestattet Ihnen, in Requests und/oder Responses auf SOAP-Methoden Breakpoints zu setzen. Der Debugger markiert die Funktion, die den Breakpoint ausgelöst hat. Datenpakete, die vom und an den Client gesendet werden, werden analysiert und den entsprechenden Funktionen aus der WSDL-Datei zugeordnet. Wenn ein Breakpoint für eine bestimmte Methode gesetzt wird, dann hält der SOAP Debugger an dieser Stelle an. Die Symbolleistenschaltflächen werden zu diesem Zeitpunkt aktiviert.

Die Daten werden in den Dokumentenfenstern "SOAP Request" oder "SOAP Response" angezeigt. Die in den SOAP-Fenstern angezeigten SOAP-Dokumente können nun geändert werden. Die Daten werden gesendet, sobald Sie auf eine der Symbolleistenschaltflächen (mit Ausnahme des Symbols "Server anhalten") klicken.

Conditional-Breakpoints

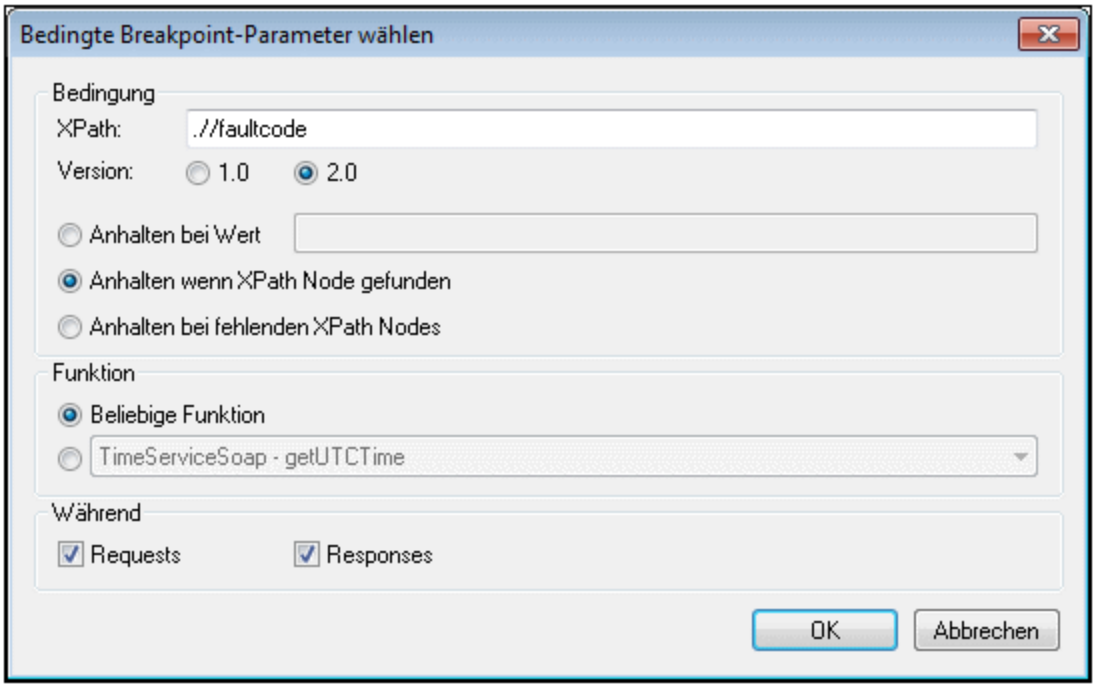
Auf dem Register "Conditional Breakpoints" (*Abbildung unten*) können Sie XPath-Ausdrücke verwenden, um Breakpoints zu definieren. Wenn ein SOAP **Request** einen Fehler verursacht, muss der SOAP **Response** ein `faultcode`-Element enthalten. Wir möchten daher, dass an jedem `faultcode`-Element ein Breakpoint ausgelöst wird.



So fügen Sie einen bedingten Breakpoint (conditional Breakpoint) hinzu:

1. Klicken Sie auf das Register "Conditional Breakpoints" und anschließend auf die Schaltfläche **Hinzufügen.** Daraufhin wird das unten gezeigte Dialogfeld angezeigt.

© 2016-2022 Altova GmbH

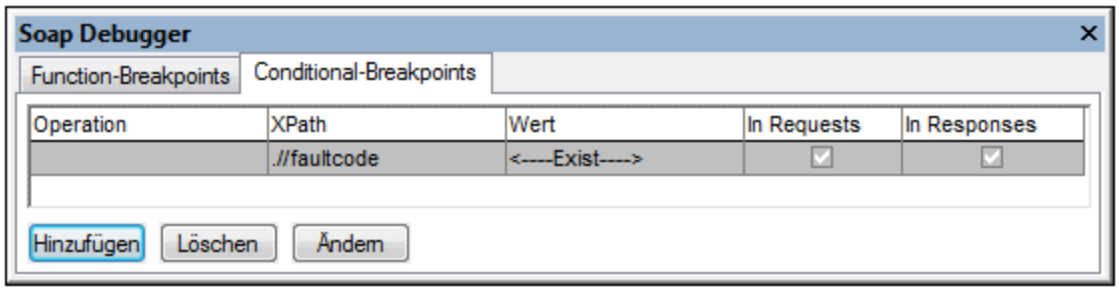


2. Geben Sie den XPath-Ausdruck (z.B. `.//faultcode`) in das Feld XPath ein.
3. Wählen Sie die XPath-Version aus (1.0 oder 2.0) und aktivieren Sie das Optionsfeld "*Anhalten wenn XPath Node gefunden*".
4. Bestätigen Sie die Einstellungen durch Klicken auf OK.
   Der SOAP Debugger hält nun immer dann an, wenn im SOAP Request oder Response ein `.//faultcode`-Element erscheint.

Im Folgenden sind die verschiedenen Optionen in diesem Dialogfeld beschrieben:

- Feld *XPath*: Geben Sie hier den gewünschten XPath-Ausdruck/Node ein. Sie müssen hier einen XPath eingeben, um eine der Optionen der Optionsfelder verwenden zu können.
- *Version:* Die XPath-Version, die für den XPath-Ausdruck verwendet werden soll.
- Optionsfelder "*Anhalten wenn,...*": Der Debugger hält an, wenn eine der ausgewählten Optionen zutrifft. Zur Auswahl stehen die folgenden Optionen: (i) Der Debugger hält an, wenn einer der XPath Nodes mit dem in diesem Feld angegebenen Wert übereinstimmt. (ii) Der Debugger hält an, wenn der angegebene XPath-Node im SOAP Request oder der Response vorkommt und (iii) Der Debugger hält an, wenn der angegebene XPath-Node im SOAP Request oder der SOAP Response nicht vorkommt.
- *Requests und Responses*: Gibt an, ob die Optionen im Dialogfeld in SOAP Responses und/oder Requests angewendet werden sollen.
- *Funktionen*: Es werden entweder alle Methoden/Funktionen nach der von Ihnen definierten Bedingung durchsucht (Optionsfeld "*Beliebige Funktion*") oder Sie geben eine bestimmte Methode/Funktion an, die durchsucht werden soll.

Für die im oben gezeigten Dialogfeld definierte Bedingung wird auf dem Register "Conditional-Breakpoints" der folgende bedingte Breakpoint aufgelistet.



Im Folgenden finden Sie eine Beschreibung der Spalten auf diesem Register:

- Die Spalte *Operation* enthält die Methode, nach der gesucht wird. Wenn Sie das Optionsfeld *Beliebige Funktion* aktiviert haben, bleibt dieses Feld leer. Wenn Sie eine bestimmte Methode/Funktion ausgewählt haben, wird diese Methode/Funktion hier angezeigt.
- Die Spalte *XPath* enthält den von Ihnen definierten XPath-Ausdruck.
- Die Spalte *Wert* enthält den XPath-Wert, auf den die zurückgegebenen Nodes überprüft werden. Wenn Sie die Option "*Anhalten bei Wert*" gewählt haben, wird der von Ihnen eingegebene String hier angezeigt. Wenn Sie die Option *Anhalten wenn XPath Node gefunden*" gewählt haben, so wird <--Exist--> angezeigt. Wenn Sie die Option "*Anhalten bei fehlenden XPath Nodes*" gewählt haben, wird <--Missing--> angezeigt.
- Die Kontrollkästchen *In Requests* und *In Responses* geben an, wo die Bedingung überprüft wird. Sie können die Einstellungen ändern, indem Sie das Kontrollkästchen direkt in der Spalte aktivieren.

Um bedingte Breakpoints zu bearbeiten, doppelklicken Sie auf die entsprechende Zeile im Register oder klicken Sie auf die Schaltfläche "Ändern" (*Siehe Abbildung oben*). Um einen bedingten Breakpoint zu löschen, klicken Sie auf die zu löschende Zeile und anschließend auf die Schaltfläche "Löschen".

© 2016-2022 Altova GmbH

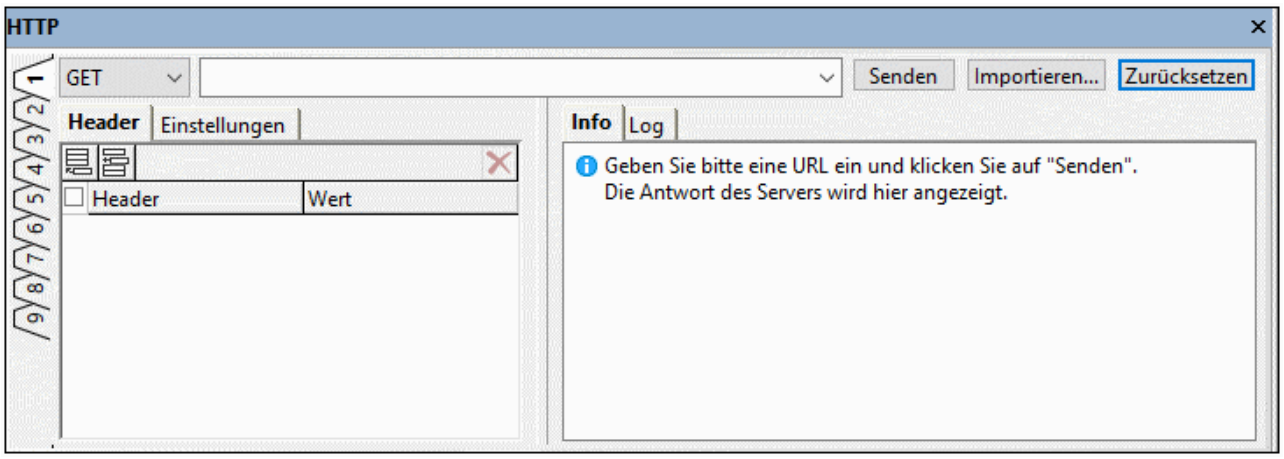
16 HTTP

HTTP (Hypertext Transfer Protocol) ist das Protokoll (oder Regelwerk), in dem definiert ist, wie Dateien (Text, Bilder, Audio, Video und andere Multimediadateien) über das Internet übertragen werden. Auf jedem Webserver läuft ein Programm (ein sogenannter Daemon), das ständig auf HTTP Requests wartet und jeden davon bei Empfang behandelt. Wenn Sie z.B. die Homepage einer Website besuchen, sendet Ihr Browser einen HTTP-Befehl an den Webserver der Website, der den Download der Homepage anfordert; der HTTP-Daemon des Servers empfängt den Request und sendet die gewünschte Seite. Eine wichtige Eigenschaft von HTTP ist, dass es *zustandslos* ist, d.h. jeder HTTP-Befehl wird unabhängig, ohne jegliche Bezugnahme auf vorhergehende oder folgende Befehle ausgeführt.

Sie können HTTP-Befehle in XMLSpy im HTTP-Ausgabefenster[126] (*Abbildung unten*) testen. Hier können Sie einen HTTP Request erstellen, an einen Webserver senden und die Response (Antwort) empfangen und überprüfen.

Bestandteile des HTTP-Ausgabefensters

Das HTTP-Ausgabefenster hat neun Register (*siehe Abbildung unten*). Sie können auf jedem Register einen separaten Request speichern und zwischen den Registern wechseln. Nach Erstellung eines Request können Sie diesen durch Klick auf die Schaltfläche **Senden** senden. Die Antwort (Response) wird direkt im Fenster angezeigt.



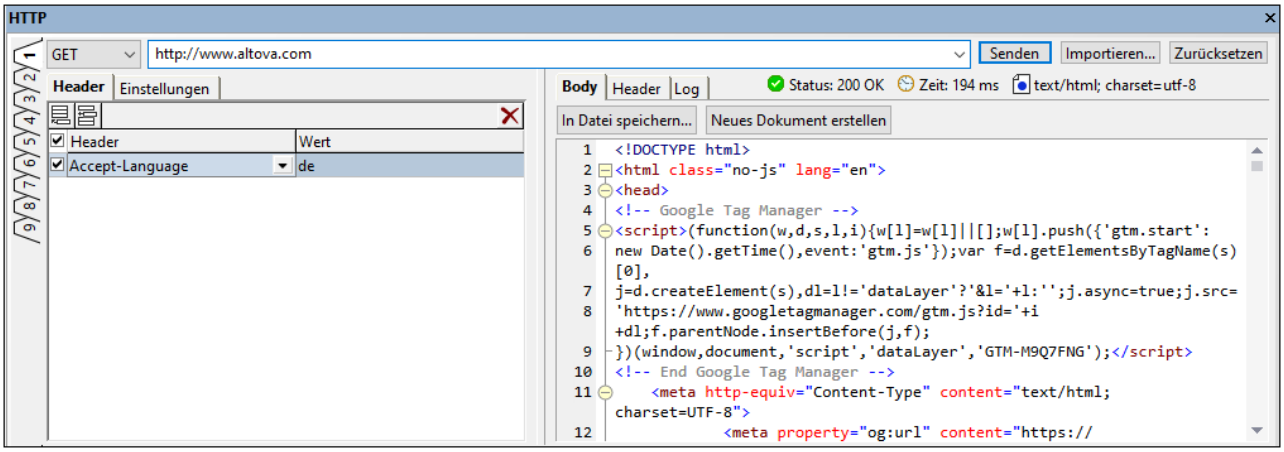
Das Fenster besteht aus den folgenden Teilen:

- Im oberen Bereich finden Sie: (i) eine Auswahlliste, in der Sie die gewünschte HTTP-Methode auswählen können; (ii) ein Eingabefeld für die URL des Webservers; (iii) Schaltflächen im Zusammenhang mit der Ausführung von HTTP Requests (**Senden**, **Importieren** und **Zurücksetzen**).
- Einem Bereich auf der linken Seite zum Erstellen des Request.[818]
- Einem Bereich auf der rechten Seite zur Anzeige der Response[825].

In den Unterabschnitten dieses Abschnitts finden Sie eine Beschreibung der Funktionsweise des HTTP-Ausgabefensters.

16.1 Senden des Request

Sie können im HTTP-Ausgabefenster (*Abbildung unten*) einen HTTP Request senden. Ein Request wird im Bereich auf der **linken Seite des Fensters definiert**. Sie können für jedes der neun Register des Fensters einen anderen Request definieren. Jeder Request besteht aus den folgenden Bestandteilen: (i) der HTTP-Methode[819] des Request und der Ziel-URL (im oberen Teil des Dialogfelds definiert); (ii) den HTTP Headern des Request (auf dem Register *Header*); (iii) den Verbindungseinstellungen (auf dem Register *Einstellungen*) und (iv) im Fall der POST- und PUT-Methode[819], dem HTTP Message Body (auf dem Register *Body*; wird in der Abbildung unten nicht angezeigt). Durch Klicken auf **Zurücksetzen** (rechts oben im Fenster) können Sie einen Request auf den leeren Zustand zurücksetzen.



Um einen HTTP Request zu senden, gehen Sie folgendermaßen vor:

1. Wählen Sie in der Auswahlliste links oben (*siehe Abbildung*) eine HTTP-Methode aus (**GET**, **POST**, **PUT**, **DELETE**, **HEAD** oder **OPTIONS**).
2. Geben Sie die URL der Ziel-Webseite ein (z.B. **http://www.altova.com**. Sie können auch nur **altova.com** eingeben; der **http://** Teil der URL wird für Sie ausgefüllt).
3. Auf dem Register *Header* können Sie die HTTP-Header-Werte definieren (*siehe Abbildung oben*). Sie können einen Header auswählen oder eingeben und anschließend seinen Wert eingeben. (Eine Liste von HTTP 1.1-Headern finden Sie hier). Über die Schaltflächen **Einfügen**, **Anhängen** und **Löschen** der Symbolleiste des Registers können Sie Header hinzufügen oder löschen. Anstatt einen Header zu löschen, können Sie diesen auch durch Deaktivieren des Kontrollkästchens *Aktivieren* links vom Namen des Headers deaktivieren. Dadurch müssen Sie die Informationen für einen Header nicht erneut eingeben, sollten Sie ihn später wieder verwenden wollen. Nähere Informationen dazu finden Sie im Abschnitt Der Accept Header[827]. (Wenn Sie für einen Header einen Wert, der beim Senden automatisch hinzugefügt würde, definieren, so wird anstatt des automatisch hinzugefügten Werts dieser von Ihnen definierte Wert verwendet.)
4. Bei Sendung eines **POST**- oder **PUT**-Request steht neben den Registern *Header* und *Einstellungen* ein zusätzliches Register *Body* zur Verfügung. Eine Anleitung zur Erstellung des Body eines **POST**- oder **PUT**-Request finden Sie im Abschnitt Der Body von POST- und PUT-Requests[819].
5. Auf dem Register *Einstellungen* können Sie Timeout- und Sicherheitseinstellungen definieren. Eine Beschreibung dieses Registers finden Sie weiter unten unter Einstellungen für den HTTP Request[821].
6. Klicken Sie (*rechts oben im Fenster*) auf **Senden**, um den Request zu senden.
7. Um wieder den leeren Zustand eines Registers herzustellen, klicken Sie auf **Zurücksetzen**. Daraufhin geschieht Folgendes: (i) Die zu verwendende Methode wird auf die erste Methode in der Dropdown-

© 2016-2022 Altova GmbH

Auswahlliste zurückgesetzt (also auf **GET**); (ii) der aktuelle URL-Eintrag wird entfernt, (iii) alle Header-, Einstellungen- und Body-Definitionen werden entfernt.

Anmerkung: Sie können über die Schaltfläche **Importieren** 823 des Fensters einen Request auch aus einer WADL-Datei in das HTTP-Ausgabefenster importieren.

Anmerkung: Der Request wird in UTF-8-Kodierung gesendet. Alle anderen Kodierungen werden in UTF-8 konvertiert und die UTF-8-Daten werden gesendet.

HTTP-Methoden

Es werden die folgenden HTTP-Methoden unterstützt:

GET
Mit der **GET**-Methode wird die Ressource unter der angegebenen URL angefordert. Sie können zur URL auch eine Abfrage hinzufügen, z.B. `http://www.altova.com?name1=value1&name2=value2`. Die Ressource wird in einer Nachricht zurückgesendet, die einen Header und einen Body, also einen Kopf- und einen Hauptteil enthält.

HEAD
Die **HEAD**-Methode ist mit dem **GET**-Request identisch, gibt aber keinen Message Body sondern nur einen Message Header zurück, der Metainformationen über die Ressource unter der angegebenen URL enthält.

POST
Mit Hilfe der **POST**-Methode wird eine vorhandene Ressource unter der angegebenen URL aktualisiert oder es wird eine neue Ressource unter der angegebenen URL erstellt. Die Daten, die an die Ressource gesendet werden sollen, werden in den Body des HTTP Request platziert. Nähere Informationen dazu finden Sie unter *Der Body von POST- und PUT-Requests* 819.

PUT
Mit Hilfe der **PUT**-Methode wird unter der angegebenen URL eine neue Ressource erstellt. Die Daten, die an die Ressource gesendet werden sollen, werden in den Body des HTTP Request platziert. Nähere Informationen dazu finden Sie unter *Der Body von POST- und PUT-Requests* 819.

DELETE
Mit Hilfe der **DELETE**-Methode wird die Ressource unter der angegebenen URL gelöscht.

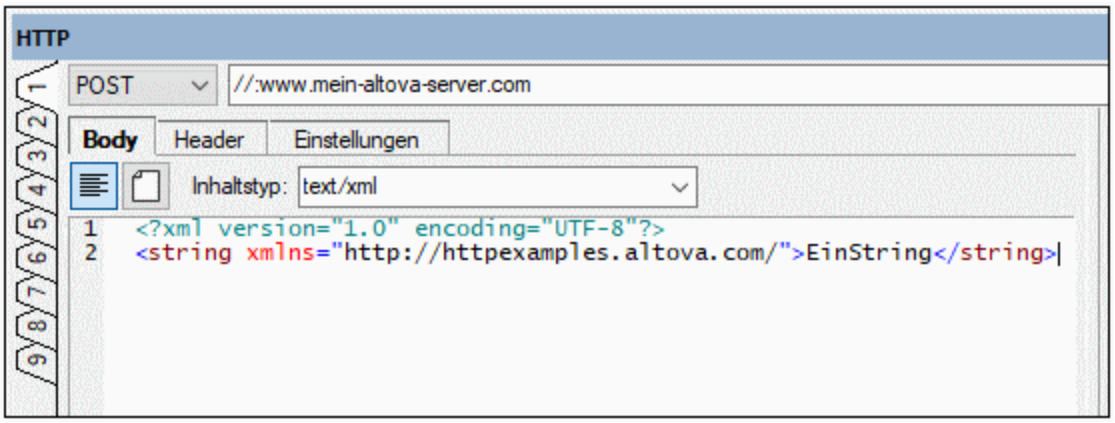
OPTIONS
Die **OPTIONS**-Methode gibt eine Liste der vom Server unterstützten HTTP-Methoden zurück.

Der Body von POST- und PUT-Requests

Bei **POST**- und **PUT**-Requests steht ein zusätzliches Register *Body* zur Verfügung, in dem der Body (Rumpf) des **POST**- oder **PUT**-Request definiert werden kann (*siehe Abbildung unten*). Das Register *Body* hat zwei Modi: **Editor** und **Datei**. Sie können über die Symbolleisten-Schaltflächen links oben auf dem Register *Body* (*siehe Abbildung*) zwischen diesen beiden Modi hin- und herwechseln. Im Editor-Modus (*in der Abbildung unten ausgewählt)* können Sie den HTTP-Request direkt im Fenster bearbeiten, während Sie im Datei-Modus eine Datei auswählen können, die den Body des HTTP-Request enthält.

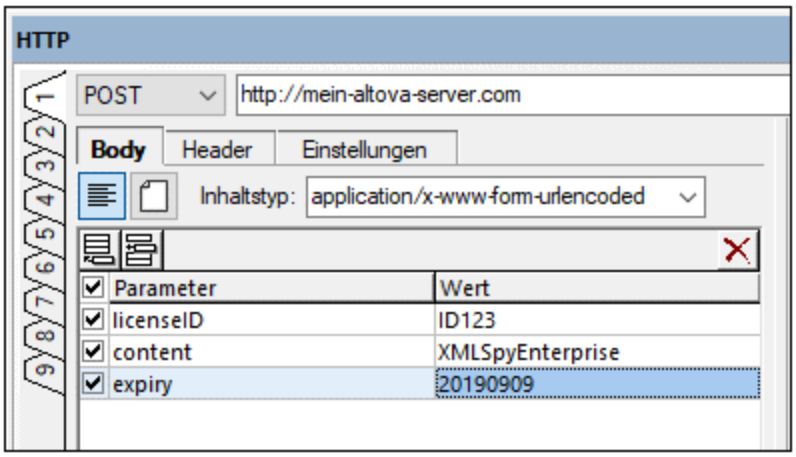
Über das Feld *Inhaltstyp* können Sie den `Content-Type` Header des Request definieren. Die Optionen der Auswahlliste für dieses Feld sind je nach Modus (Editor und Datei) unterschiedlich. Sie können eine der verfügbaren Auswahllistenoptionen auswählen oder einen MIME-Typ eingeben. Beachten Sie, dass der in

dieses Feld eingegebene Wert alle eventuell auf dem Register *Header* oder *Body* definierten **Content-Type**-Header außer Kraft setzt.



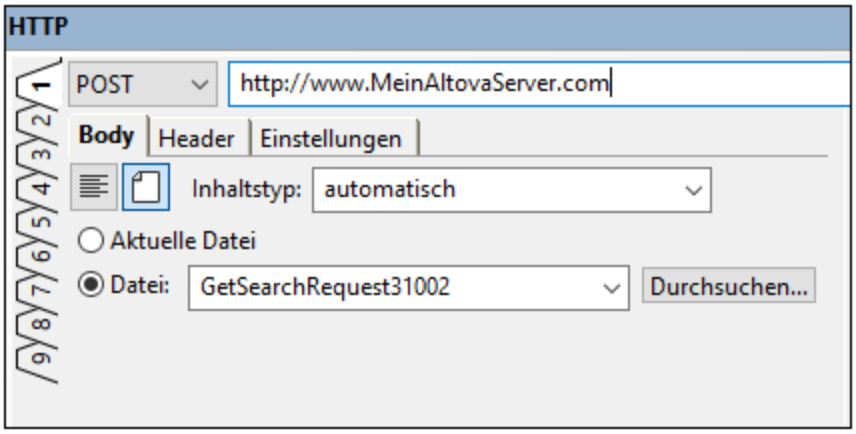
Editor-Modus
Der Stil des Editors hängt vom ausgewählten Inhaltstyp aus:

- Für die Inhaltstypen **text/plain**, **text/xml**, **application/xml** und **application/json**: Ein Text-Editor, der intelligente Bearbeitungsfunktionen wie Syntaxfärbung für XML- und JSON-Dokumente und Zeilennummerierung bereitstellt. In der Abbildung oben sehen Sie den Editor für den Inhaltstyp `text/xml`. Der Body des Request wird im Editor eingegeben. Der Inhaltstyp des Request wird im Feld *Inhaltstyp* definiert und kann nicht von Einträgen an anderen Stellen des Request außer Kraft gesetzt werden.
- Für den Inhaltstyp **application/x-www-form-urlencoded**: Der Editor ist eine Rasteransicht (*Abbildung unten*), in der jede neue Zeile für ein Name-Wert-Paar im Body des Request steht.



Datei-Modus
Im Datei-Modus (*Abbildung unten*) bildet der Inhalt der ausgewählten Datei den Body des Request. Bei dieser Datei kann es sich entweder um die derzeit im Hauptfenster aktive Datei (Option *Aktuelle Datei*) oder um eine externe Datei, zu der Sie mit "Durchsuchen" navigieren können, handeln.

© 2016-2022 Altova GmbH

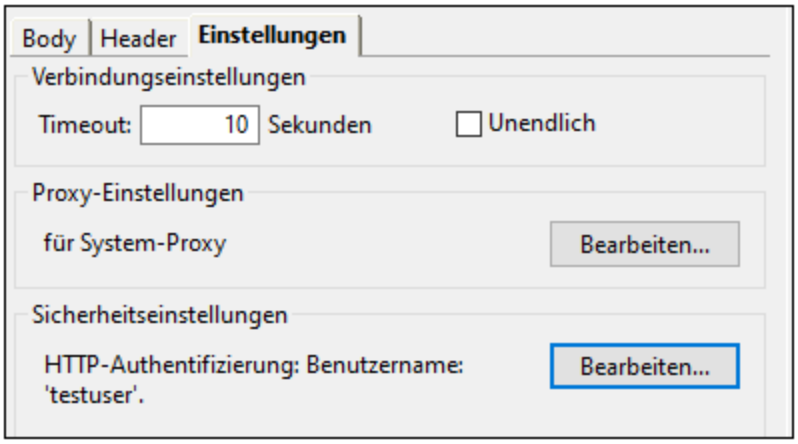


Um in den Datei-Modus zu wechseln, klicken Sie links oben auf dem Register *Body* auf das Symbol **Datei** (*siehe Abbildung oben*). Damit der Inhaltstyp des Body automatisch ermittelt werden kann, wählen Sie im Feld *Inhaltstyp* **`automatisch`** aus. Die automatische Ermittelung des Inhaltstyps erfolgt auf Basis der Dateierweiterung. Wenn Sie einen Inhaltstyp eingeben, wird der Request mit diesem Inhaltstyp gesendet; in diesem Fall müssen Sie sicherstellen, dass der Inhaltstyp korrekt ist.

Die Header und Einstellungen des Request können auf dieselbe Art und Weise wie bei anderen Requests definiert werden (d.h. auf den Registern *Header* bzw. *Einstellungen*).

Einstellungen für den HTTP-Request

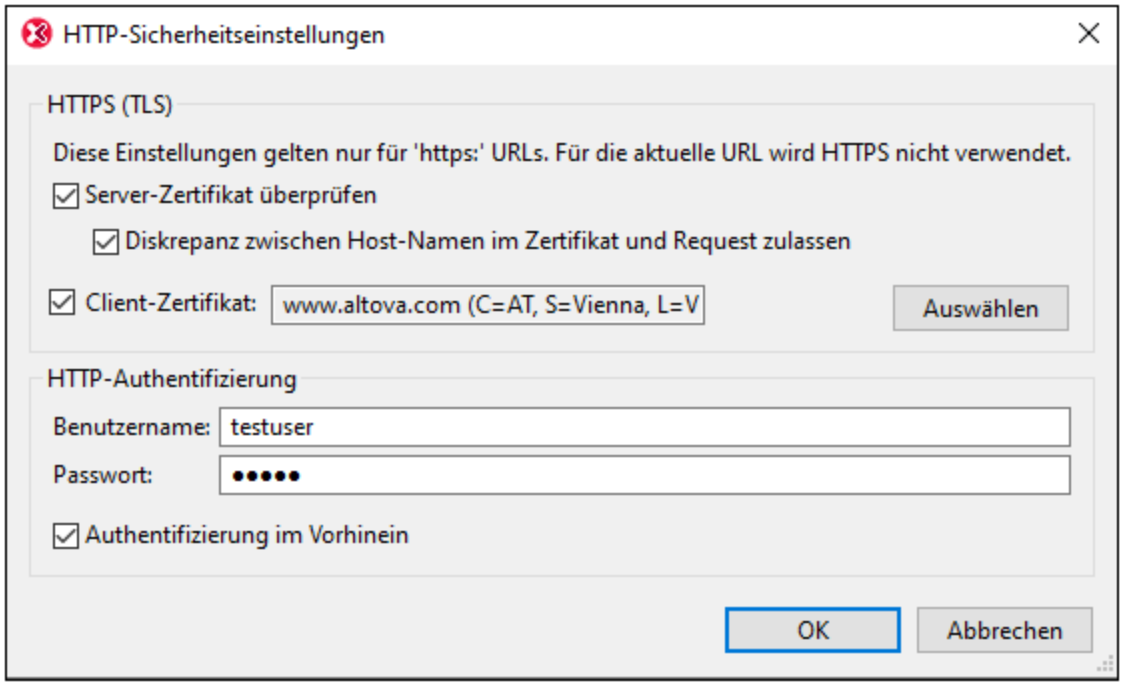
Auf dem Register *Einstellungen* des HTTP-Ausgabefensters (*Abbildung unten*) können Sie ein (i) Verbindungs-Timeout und (ii) die Sicherheitseinstellungen eines Request definieren. Beachten Sie, dass Sie für jedes der neun Register des Fensters separate Verbindungseinstellungen definieren können. In der Abbildung unten sehen Sie die Einstellungen für eine HTTPS URL.



Verbindungseinstellungen
Sie können die Zeitspanne in Sekunden, die XMLSpy versucht, eine Verbindung zum Webserver herzustellen, definieren. Nach Überschreitung dieser Zeitspanne erhalten Sie den Fehler **`I/O Error 28: Timeout was reached`**, wenn bis dahin keine Verbindung zustande gekommen ist. Wenn Sie kein Timeout definieren möchten, aktivieren Sie das Kontrollkästchen *Unendlich*.

Proxy-Einstellungen
Bietet eine Zusammenfassung der aktuellen Proxy-Einstellungen und eine Schaltfläche zum Öffnen des Abschnitts "Proxy-Einstellungen" des Dialgofelds "Öffnen"[1647].

Sicherheitseinstellungen
Klicken Sie auf **Bearbeiten**, um die Sicherheitseinstellungen eines Request zu bearbeiten. Daraufhin wird das Dialogfeld "HTTP-Sicherheitseinstellungen" (*Abbildung unten*) angezeigt. Hier können Sie HTTPS-Sicherheitseinstellungen definieren und die HTTP-Authentifizierungsinformationen für über dieses Register erfolgte Requests einstellen. Wenn auf dem Ziel-Webserver des Request SSL nicht verwendet wird, werden nur die HTTP-Authentifizierungsinformationen verwendet. Wenn auf dem Ziel-Webserver SSL verwendet wird, werden sowohl die HTTPS-Sicherheitseinstellungen als auch die HTTP-Authentifizierungsinformationen verwendet.



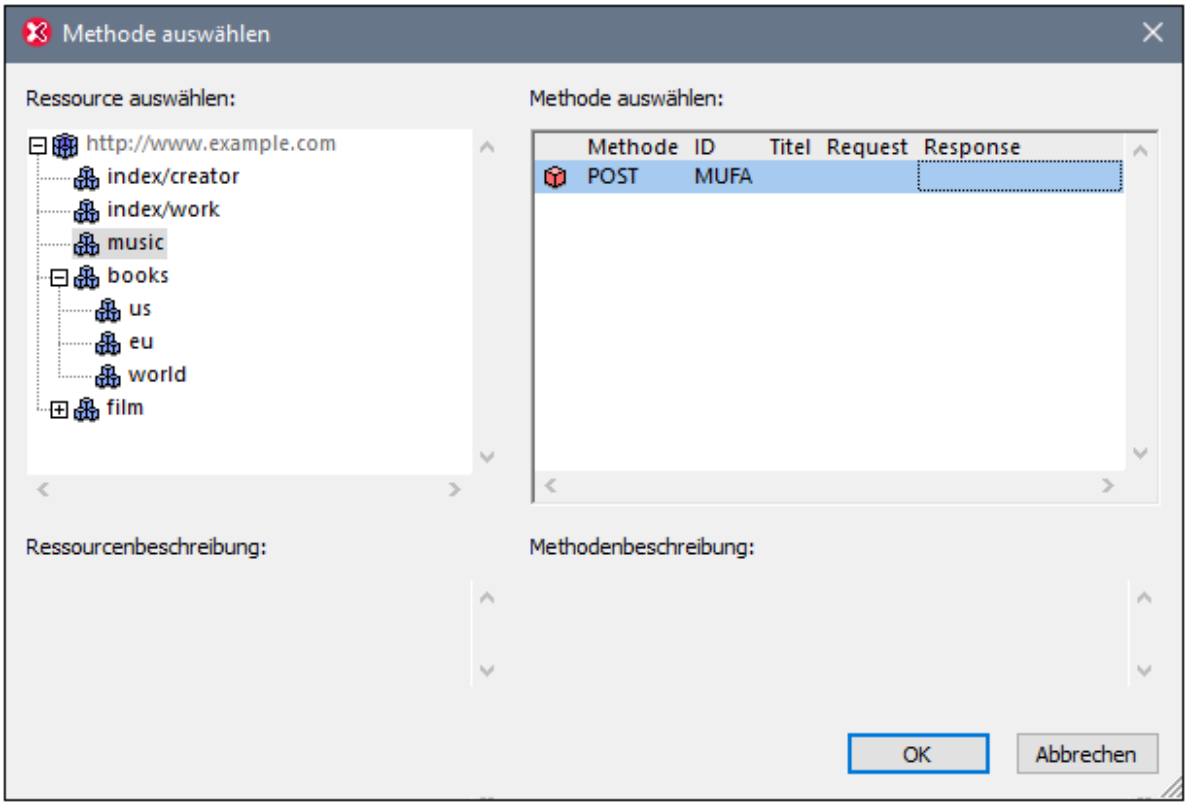
- *HTTPS-Sicherheitseinstellungen* Standardmäßig ist die Option *Server-Zertifikat überprüfen* aktiviert und Sie können angeben, ob der Host-Name im Request ein anderer als der Host-Name im Zertifikat sein darf. Wenn das Ziel eine Intranet-URL (z.B. in Ihrem Firmennetzwerk) ist, können Sie ein Zertifikat auf dem Intranet-Server mit Hilfe eines Client-Zertifikats (das sich normalerweise in Ihrem lokalen Zertifikatspeicher befindet) verifizieren.
- *HTTP-Authentifizierung:* Für einige Requests an einen Server ist eventuell eine Benutzerauthentifizierung erforderlich. Für diese Fälle können Sie hier einen Benutzernamen und ein Passwort eingeben. Wenn der Server nun eine Authentifizierung verlangt, wird diese automatisch gesendet. Andernfalls werden Sie eventuell nach Herstellung der Verbindung zum Server danach gefragt. Wenn der Anfangs-Request an den Server die Authentifizierungsinformationen enthält, wird dieser Vorgang als Authentifizierung im Vorhinein bezeichnet. Falls dies vom Server verlangt wird, aktivieren Sie die Option *Authentifzierung im Vorhinein*.

© 2016-2022 Altova GmbH

16.2 Importieren eines Requests zu senden

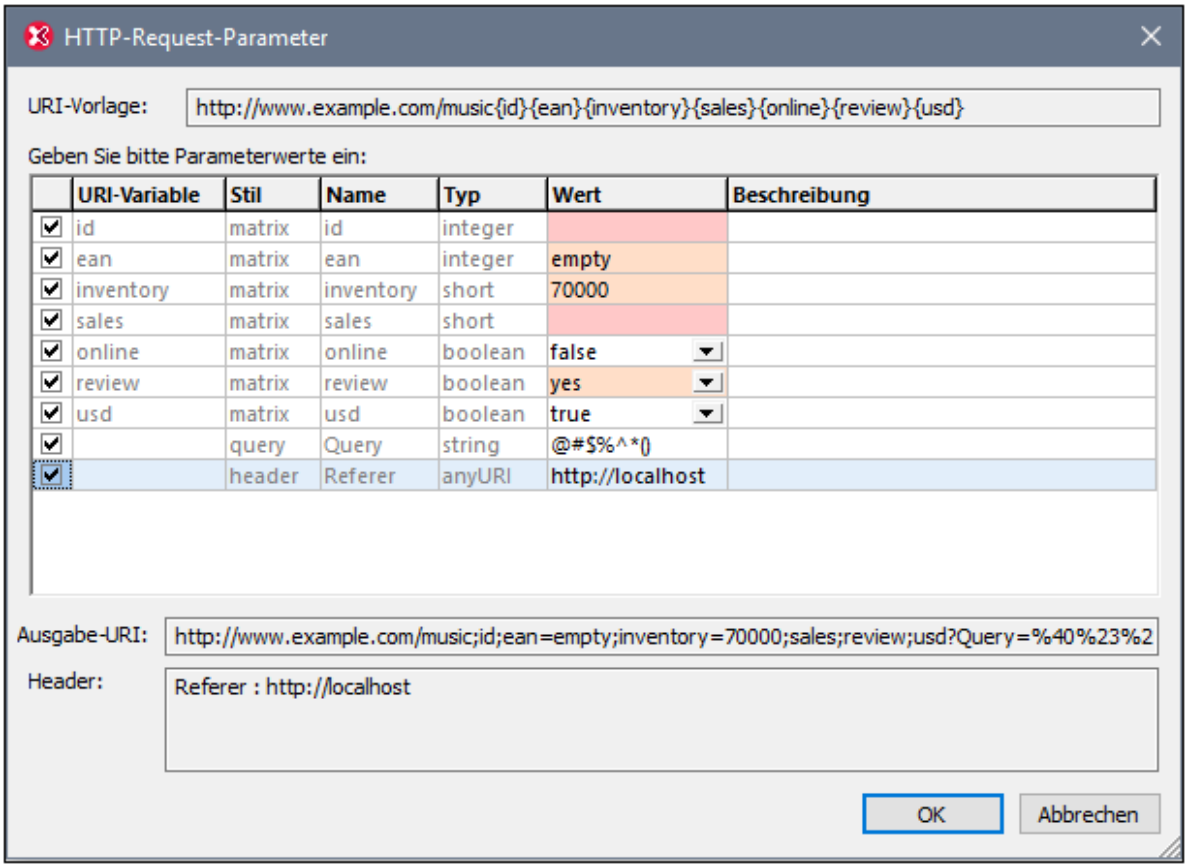
Sie können im HTTP-Ausgabefenster[817] einen Request aus einer WSDL 1.1-, WSDL 2.0- oder Web Application Development Language (WADL)-Datei importieren und diesen anschließend senden. Dies geschieht über den WSDL/WADL-Importassistenten von XMLSpy. Der Assistent öffnet eine WSDL- oder WADL-Datei, wählt einen Request aus einem der WSDL-Endpoints oder einer der WADL-Ressourcen der Datei aus, ermöglicht die Bearbeitung der editierbaren Parameter des Request und importiert diesen Request in das HTTP-Ausgabefenster[817]. Gehen Sie dazu folgendermaßen vor:

1. Klicken Sie im HTTP-Ausgabefenster[817] auf die Schaltfläche **Importieren**, um den WSDL/WADL-Importassistenten zu starten.
2. Navigieren Sie im daraufhin angezeigten Dialogfeld zur Auswahl einer Datei zur WSDL- oder WADL-Datei, die den gewünschten Request enthält und klicken Sie auf **OK**. Daraufhin wird der WSDL/WADL-Importassistent gestartet (*Abbildung unten*).



3. Wählen Sie im linken Bereich (*siehe Abbildung oben*) den entsprechenden WSDL-Endpoint oder die entsprechende WADL-Ressource (diejenige, die den gewünschten Request enthält) aus.
4. Wählen Sie im rechten Bereich (*siehe Abbildung oben*) den gewünschten Request (die WSDL-Operation oder die WADL-Methode) aus. Beachten Sie, dass für den Import über: (i) HTTP der Import nur für SOAP- und HTTP-Erweiterungen zur Verfügung steht und (ii) dass in diesem Bereich (auf der rechten Seite) nur unterstützte Bindings (SOAP und HTTP) angezeigt werden.
5. Klicken Sie auf **OK**. Wenn der Request einen oder mehrere Parameter enthält, werden im nächsten Bildschirm des Assistenten (*Abbildung unten*) die Parameter des ausgewählten Request angezeigt. Andernfalls wird der Request in das HTTP-Ausgabefenster[817] importiert und der Assistent wird geschlossen, *siehe Punkt 7 weiter unten*. Parameter sind Bestandteile des Request. So kann z.B. in einem Such-Request ein Parameter der Suchbegriff sein. Der Assistent validiert den Wert eines

Parameters anhand seines Datentyps und zeigt einen von drei Zuständen an. Ein rosafarbener Hintergrund kennzeichnet einen ungültigen Wert; ein beiger Hintergrund markiert einen falschen Wert an, der jedoch in den Request eingetragen und gesendet wird; ein weißer Hintergrund zeigt einen gültigen Wert an.

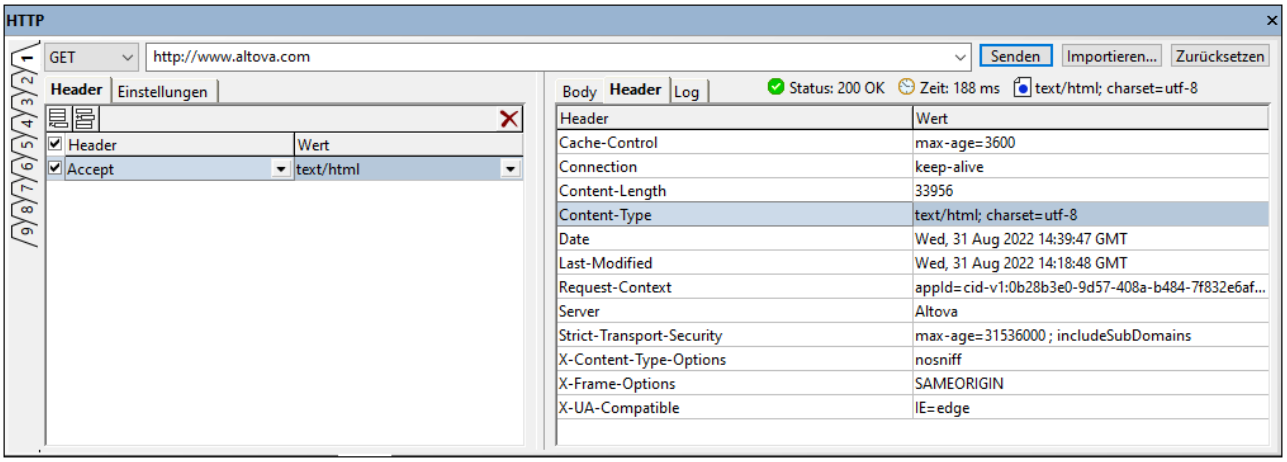


6. Geben Sie im Bildschirm "HTTP-Request-Parameter" (*Abbildung oben*) die gewünschten Parameterwerte in die Spalte *Wert* ein bzw. bearbeiten Sie diese. Beachten Sie, dass die Bearbeitung einiger Parameterwerte möglicherweise nicht zulässig ist; die Zellen solcher Werte sind für die Bearbeitung deaktiviert. Wenn Sie einen Parameter nicht verwenden möchten, deaktivieren Sie ihn durch Deaktivieren des *Aktivierungskontrollkästchens* in der ersten Spalte. Beachten Sie, dass einige Parameter obligatorisch sind, daher ist bei diesen das Aktivierungskontrollkästchen gesperrt und kann nicht deaktiviert werden. Während Sie die Parameterwerte bearbeiten, wird im Feld *Ausgabe-URI* der Request erstellt. Das Parameterraster enthält (am unteren Rand des Rasters) auch eine oder mehrere Zeilen für Header. Diese Header stammen aus der WADL-Datei; ihre Werte können im Raster bearbeitet werden, wenn dies gemäß den Definitionen in der WADL-Datei zulässig ist. Im Feld *Header* am unteren Rand des Fensters wird eine Zusammenfassung der Header aufgelistet.
7. Klicken Sie auf **OK**. Der Request wird in das HTTP-Ausgabefenster importiert[817] und dort auf folgende Weise angezeigt: (i) In der Methoden-Auswahlliste wird die HTTP-Methode des Request angezeigt; (ii) die URL wird auf Basis der Request-Parameter erstellt; (ii) die HTTP-Header des Request werden auf dem Register *Header* eingegeben. Wenn es sich beim Request um einen `POST`- oder `PUT`-Request handelt, wird der Rumpf des Request nicht auf dem *Body* eingegeben, sondern muss manuell hinzugefügt werden.
8. Überprüfen Sie das Register *Einstellungen*, um zu sehen, ob Sie die Einstellungen ändern müssen.
9. Klicken Sie auf **Senden**, um den Request zu senden.

© 2016-2022 Altova GmbH

16.3 Empfangen der Response

Die Antwort (Response) auf einen HTTP-Request wird im Bereich auf der rechten Seite des HTTP-Ausgabefensters angezeigt (*siehe Abbildung unten*).

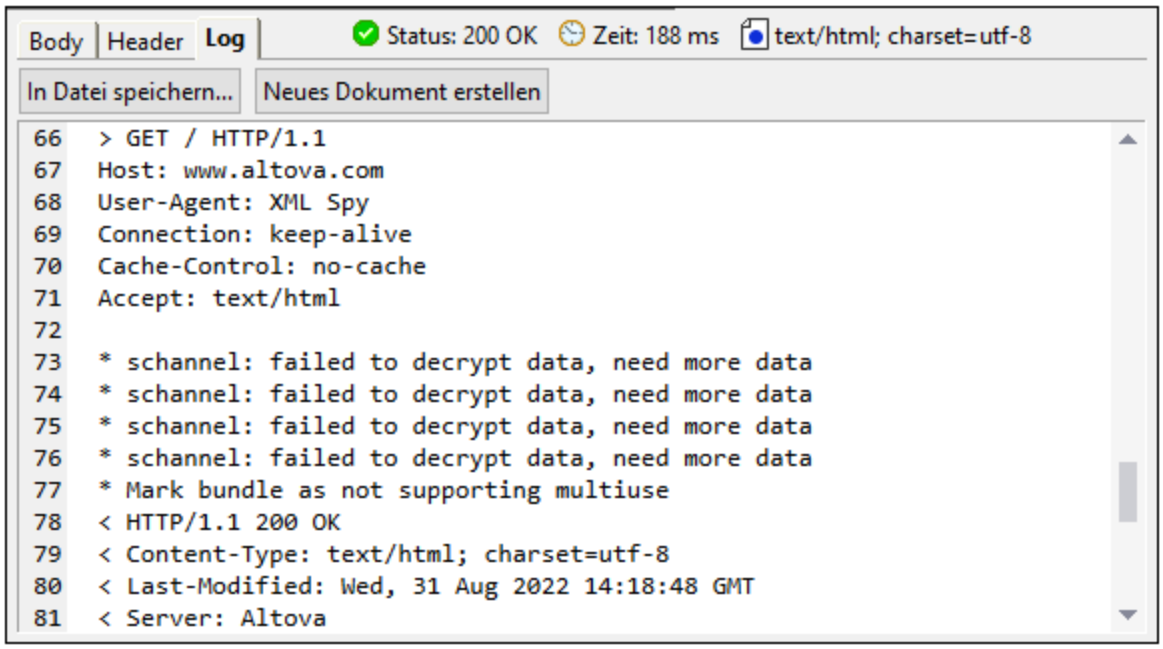


Rechts von den Registernamen *Body* und *Header* werden die folgenden Informationen über die Response aufgelistet:

- Der HTTP-Statuscode (Erklärung siehe unten 827)
- Die von der Herstellung der Verbindung bis zum Empfang des letzten Response-Teils verstrichene Zeit
- Der **`Content-Type`** (Inhaltstyp) der Response

Der Response-Bereich hat drei Register: *Body* (*Abbildung links unten*), *Header* (*Abbildung oben*) und Log (*Abbildung rechts unten*).





- *Das Register Body:* Der Body der Response wird bei HTML-, XML- oder JSON-Dokumenten mit Syntaxfärbung sowie mit Zeilennummerierung angezeigt. Das Register hat zwei Schaltflächen: (i) **In Datei speichern,** um den Body in einer Datei zu speichern und (ii) **Neues Dokument erstellen,** um in XMLSpy ein neues Dokument zu erstellen und das neu erstellte Dokument im Hauptfenster der Benutzeroberfläche anzuzeigen. Ein neu erstelltes Dokument kann auf die übliche Weise bearbeitet und gespeichert werden. Wenn anhand des Body der Response kein neues Dokument erstellt werden kann (z.B., wenn der Body ein Bild ist), ist die Schaltfläche **Neues Dokument erstellen** deaktiviert (*siehe Abbildung weiter unten*).
- *Das Register Header:* Enthält die Header der Response. Der `Content-Type` Header wird ebenso am oberen Rand des Bereichs angezeigt.

© 2016-2022 Altova GmbH

- *Das Register Log:* Ereignisse und Informationen im Zusammenhang mit dem Request werden auf dem Register "Log" angezeigt, welches eine Zeilennummerierung hat. Das Log kann in einer Datei gespeichert oder als neues Dokument im Hauptfenster erstellt werden. Ein neu erstelltes Dokument kann auf die übliche Weise bearbeitet und gespeichert werden.

HTTP-Statuscodes

Statuscodes werden in die folgenden Kategorien eingeteilt:

- 2XX-Codes werden für erfolgreiche Requests verwendet.
- 3XX-Codes werden für Redirects (Umleitungen) verwendet.
- 4XX-Codes werden verwendet, wenn es mit dem Request ein Problem gab.
- 5XX-Codes werden verwendet, wenn es mit dem Server ein Problem gab.

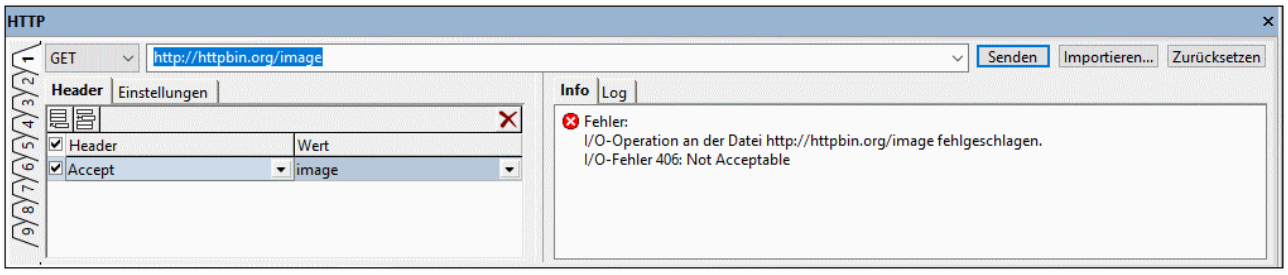
Einige häufig vorkommende Codes:

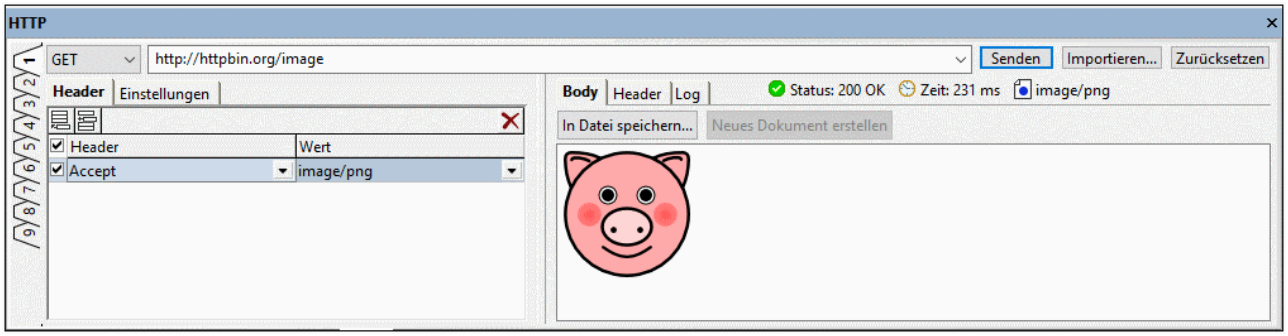
- *200 OK*: Response auf einen erfolgreichen Request wurde gesendet.
- *206 Partial Content*: Der Server sendet nur einen Teil der Ressource, da nur ein bestimmter Teil der Ressource angefordert wurde.
- *301 Moved Permanently*: Der Request sollte an die angegebene URL weitergeleitet werden.
- *401 Unauthorized*: Für die Ressource ist eine Authentifizierung erforderlich und diese ist entweder fehlgeschlagen oder wurde nicht angegeben.
- *403 Forbidden*: Gültiger Request, doch verweigert der Server die Aktion, eventuell, weil der Benutzer nicht die nötigen Anmeldeinformationen angegeben hat.
- *404 Not Found*: Die Ressource wurde nicht gefunden.
- *500 Internal Server Error*: Eine generische Fehlermeldung; wird gesendet, wenn keine genaueren Angaben verfügbar sind.

Siehe: Liste aller HTTP-Statuscodes.

Der Accept Header

Im Accept Header des Request ist der Inhaltstyp definiert, der in der Antwort akzeptiert werden muss. Im Folgenden sehen Sie z.B. den Unterschied zwischen der Response bei **`Accept=image`** (*erste Abbildung unten*) und der Response bei **`Accept=image/png`** (*zweite Abbildung unten*). Da im ersten Fall nicht angegeben ist, welches Bildformat in der Response gesendet werden soll, werden ein Fehlerstatuscode und eine JSON-Nachricht mit näheren Informationen gesendet.





© 2016-2022 Altova GmbH

17 XBRL

Die XBRL-Ansicht[266] von XMLSpy ist ein grafischer XBRL-Taxonomie-Editor, der eine grafische Übersicht über XBRL-Taxonomien sowie Funktionen zur intelligenten Taxonomiebearbeitung enthält. In diesem Abschnitt werden die verschiedenen Aspekte der XBRL-Ansicht beschrieben und erläutert, wie Sie Taxonomien in der XBRL-Ansicht bearbeiten.

Dieser Abschnitt ist in die folgenden Kapitel gegliedert:

- Taxonomie-Manager[831]: Hier wird beschrieben, wie Sie Taxonomien mit Hilfe des Taxonomie-Manager Tools für die Verwendung mit XMLSpy installieren, aktualisieren und verwalten.
- Grundlegende Arbeitsweise[847]: Hier wird erklärt, wie man Taxonomien mit den wichtigsten Komponenten erstellt.
- Zusätzliche Verfahren[873]: Hier werden zusätzliche Funktionalitäten wie z.B. das Arbeiten mit bevorzugten Labels und doppelt vorhandenen Facts beschrieben.
- XBRL Formula Editor[878]: Eine Anleitung zum Arbeiten mit XBRL Formulas in der XBRL-Ansicht
- XBRL-Tabellendefinitionseditor[904]: Hier wird die Tabellenstruktur[908] beschrieben, wie man Tabellen mit dem Editor definiert und wie die Tabellenlayoutvorschau[933] funktioniert. Weiters wird in diesem Abschnitt erklärt, wie man Tabellenparameter[929] verwendet und wie Tabellenparameter in Tabellengruppen verwendet werden.
- In XBRL suchen[947]: Eine Beschreibung der umfangreichen XBRL-spezifischen Suchfunktionen von XMLSpy.#
- OIM[953]: hier finden Sie eine Übersicht über OIM-Funktionalitäten in XMLSpy.
- Anmerkungen zur Validierung von XBRL-Instanzen und -Taxonomien[954]

Nähere Informationen zu diesem Thema finden Sie in den Abschnitten: Bearbeitungsansichten | XBRL-Ansicht[266] und unter der Beschreibung der Befehle im Menü XBRL[1533]. So finden Sie z.B. Informationen zur Generierung der Dokumentation für die Taxonomie (wie sie in der XBRL-Ansicht angezeigt wird) im Abschnitt Menübefehle | Menü "XBRL" | Dokumentation generieren[1541].

XML-Signaturen[417] für XBRL-Dateien in der XBRL-Ansicht können als externe Dateien erstellt werden. Eine Anleitung zur Arbeit mit Signaturen finden Sie im Abschnitt XML-Signaturen[417].

Unterstützung für US-GAAP und andere Taxonomien in XMLSpy

XMLSpy unterstützt die folgenden Taxonomien:

- US-GAAP 1.0, 2005, 2008, 2009, 2011 to 2022
- IFRS

Die neuesten Versionen von US-GAAP werden mit XMLSpy installiert. Über ein Taxonomie-Installationsprogramm, das Sie von der Altova Website kostenlos herunterladen können, stehen weitere Taxonomien, u.a. ältere US-GAAP-Taxonomien zur Installation zur Verfügung.

XBRL-Zertifizierung

XMLSpy wurde von XBRL International XBRL-zertifiziert. Nähere Informationen zur XBRL-Zertifizierung finden Sie unter XBRL Software Certification.

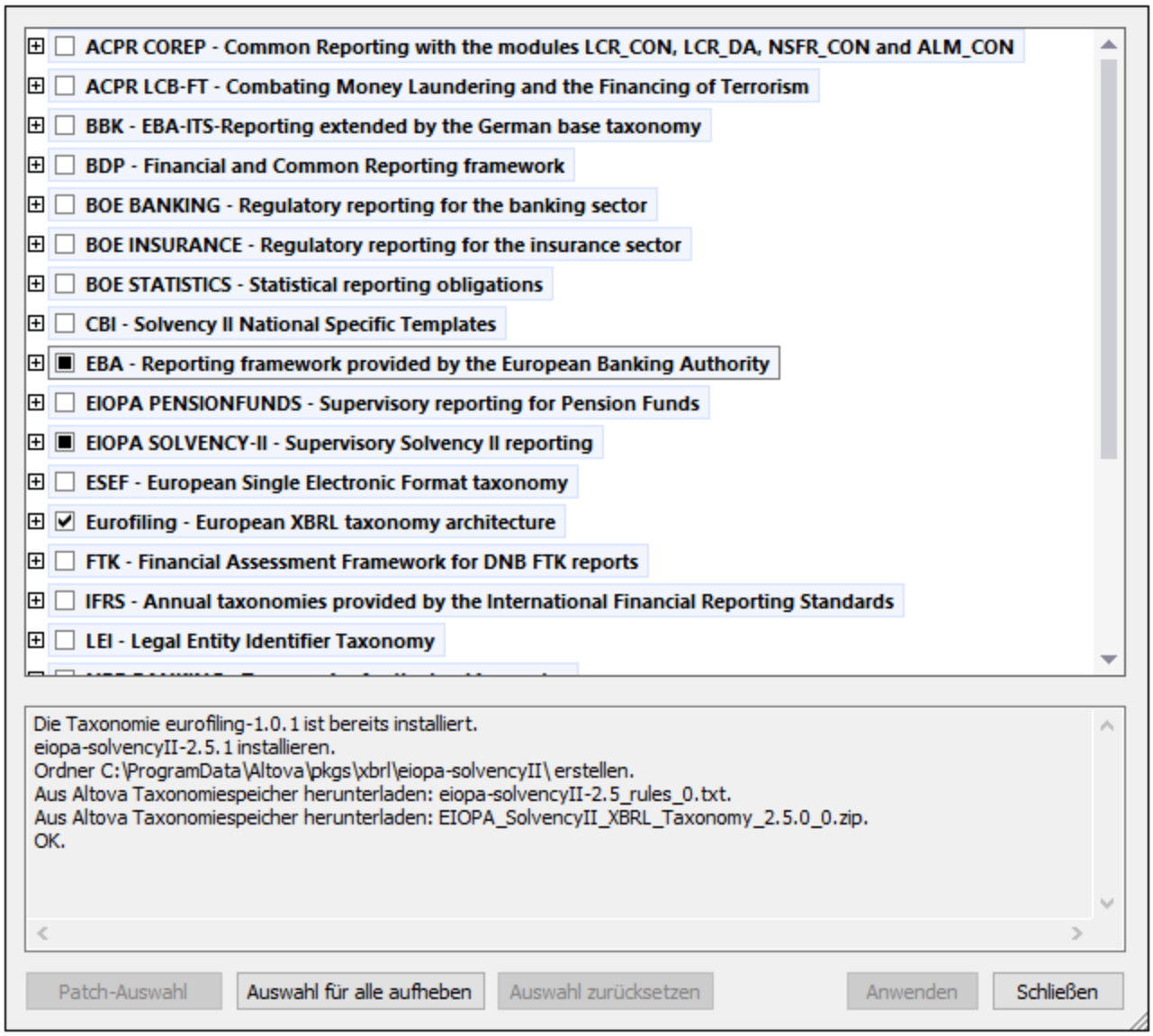
Altova Website: XBRL-Taxonomie-Editor, XBRL Validator

© 2016-2022 Altova GmbH

17.1 Taxonomie-Manager

Der XBRL-Taxonomie-Manager ist ein Altova-Tool, mit dem Sie XBRL-Taxonomien zentral installieren und verwalten können, um diese in allen XBRL-fähigen Applikationen von Altova einschließlich XMLSpy verwenden zu können.

- Unter Windows hat der Taxonomie-Manager eine grafische Benutzeroberfläche (*siehe Abbildung unten*) und steht auch über die Befehlszeile zur Verfügung. (Die Desktop-Applikationen von Altova stehen nur unter Windows zur Verfügung; *siehe Liste unten*).
- Unter Linux und MacOS steht der Taxonomie-Manager nur über die Befehlszeile zur Verfügung. (Die Server-Applikationen von Altova stehen unter Windows, Linux und macOS zur Verfügung; *siehe Liste unten*).



XBRL-fähige Applikationen von Altova

Desktop-Applikationen (nur Windows)	Server-Applikationen (Windows, Linux, macOS)
Altova XBRL Add-ins für Excel (EBA, ESEF, Solvency II)	MapForce Server (Standard und Advanced Edition)
MapForce Enterprise Edition	RaptorXML+XBRL Server
StyleVision Enterprise Edition	StyleVision Server
XMLSpy Enterprise Edition	

Installation und Deinstallation des Taxonomie-Managers

Der Taxonomie-Manager wird bei der ersten Installation einer neuen Version der Altova Mission Kit Enterprise Edition oder einer der XBRL-fähigen Applikationen von Altova (*siehe Tabelle oben*) automatisch installiert.

Ebenso wird er auch automatisch entfernt, wenn Sie die letzte XBRL-fähige Applikation von Altova auf Ihrem Rechner deinstallieren.

Taxonomie-Manager-Funktionalitäten

Im Taxonomie-Manager stehen die folgenden Funktionalitäten zur Verfügung:

- Anzeigen der auf Ihrem Rechner installierten XBRL-Taxonomien und Überprüfung, ob neue Versionen zum Download zur Verfügung stehen.
- Download neuer Versionen von XBRL-Taxonomien unabhängig vom Altova Produkt-Release-Zyklus. (Die Taxonomien werden von Altova online bereitgestellt und können über den Taxonomie-Manager heruntergeladen werden).
- Installation oder Deinstallation jeder beliebigen (oder ggf. aller) der zahlreichen Versionen einer bestimmten Taxonomie.
- Eine XBRL-Taxonomie kann Abhängigkeiten von anderen Taxonomien aufweisen. Bei der Installation oder Deinstallation einer bestimmten Taxonomie informiert Sie der Taxonomie-Manager über davon abhängige Taxonomien und installiert bzw. entfernt diese ebenfalls automatisch.
- Der Taxonomie-Manager ordnet Schema-Referenzen mit Hilfe des XML-Katalogs lokalen Dateien zu. Dadurch lassen sich große XBRL-Taxonomien schneller verarbeiten, als wenn sie sich unter einem entfernten Pfad befinden.
- Alle wichtigen Taxonomien werden über den Taxonomie-Manager bereitgestellt und regelmäßig auf die jeweils neuesten Version aktualisiert. Dadurch können alle Ihre Taxonomien zentral verwaltet werden und stehen allen XBRL-fähigen Applikationen von Altova jederzeit zur Verfügung.
- Im Taxonomie-Manager vorgenommene Änderungen werden für alle auf dem Rechner installierten Altova-Produkte wirksam.

Benutzerdefinierte XBRL-Taxonomien

Wenn Sie mit benutzerdefinierten XBRL-Taxonomien, die nicht Teil des Taxonomie-Manager sind, arbeiten müssen, können Sie diese folgendermaßen zum Taxonomie-Manager hinzufügen, um diese verwalten zu können:

- *In Altova-Desktop-Applikationen:* Wählen Sie den Menübefehl **Extras | Optionen** und gehen Sie anschließend zum Abschnitt *XBRL* | *Taxonomiepakete*. Navigieren Sie zum ZIP-Paket Ihrer

© 2016-2022 Altova GmbH

benutzerdefinierten XBRL-Taxonomie. Nähere Informationen dazu finden Sie in Ihrer Desktop-Produktdokumentation unter der Beschreibung dieses Befehls.
- *In Altova-Server-Applikationen:* Geben Sie bei der Befehlszeilenausführung von Befehlen, die benutzerdefinierte Taxonomien unterstützen, die Option **--taxonomy-package** oder **--taxonomy-package-config-file** ein. Diese Optionen werden z.B. in RaptorXML+XBRL Server von XBRL-Validierungsbefehlen wie valxbrl oder valxbrltaxonomy oder in MapForce vom Befehl run unterstützt.

Funktionsweise

Alle in Altova-Produkten verwendeten XBRL-Taxonomien werden von Altova online bereitgestellt. Dieser Speicher wird bei Veröffentlichung neuer Versionen der Taxonomien aktualisiert. Im Taxonomie-Manager werden sowohl bei Aufruf über die Benutzeroberfläche als auch über das CLI Informationen über die neuesten verfügbare Taxonomien angezeigt. Sie können die gewünschten Taxonomien dann über den Taxonomie-Manager installieren, aktualisieren oder deinstallieren.

Taxonomien können vom Taxonomie-Manager auch auf eine weitere Art installiert werden. Sie können eine Taxonomie und die davon abhängigen Taxonomien auf der Altova Website (https://www.altova.com/de/taxonomy-manager) auswählen. Daraufhin wird auf der Website eine Datei des Typs **.altova_taxonomies** mit Informationen über Ihre ausgewählte Taxonomie zum Download vorbereitet. Bei Doppelklick auf diese Datei oder bei Übergabe an den Taxonomie-Manager über das CLI als Argument des Befehls **install** 841 installiert der Taxonomie-Manager die ausgewählten Taxonomien.

Lokaler Cache: Überprüfung Ihrer Taxonomien

Alle Informationen über installierte Taxonomien werden in einem zentralen Cache-Verzeichnis auf Ihrem Rechner aufgezeichnet. Das Verzeichnis befindet sich hier:

Windows	`C:\ProgramData\Altova\pkgs\.cache`
Linux	`/var/opt/Altova/pkgs\.cache`
macOS	`/var/Altova/pkgs`

Dieses Cache-Verzeichnis wird regelmäßig mit dem neuesten Status der Taxonomien im Online-Speicher von Altova aktualisiert. Diese Aktualisierungen finden unter den folgenden Bedingungen statt:

- bei jedem Start von Taxonomie-Manager.
- Wenn Sie XMLSpy zum ersten Mal an einem bestimmten Kalendertag starten.
- Wenn XMLSpy länger als 24 Stunden geöffnet ist, findet alle 24 Stunden eine Aktualisierung des Cache statt.
- Sie können den Cache auch durch Ausführung des **update** 845-Befehls über die Befehlszeilenschnittstelle aktualisieren.

Der Taxonomie-Manager kann somit Ihre installierten Taxonomien über den Cache ständig anhand der online verfügbaren Taxonomien auf der Altova Website überprüfen.

Nehmen Sie keine manuellen Änderungen am Cache vor!

Das lokale Cache-Verzeichnis wird automatisch auf Basis der installierten oder deinstallierten Taxonomien verwaltet; es sollte nicht manuell geändert oder gelöscht werden. Falls Sie den Taxonomie-Manager je in seinen Originalzustand zurücksetzen möchten, (i) führen Sie den CLI-Befehl **reset** der Befehlszeilenschnittstelle und (ii) anschließend den Befehl initialize 841 aus. (Führen Sie alternativ dazu den Befehl reset 843 mit der Option **-i** aus).

17.1.1 Ausführen des Taxonomie-Managers

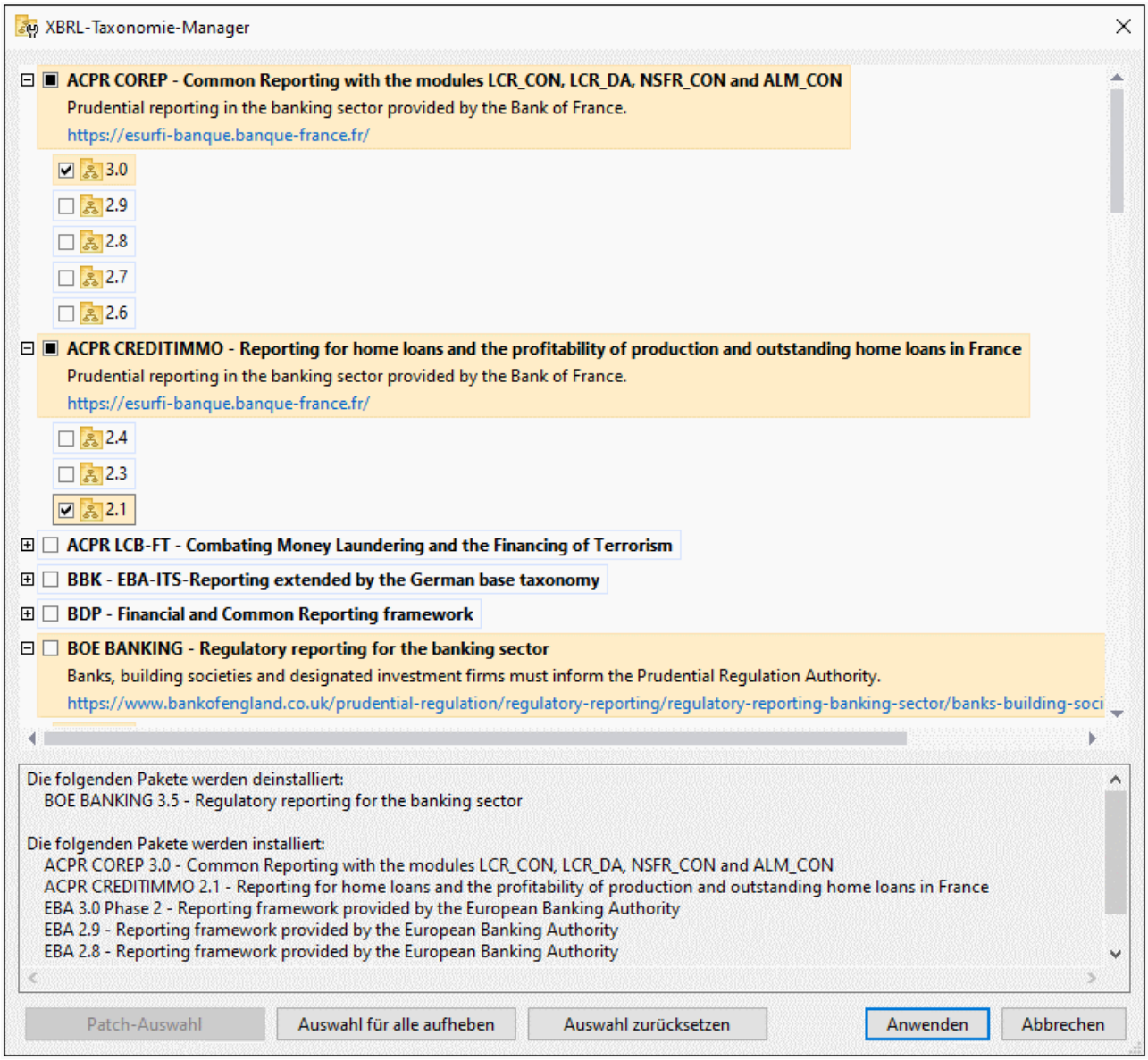
Grafische Benutzeroberfläche

Sie können die Benutzeroberfläche des Taxonomie-Managers auf eine der folgenden Arten aufrufen:

- *Bei der Installationen von XMLSpy:* Aktivieren Sie gegen Ende der Installation das Kontrollkästchen *Altova-Taxonomie-Manager aufrufen*, wodurch Sie die Benutzeroberfläche des XBRL-Taxonomie-Managers direkt aufrufen können. Auf diese Art können Sie Taxonomien während der Installation Ihrer Altova-Applikation installieren.
- *Nach der Installation von XMLSpy:* Nachdem die Applikation installiert wurde, können Sie die Benutzeroberfläche des Taxonomie-Managers jederzeit über den Menübefehl **Tools | XBRL-Taxonomie-Manager** aufrufen .
- über die vom XBRL-Taxonomie-Download-Center von Altova heruntergeladene `.altova_taxonomies`-Datei: Doppelklicken Sie auf die heruntergeladene Datei, um den Taxonomie-Manager zu starten, der daraufhin die (auf der Website) ausgewählten Taxonomien installiert.

Nachdem die Benutzeroberfläche des Taxonomie-Managers geöffnet wurde (*Abbildung unten*), werden bereits installierte Taxonomien markiert angezeigt. Wenn eine zusätzliche Taxonomie installiert werden soll, aktivieren Sie diese. Wenn eine bereits installierte Taxonomie deinstalliert werden soll, deaktivieren Sie diese. Nachdem Sie Ihre Auswahl getroffen haben, können Ihre Änderungen angewendet werden. Die Taxonomien, die installiert bzw. deinstalliert werden, werden markiert und im Fenster "Meldungen" am unteren Rand des Taxonomie-Manager-Fensters (*siehe Abbildung*) erscheint eine Meldung über die bevorstehenden Änderungen.

© 2016-2022 Altova GmbH



Befehlszeilenschnittstelle

Sie können den Taxonomie-Manager über eine Befehlszeilenschnittstelle starten, indem Sie Befehle an die ausführbare Datei **`taxonomymanager.exe`** senden.

Die Datei **`taxonomymanager.exe`** steht im folgenden Ordner zur Verfügung:

- *unter Windows:* `C:\ProgramData\Altova\SharedBetweenVersions`
- *Unter Linux oder macOS (nur Server-Applikationen):* `%INSTALLDIR%/bin`, wobei `%INSTALLDIR%` das Installationsverzeichnis des Programms ist.

Anschließend können Sie jeden der im Abschnitt zur CLI-Befehlsreferenz aufgelisteten Befehle verwenden.

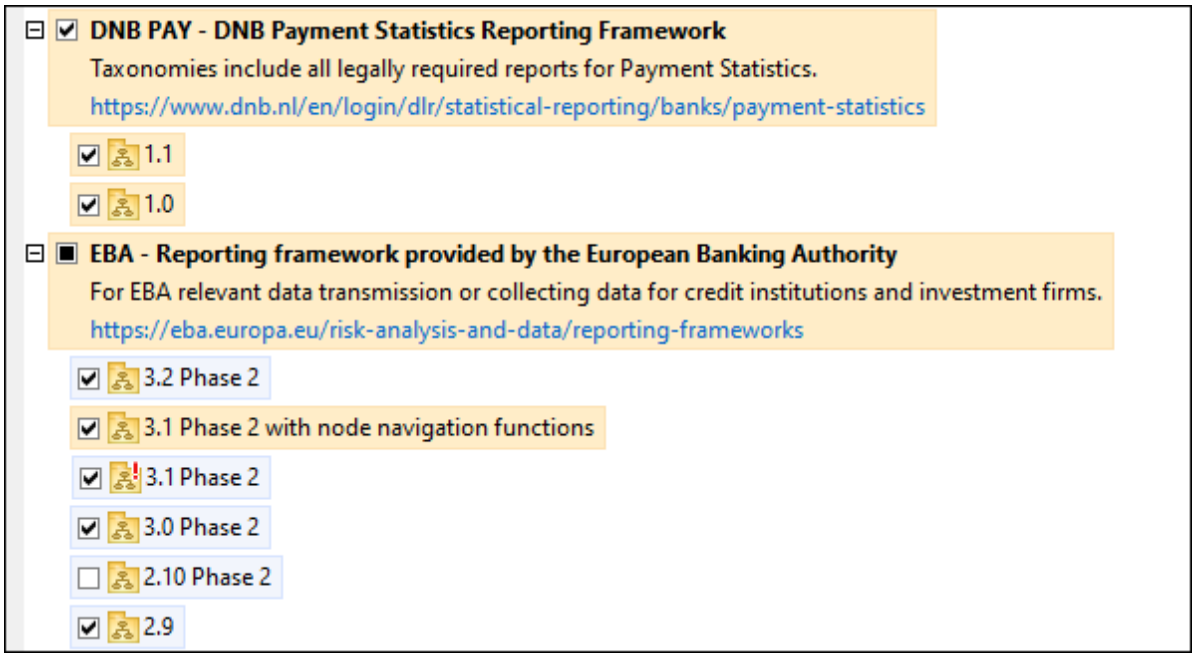
Um die Hilfe zu den Befehlen anzuzeigen, führen Sie den folgenden Befehl aus:

- *unter Windows:* `taxonomymanager.exe --help`
- *Unter Linux oder macOS (nur Server-Applikationen):* `sudo ./taxonomymanager --help`

17.1.2 Statuskategorien

Der Taxonomie-Manager unterscheidet folgendermaßen zwischen den von ihm verwalteten Taxonomien:

- *Installierte Taxonomien* Diese werden auf der Benutzeroberfläche mit einem Häkchen angezeigt (*in der Abbildung unten sind die mit einem Häkchen versehenen Versionen der DNB- und EBA-Taxonomie installiert*). Wenn alle Versionen einer Taxonomie ausgewählt sind, wird ein Häkchen angezeigt. Wenn zumindest eine Version nicht ausgewählt ist, wird ein gefülltes Quadrat angezeigt. Sie können die Auswahl für eine installierte Taxonomie aufheben, um sie zu **deinstallieren**.
- *Nicht installierte verfügbare Taxonomien.* Diese werden auf der Benutzeroberfläche mit einem deaktivierten Kontrollkästchen angezeigt. Sie können die Taxonomien, die **installiert** werden sollen, auswählen.



- *Taxonomien, für die ein Upgrade zur Verfügung steht* sind diejenigen, die seit ihrer Installation vom Herausgeber überarbeitet wurden. Sie werden auf der Benutzeroberfläche durch ein Symbol gekennzeichnet (*siehe Abbildung oben*). Sie können für eine installierte Taxonomie ein **Patch** der verfügbaren überarbeiteten Version installieren.

Wichtige Punkte

- In der Abbildung oben sind beide DNB-Taxonomien und einige der EBA-Taxonomien ausgewählt. Diejenigen mit einem blauen Hintergrund sind bereits installiert. Diejenige mit gelbem Hintergrund sind nicht installiert und wurden für die Installation ausgewählt. Beachten Sie, dass (i) die EBA 2.10 Phase

© 2016-2022 Altova GmbH

2-Taxonomie nicht installiert wurde und nicht für die Installation ausgewählt wurde, (ii) die EBA 3.1 Phase 2-Taxonomie installiert wurde, dafür aber seit ihrer Installation ein Patch vom Herausgeber zur Verfügung gestellt wurde, das noch nicht installiert wurde.

- Bei Ausführung des Taxonomie-Managers über die Befehlszeile wird der Befehl `list` [842] mit verschiedenen Optionen verwendet, um verschiedene Taxonomiekategorien aufzulisten:

`taxonomymanager.exe list`	Listet alle installierten und verfügbaren Taxonomien auf; auch verfügbare Upgrades werden angezeigt.
`taxonomymanager.exe list -i`	Listet nur installierte Taxonomien auf; auch verfügbare Upgrades werden angezeigt.
`taxonomymanager.exe list -u`	Listet für Taxonomien auf, für die Upgrades zur Verfügung stehen

Anmerkung: Verwenden Sie unter Linux und macOS, `sudo ./taxonomymanager list`

17.1.3 Anwenden eines Patch oder Installation einer Taxonomie

Anwenden eines Patch auf eine installierte Taxonomie

Von Zeit zu Zeit werden von den Herausgebern der XBRL-Taxonomien Patches (Upgrades oder Überarbeitungen) veröffentlicht. Wenn der Taxonomie-Manager erkennt, dass Patches zur Verfügung stehen, werden diese in der Taxonomieliste des Taxonomie-Managers angezeigt und Sie können diese Patches schnell installieren.

Über die Benutzeroberfläche

Patches werden mit dem Symbol gekennzeichnet. (*Siehe auch vorhergehendes Kapitel über Statuskategorien* [836]*).* Falls Patches zur Verfügung stehen, ist die Schaltfläche **Patch-Auswahl** aktiv. Klicken Sie darauf, um alle Patches für die Installation auszuwählen und vorzubereiten. Auf der Benutzeroberfläche ändert sich das Symbol von Taxonomien, für die ein Patch installiert wird von in , und im Fenster "Meldungen" am unteren Rand des Dialogfelds werden die Patches, die angewendet werden, aufgelistet. Sobald Sie mit der Auswahl fertig sind, klicken Sie auf **Anwenden**. Alle Patches werden gemeinsam angewendet. Beachten Sie, dass eine für die Installation eines Patch markierte Taxonomie deinstalliert wird, wenn Sie die Auswahl für diese Taxonomie aufheben.

Über das CLI

So wenden Sie einen Patch über die Befehlszeilenschnittstelle an:

1. Führen Sie den Befehl `list -u` [842] aus. Daraufhin werden alle Taxonomien, für die Patch Upgrades zur Verfügung stehen, aufgelistet.
2. Führen Sie den Befehl `upgrade` [845] aus, um alle Patches zu installieren.

Installieren einer verfügbaren Taxonomie

Sie können Taxonomien entweder über die Benutzeroberfläche des Taxonomie-Managers oder durch Senden der Taxonomie-Manager-Installationsbefehle über die Befehlszeile installieren.

Anmerkung: Wenn die aktuelle Taxonomie andere Taxonomien referenziert, werden auch die referenzierten Taxonomien installiert.

Über die Benutzeroberfläche
Um Taxonomien über die Benutzeroberfläche des Taxonomie-Managers zu installieren, wählen Sie die gewünschten Taxonomien aus und klicken Sie auf **Anwenden**.

Sie können die gewünschten Taxonomien auch auf der Altova Website auswählen und eine herunterladbare `.altova_taxonomies`-Datei generieren. Bei Doppelklick auf diese Datei wird der Taxonomie-Manager aufgerufen, in dem die gewünschten Taxonomien bereits vorausgewählt sind. Sie müssen nur mehr auf **Anwenden** klicken.

Über das CLI
Um Taxonomien über die Befehlszeile zu installieren, rufen Sie den Befehl `install` [841] auf:

```
taxonomymanager.exe install [options] Taxonomy+
```

wobei es sich bei `Taxonomy` um die gewünschte(n) Taxonomie(n) bzw. eine `.altova_taxonomies`-Datei handelt. Eine Taxonomie wird von einem Identifier im Format `<name>-<version>` referenziert. (Die Identifier von Taxonomien werden angezeigt, wenn Sie den Befehl `list` [842] ausführen.) Sie können beliebig viele Taxonomien eingeben. Nähere Informationen dazu finden Sie unter der Beschreibung des Befehls `install` [841].

Anmerkung: Verwenden Sie unter Linux oder macOS den Befehl `sudo ./taxonomymanager`.

Installation einer benötigten Taxonomie
Wenn Sie einen XBRL-Befehl in XMLSpy ausführen und XMLSpy erkennt, dass eine zur Ausführung des Befehls erforderliche Taxonomie nicht vorhanden oder unvollständig ist, wird der Taxonomie-Manager mit Informationen über die fehlende Taxonomie aufgerufen. Sie können die gewünschten Taxonomien dann über den Taxonomie-Manager direkt installieren.

Alle bereits installierten Taxonomien können jederzeit durch Aufruf des Taxonomie-Managers über **Extras | Taxonomie-Manager** über die Benutzeroberfläche des Taxonomie-Managers angezeigt werden.

17.1.4 Deinstallieren einer Taxonomie, Zurücksetzen

Deinstallieren einer Taxonomie

Sie können Taxonomien entweder über die Benutzeroberfläche des Taxonomie-Managers oder durch Senden der Taxonomie-Manager-Deinstallationsanweisungen über die Befehlszeile deinstallieren.

Anmerkung: Wenn die gewünschte Taxonomie andere Taxonomien referenziert, so werden auch die referenzierten Taxonomien deinstalliert.

Über die Benutzeroberfläche
Um Taxonomien über die Benutzeroberfläche des Taxonomie-Managers zu deinstallieren, deaktivieren Sie die Kontrollkästchen der entsprechenden Taxonomien und klicken Sie auf **Anwenden**. Daraufhin werden die ausgewählten Taxonomien und die davon referenzierten Taxonomien deinstalliert.

© 2016-2022 Altova GmbH

Um alle Taxonomien zu deinstallieren, klicken Sie auf **Auswahl für alle aufheben** und anschließend auf **Anwenden**.

Über das CLI
Um Taxonomien über die Befehlszeile zu deinstallieren, rufen Sie den Befehl **uninstall** auf:

```
taxonomymanager.exe uninstall [options] Taxonomy+
```

wobei es sich beim Argument **Taxonomy** eine zu deinstallierende Taxonomie oder eine **.altova_taxonomies**-Datei handelt. Eine Taxonomie wird von einem Identifier im Format **<name>-<version>** definiert. (Die Identifier von Taxonomien werden angezeigt, wenn Sie den Befehl **list** 842 ausführen.) Sie können beliebig viele Taxonomien eingeben. Nähere Informationen dazu finden Sie unter der Beschreibung des Befehls uninstall 844.

Anmerkung: Verwenden Sie unter Linux oder macOS den Befehl **sudo ./taxonomymanager**.

Zurücksetzen des Taxonomie-Managers

Sie können den Taxonomie-Manager zurücksetzen. Damit werden alle installierten Taxonomien und das Cache-Verzeichnis entfernt.

- Klicken Sie auf der Benutzeroberfläche auf **Auswahl zurücksetzen**.
- Führen Sie über die Benutzeroberfläche den Befehl reset 843 aus.

Nachdem Sie diesen Befehl ausgeführt haben, muss der Befehl initialize 841 ausgeführt werden, um das Cache-Verzeichnis neu zu erstellen. Führen Sie alternativ dazu den Befehl reset 843 mit der Option `-i` aus.

Beachten Sie, dass mit reset-i 843 die Originalinstallation des Produkts wiederhergestellt wird, daher wird empfohlen, nach dem Zurücksetzen auch den Befehl update 845 auszuführen. Führen Sie alternativ dazu den Befehl reset 843 mit den Optionen `-i` und `-u` aus.

17.1.5 Befehlszeilenschnittstelle (CLI)

Um den Taxonomie-Manager über die Befehlszeile aufzurufen, müssen Sie den Pfad zur ausführbaren Datei kennen. Standardmäßig befindet sich die ausführbare Taxonomie-Manager-Datei hier:

```
C:\ProgramData\Altova\SharedBetweenVersions\TaxonomyManager.exe
```

Anmerkung: Nachdem Sie auf Linux- und macOS-Systemen das Verzeichnis in dasjenige, das die ausführbare Datei enthält, geändert haben, können Sie die ausführbare Datei mit **sudo ./taxonomymanager** aufrufen. Das Präfix **./** gibt an, dass sich die ausführbare Datei im aktuellen Verzeichnis befindet. Das Präfix **sudo** gibt an, dass der Befehl mit Root-Rechten ausgeführt werden muss.

Befehlszeilensyntax

Die allgemeine Syntax zur Verwendung der Befehlszeile lautet folgendermaßen:

```
<exec> -h | --help | --version | <command> [options] [arguments]
```

Der senkrechte Balken `|` im Codefragment oben trennt eine Gruppe einander gegenseitig ausschließender Elemente. Optionale Elemente stehen innerhalb von eckigen Klammern `[]`. Im Prinzip können Sie den Pfad zur ausführbaren Datei, gefolgt von entweder `--h`, `--help` oder `--version`-Optionen oder gefolgt von einem Befehl eingeben. Jeder Befehl kann Optionen und Argumente haben. Die Liste der Befehle wird in den folgenden Abschnitten beschrieben.

17.1.5.1 help

Mit diesem Befehl erhalten Sie Hilfe zu Befehlen zur ausführbaren Taxonomie-Manager-Datei.

Syntax

```
<exec> help [Befehl]
```

`[Befehl]` ist hierbei ein optionales Argument zur Angabe jedes beliebigen gültigen Befehlsnamens.

Beachten Sie dazu Folgendes:

- Sie können die Hilfe zu einem Befehl auch durch Eingabe des Befehls, gefolgt von **`-h`** oder **`--help`** aufrufen, z.B: **`<exec> list -h`**
- Wenn Sie **`-h`** oder **`--help`** direkt nach dem Namen der ausführbaren Datei und vor einem Befehl eingeben, wird die allgemeine Hilfe (und nicht die Hilfe zu einem bestimmten Befehl) angezeigt, z.B: **`<exec> -h list`**

Beispiel

Mit dem folgenden Befehl wird Hilfe zum Befehl `list` angezeigt:

```
taxonomymanager help list
```

17.1.5.2 info

Mit diesem Befehl werden ausführliche Informationen über die einzelnen als `Taxonomy`-Argument angegebenen Taxonomien angezeigt. Darin enthalten sind Titel, Version, Beschreibung, Herausgeber der jeweils angegebenen Taxonomie und davon abhängige Taxonomien sowie die Information, ob die Taxonomie installiert ist oder nicht.

Syntax

```
<exec> info [options] Taxonomy+
```

- Das Argument **`Taxonomy`** ist der Name einer Taxonomie oder Teil eines Taxonomienamens. (Die Paket-ID einer Taxonomie und detaillierte Informationen über ihren Installationsstatus erhalten Sie mit dem Befehl `list`[842].)
- Mit **`<exec> info -h`** können Sie die Hilfe zum Befehl anzeigen.

© 2016-2022 Altova GmbH

Beispiel

Mit dem folgenden Befehl werden Informationen über die Taxonomien **eba-2.10** und **us-gaap-2020.0** angezeigt:

```
taxonomymanager info eba-2.1.0 us-gaap-2020.0
```

17.1.5.3 initialize

Mit diesem Befehl wird die Taxonomie-Manager-Umgebung initialisiert. Sie erstellen damit ein Cache-Verzeichnis, in dem Informationen über alle Taxonomien lokal gespeichert werden. Die Initialisierung erfolgt automatisch bei der ersten Installation einer XBRL-fähigen Altova-Applikation. Normalerweise muss dieser Befehl nicht ausgeführt werden. Nach Ausführung des **reset**-Befehls ist dies allerdings erforderlich.

Syntax

```
<exec> initialize | init [options]
```

Optionen

Für den Befehl **initialize** stehen die folgenden Optionen zur Verfügung:

`--silent, --s`	Nur Fehlermeldungen anzeigen. Der Standardwert ist **false**.
`--verbose, --v`	Anzeige detaillierter Informationen während der Ausführung. Der Standardwert ist **false**.
`--help, --h`	Anzeige der Hilfe zum Befehl.

Beispiel

Mit dem folgenden Befehl wird der Taxonomie-Manager initialisiert:

```
taxonomymanager initialize
```

17.1.5.4 install

Mit diesem Befehl installieren Sie eine oder mehrere Taxonomien.

Syntax

```
<exec> install [options] Taxonomy+
```

Um mehrere Taxonomien zu installieren, fügen Sie das Argument **Taxonomy** mehrmals hinzu.

Als **Taxonomy**-Argument kann eines der folgenden verwendet werden:

- Ein Taxonomie-Identifier (im Format **<name>-<version>**, z.B: **eba-2.10**). Um die Taxonomie-Identifier der gewünschten Taxonomien zu eruieren, führen Sie den Befehl `list` [842] aus. Sie können auch einen abgekürzten Identifier verwenden, sofern dieser eindeutig ist, z.B. **eba**. Falls Sie einen abgekürzten Identifier verwenden, wird die neueste Version dieser Taxonomie installiert.
- Der Pfad zu einer von der Altova-Website heruntergeladenen **.altova_taxonomies**-Datei. Informationen zu diesen Dateien finden Sie in *der Einführung zum Taxonomie-Manager.: Funktionsweise* [831].

Optionen
Für den Befehl **install** stehen die folgenden Optionen zur Verfügung:

`--silent, --s`	Nur Fehlermeldungen anzeigen. Der Standardwert ist **false**.
`--verbose, --v`	Anzeige detaillierter Informationen während der Ausführung. Der Standardwert ist **false**.
`--help, --h`	Anzeige der Hilfe zum Befehl.

Beispiel

Mit dem folgenden Befehl werden die neuesten **eba** (European Banking Authority) und **us-gaap** (US Generally Accepted Accounting Principles) Taxonomien installiert:

```
taxonomymanager install eba us-gaap
```

17.1.5.5 list

Mit diesem Befehl werden vom Taxonomie-Manager verwaltete Taxonomien aufgelistet. In der Liste wird eine der folgenden Informationen angezeigt:

- Alle verfügbaren Taxonomien
- Taxonomien, die im Namen den im Argument **Taxonomy** angegebenen String enthalten
- Nur installierte Taxonomien
- Nur Taxonomien, für die ein Upgrade installiert werden kann

Syntax

```
<exec> list | ls [options] Taxonomy?
```

Wenn kein **Taxonomy**-Argument angegeben wird, werden alle verfügbaren Taxonomien aufgelistet. Andernfalls werden die durch die angegebenen Optionen definierten Taxonomien aufgelistet (*siehe Beispiel unten*). Beachten Sie, dass Sie das Argument **Taxonomy** mehrfach angeben können.

Optionen
Für den Befehl **list** stehen die folgenden Optionen zur Verfügung:

`--installed, --i`	Auflisten nur der installierten Taxonomien. Der Standardwert ist **false**.
`--upgradeable, --u`	Auflisten nur derjenigen Taxonomien, für die Upgrades (Patches) zur Verfügung stehen. Der Standardwert ist **false**.

© 2016-2022 Altova GmbH

`--help, --h`	Anzeige der Hilfe zum Befehl.

Beispiele

- Um alle verfügbaren Taxonomien aufzulisten, führen Sie den folgenden Befehl aus: **`taxonomymanager list`**
- Um installierte Taxonomien aufzulisten, führen Sie **`taxonomymanager list -i`** aus.
- Um Taxonomien, die in ihrem Namen entweder "eba" oder "us-gaap" enthalten, aufzulisten, führen Sie **`taxonomymanager list eba us-gaap`** aus.

17.1.5.6 reset

Mit diesem Befehl werden alle installierten Taxonomien und das Cache-Verzeichnis entfernt. Ihre Taxonomieumgebung wird vollständig zurückgesetzt. Nachdem Sie diesen Befehl ausgeführt haben, muss der Befehl `initialize` 841 ausgeführt werden, um das Cache-Verzeichnis neu zu erstellen. Führen Sie alternativ dazu den Befehl `reset` mit der Option `-i` aus. Da mit `reset-i` die Originalinstallation des Produkts wiederhergestellt wird, wird empfohlen, nach dem Zurücksetzen und Initialisieren auch den Befehl `update` 845 auszuführen. Führen Sie alternativ dazu den Befehl reset mit den Optionen `-i` und `-u` aus.

Syntax

```
<exec> reset [Optionen]
```

Optionen
Für den Befehl **`reset`** stehen die folgenden Optionen zur Verfügung:

`--init, --i`	Initialisierung des Taxonomie-Managers nach dem Zurücksetzen. Der Standardwert ist **`false`**.
`--update, --u`	Aktualisiert die Liste der verfügbaren Taxonomien im Cache. Der Standardwert ist **`false`**.
`--silent, --s`	Nur Fehlermeldungen anzeigen. Der Standardwert ist **`false`**.
`--verbose, --v`	Anzeige detaillierter Informationen während der Ausführung. Der Standardwert ist **`false`**.
`--help, --h`	Anzeige der Hilfe zum Befehl.

Beispiele

- Um den Taxonomie-Manager zurückzusetzen, führen Sie den folgenden Befehl aus: **`taxonomymanager reset`**
- Um den Taxonomie-Manager zurückzusetzen und ihn zu initialisieren, führen Sie **`taxonomymanager reset -i`** aus.
- Um den Taxonomie-Manager zurückzusetzen, ihn zu initialisieren und seine Taxonomieliste zu aktualisieren, führen Sie **`taxonomymanager reset -i -u`** aus.

17.1.5.7 uninstall

Mit diesem Befehl deinstallieren Sie eine oder mehrere Taxonomien. Standardmäßig werden auch alle Taxonomien, die von der aktuellen Taxonomie referenziert werden, deinstalliert. Um nur die aktuelle Taxonomie zu deinstallieren und die referenzierten Taxonomien beizubehalten, setzen Sie die Option `--k`.

Syntax

```
<exec> uninstall [options] Taxonomy+
```

Um mehrere Taxonomien zu deinstallieren, fügen Sie das Argument **`Taxonomy`** mehrmals hinzu.

Als **`Taxonomy`**-Argument kann eines der folgenden verwendet werden:

- Ein Taxonomie-Identifier (im Format **`<name>-<version>`**, z.B: **`eba-2.10`**). Um die Taxonomie-Identifier der installierten Taxonomien zu eruieren, führen Sie den Befehl `list -i`[842] aus. Sie können auch einen abgekürzten Taxonomienamen verwenden, sofern dieser eindeutig ist, z.B. **`eba`**. Falls Sie einen abgekürzten Namen verwenden, werden alle Taxonomien, die die Abkürzung in ihrem Namen enthalten, deinstalliert.
- Der Pfad zu einer von der Altova-Website heruntergeladenen **`.altova_taxonomies`**-Datei. Informationen zu diesen Dateien finden Sie *der Einführung zum Taxonomie-Manager.: Funktionsweise*[831].

Optionen
Für den Befehl **`uninstall`** stehen die folgenden Optionen zur Verfügung:

`--keep-references, --k`	Definieren Sie diese Option, um referenzierte Taxonomien beizubehalten. Der Standardwert ist **`false`**.
`--silent, --s`	Nur Fehlermeldungen anzeigen. Der Standardwert ist **`false`**.
`--verbose, --v`	Anzeige detaillierter Informationen während der Ausführung. Der Standardwert ist **`false`**.
`--help, --h`	Anzeige der Hilfe zum Befehl.

Beispiel

Mit dem folgenden Befehl werden die Taxonomien **`eba-2.10`** und **`us-gaap-2020.0`** und deren Abhängigkeiten deinstalliert:

```
taxonomymanager uninstall eba-2.10 us-gaap-2020.0
```

Mit dem folgenden Befehl wird die **`eba-2.10`**-Taxonomie, nicht aber die davon referenzierten Taxonomien deinstalliert:

```
taxonomymanager uninstall --k eba-2.10
```


© 2016-2022 Altova GmbH

17.1.5.8 update

Mit diesem Befehl wird die Liste der über den Online-Speicher verfügbaren Taxonomien abgefragt und das lokale Cache-Verzeichnis wird aktualisiert. Normalerweise muss dieser Befehl nur ausgeführt werden, wenn Sie reset[843] und initialize[841] ausgeführt haben.

Syntax

```
<exec> update [options]
```

Optionen
Für den Befehl **update** stehen die folgenden Optionen zur Verfügung:

`--silent, --s`	Nur Fehlermeldungen anzeigen. Der Standardwert ist **false**.
`--verbose, --v`	Anzeige detaillierter Informationen während der Ausführung. Der Standardwert ist **false**.
`--help, --h`	Anzeige der Hilfe zum Befehl.

Beispiel

Mit dem folgenden Befehl wird der lokale Cache mit der Liste der neuesten Taxonomien aktualisiert:

```
taxonomymanager update
```

17.1.5.9 upgrade

Mit diesem Befehl werden alle installierten Taxonomien, für die ein Upgrade installiert werden kann, auf die neueste verfügbare *Patch*-Version aktualisiert. Um herauszufinden, welche Taxonomien aktualisiert werden können, starten Sie den Befehl `list -u`[842] .

Anmerkung: Der Befehl **upgrade** würde eine veraltete Taxonomie entfernen, falls keine neuere Version zur Verfügung steht.

Syntax

```
<exec> upgrade [Optionen]
```

Optionen
Für den Befehl **upgrade** stehen die folgenden Optionen zur Verfügung:

`--silent, --s`	Nur Fehlermeldungen anzeigen. Der Standardwert ist **false**.
`--verbose, --v`	Anzeige detaillierter Informationen während der Ausführung. Der Standardwert ist **false**.
`--help, --h`	Anzeige der Hilfe zum Befehl.

© 2016-2022 Altova GmbH

17.2 Grundlegende Arbeitsweise

In diesem Abschnitt wird beschrieben, wie Sie Taxonomien erstellen, die die wichtigsten Komponenten enthalten. Er enthält die folgenden Kapitel:

- Er beginnt mit einem kurzen Überblick über den Unterschied zwischen neuen und bestehenden Taxonomien[847] und der Bedeutung dieses Unterschieds. Anschließend finden Sie eine Beschreibung der Dateien, aus der die XBRL-Taxonomie besteht[848], und wie diese in der XBRL-Ansicht angezeigt werden.
- Beginnend mit dem Abschnitt Erstellen einer neuen Taxonomie[851] beschreiben wir Schritt für Schritt, wie man eine Taxonomie in der XBRL-Ansicht erstellt. Am Ende der einzelnen Abschnitte finden Sie Anleitungen, wie Sie die Informationen aus diesem Abschnitt in die Praxis umsetzen oder testen. Es wird darin die Taxonomie verwendet, die mit Hilfe der Anleitungen in den vorhergehenden Abschnitten bis zu diesem Punkt erstellt wurde.

17.2.1 Taxonomien: neue und bestehende

In der XBRL-Ansicht von XMLSpy können Sie bestehende Taxonomien bearbeiten und neue erstellen.

- *Bestehende Taxonomien:* Es gibt zwei Arten von vorhandenen Taxonomien: (i) Standardtaxonomien, die nicht bearbeitet werden sollten; und (ii) Nicht-Standardtaxonomien, die bearbeitet werden können; diese Taxonomien wurden möglicherweise von Ihnen oder einem anderen Mitarbeiter erstellt.
- *Neue Taxonomien:* Neue Taxonomien können in XMLSpy erstellt werden. Es gibt zwei Arten von neuen Taxonomien: (i) Taxonomien, die von Grund auf neu erstellt werden und (ii) Taxonomien, die eine Standardtaxonomie erweitern.

Beide Arten von Taxonomien können in der XBRL-Ansicht angezeigt und bearbeitet werden. In einigen Fällen, z.B., wenn eine Standardtaxonomie in eine Taxonomie importiert wird, die Sie erstellen (um die importierte Taxonomie zu erweitern), können Sie die importierte Taxonomie nicht bearbeiten. Elemente aus importierten Taxonomien, die nicht bearbeitet werden können (und dürfen), werden grau angezeigt.

Taxonomiepakete

Ein XBRL-Taxonomiepaket ist ein gezipptes Archiv, das eine Offline-Kopie einer Taxonomie enthält. Das Taxonomiepaket enthält eine XML-Katalogdatei, die URIs auf die Dateipfade der Offline-Taxonomie mappt, so dass die Taxonomie Applikationen offline zur Verfügung steht. Die Regeln zur Strukturierung und dem Aufbau von Taxonomiepaketen finden Sie in der Taxonomy Packages Recommendation von XBRL.org.

Wenn Sie ein Taxonomiepaket herunterladen, können Sie dieses auf XMLSpy registrieren, damit XMLSpy die Offline-Ressourcen des Pakets (wie z.B. Schemas) bei der Validierung verwenden kann. Die Registrierung des Pakets erfolgt über das Fenster **Tools | Optionen | Taxonomiepakete**[1641]**.** Eine Anleitung dazu finden Sie in der Beschreibung dieses Fensters[1641].

Schritte beim Erstellen einer neuen Taxonomie

Eine neue Taxonomie baut normalerweise auf einer bestehenden auf. In der neuen Taxonomie werden neue Elemente hinzugefügt und es werden Beziehungen zwischen diesen neuen Elementen und zwischen neuen Elementen und importierten Elementen erstellt. Im Folgenden sind die allgemeinen Anforderungen für eine neue Taxonomie und die Vorgehensweise beim Erstellen einer Taxonomie aufgelistet:

1. Die neue Taxonomie muss in ihrem eigenen Namespace erstellt werden, um sie von anderen Taxonomien zu unterscheiden. Wenn die neue Taxonomie eine bestehende erweitern soll, muss die bestehende in die neue Taxonomie importiert werden.
2. Neue Concepts (Elemente) werden in der neuen Taxonomie definiert.
3. Es werden Beziehungsdateien (oder Linkbases) erstellt, die die Definition-, Presentation-, Calculation-Label- und Reference-Beziehungen der neuen Taxonomie enthalten.
4. Beziehungen für die neue Taxonomie müssen von Grund auf neu erstellt werden.

In der Beschreibung oben haben wir mit dem Begriff *Taxonomie* die gesamte Taxonomie bestehend aus mehreren Dateien, den Concept Definitions-Dateien und den Beziehungsdateien, bezeichnet: (Eine Beschreibung der verschiedenen Dateien, aus denen eine Taxonomie besteht, finden Sie im Abschnitt Taxonomiedateien - Übersicht 848.)

Verwendung der XBRL-Ansicht

In den folgenden Abschnitten wird beschrieben, wie Sie die Funktionen der XBRL-Ansicht verwenden, um Taxonomien zu erstellen und zu bearbeiten.
Ab dem Abschnitt Erstellen einer neuen Taxonomie 851 finden Sie am Ende der einzelnen Abschnitte Anleitungen zur Erstellung Ihrer eigenen Taxonomie. Die Anleitungen bauen jeweils auf dem vorhergehenden Kapitel auf. Wenn Sie den Abschnitt Erstellen von Beziehungen: Teil 1 866 erreicht haben, ist Ihnen die XBRL-Ansicht vertraut und Sie können damit arbeiten.

Die Taxonomie, die sie erstellen, ist nach einigen Ergänzungen identisch mit der mitgelieferten Beispieldatei (`Nanonull.xsd`) aus dem Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\XBRLExamples\Nanonull`. (Beachten Sie, dass die Taxonomie-Hauptdatei immer die Erweiterung `.xsd` hat. Die Dateierweiterung `.xbrl` wird für XBRL-Instanzdateien und nicht für Taxonomiedateien verwendet.)

17.2.2 Taxonomiedateien Übersicht

In einer gut konzipierten XBRL-Taxonomie sind Taxonomie-Concepts in einer separaten Datei gespeichert, d.h. separat von den Taxonomiebeziehungen. Diese Datei wird als Taxonomie-Hauptdatei bezeichnet oder auch als die Concepts-Definitionsdatei. Da es verschiedene Arten von Beziehungen gibt, werden Beziehungen in separaten Dateien für die einzelnen Beziehungsarten gespeichert. In der nachfolgenden Tabelle sind die verschiedenen Arten von Dateien aufgelistet, aus denen ein Taxonomiedokument normalerweise besteht.

XBRL-Datei	Beschreibung	Dateityp
Concepts	Die einzelnen Concepts sind in einem XML-Schema-Element (`element`) definiert.	XML-Schema-Datei (`.xsd`) Concept-Definitionsdatei
Definition Relationships	Ein `definitionLink` Element enthält alle Locators und Definition Arcs für Concept-Beziehungen.	XML-Datei (`.xml`)
Calculation Relationships	Ein `calculationLink` Element enthält alle Locators und Calculation Arcs.	XML-Datei (`.xml`)

© 2016-2022 Altova GmbH

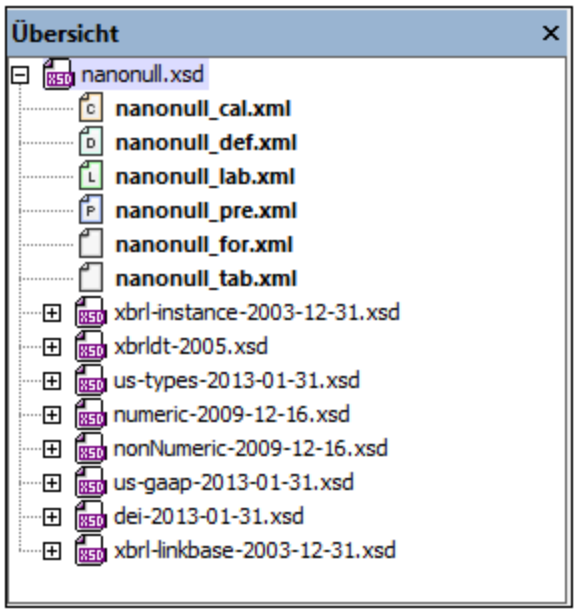
Presentation Relationships	Ein `presentationLink` Element enthält alle Locators und Presentation Arcs.	XML-Datei (`.xml`)
Labels	Ein `labelLink` Element enthält alle Locators, Label Arcs und Labels.	XML-Datei (`.xml`)
References	Ein `referenceLink` Element enthält alle Locators, Reference Arcs und Reference Ressourcen.	XML-Datei (`.xml`)

Die Pfade der Beziehungsdateien sind in der Concept-Definitionsdatei (der `.xsd`-Datei) in einem `/schema/annotation/appinfo` Element definiert:

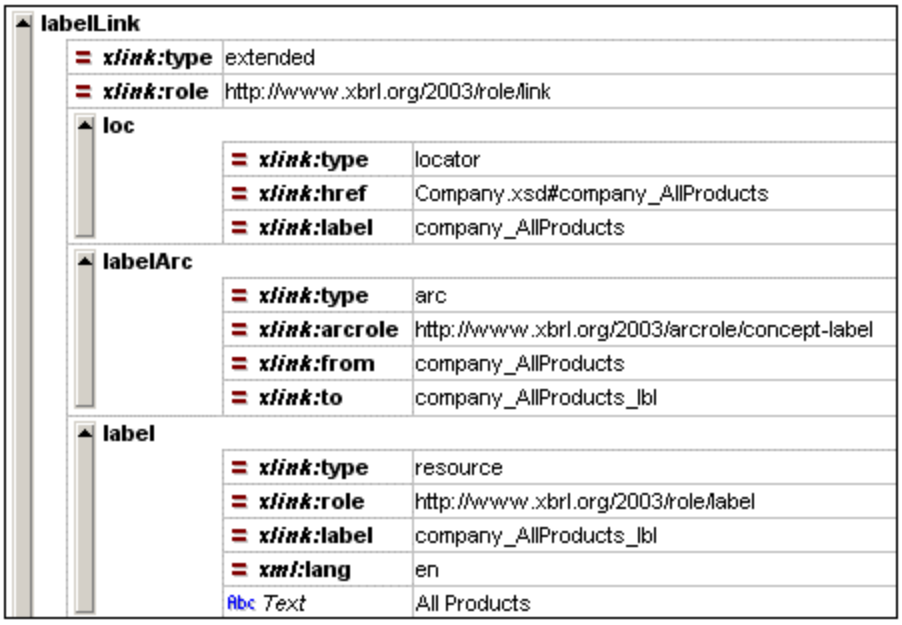
```
<xsd:annotation>
   <xsd:appinfo>
      <link:linkbaseRef xlink:arcrole="http://www.w3.org/1999/xlink/properties/linkbase"
         xlink:href="NanonullLabels.xml" xlink:type="simple"
         xlink:role="http://www.xbrl.org/2003/role/labelLinkbaseRef" />
      <link:linkbaseRef xlink:arcrole="http://www.w3.org/1999/xlink/properties/linkbase"
         xlink:href="NanonullDefinitions.xml" xlink:type="simple"
         xlink:role="http://www.xbrl.org/2003/role/definitionLinkbaseRef" />
      <link:linkbaseRef xlink:arcrole="http://www.w3.org/1999/xlink/properties/linkbase"
         xlink:href="NanonullPresentations.xml" xlink:type="simple"
         xlink:role="http://www.xbrl.org/2003/role/presentationLinkbaseRef" />
      <link:linkbaseRef xlink:arcrole="http://www.w3.org/1999/xlink/properties/linkbase"
         xlink:href="NanonullCalculations.xml" xlink:type="simple"
         xlink:role="http://www.xbrl.org/2003/role/calculationLinkbaseRef" />
      <link:linkbaseRef xlink:arcrole="http://www.w3.org/1999/xlink/properties/linkbase"
         xlink:href="NanonullReferences.xml" xlink:type="simple"
         xlink:role="http://www.xbrl.org/2003/role/referenceLinkbaseRef" />

   </xsd:appinfo>
</xsd:annotation>
```

Wenn die Concept-Definitionsdatei (die `.xsd` Datei) in der XBRL-Ansicht geöffnet ist, werden die verschiedenen Taxonomiedateien in der Eingabehilfe "Übersicht" 274 in Form einer Baumstruktur angezeigt (*Abbildung unten*).



Beachten Sie in der Abbildung oben die Symbole links von den Dateinamen. XML-Schemadateien (`.xsd`) sind durch ein XSD-Symbol gekennzeichnet. Die Beziehungsdateien haben ein farbiges Dateisymbol mit einem Buchstaben, der dem Anfangsbuchstaben der Beziehungsart entspricht. So steht z.B. ein Symbol für eine Definition-Beziehungsdatei, ein Symbol für eine Presentation-Beziehungsdatei usw. Wenn Sie auf eine dieser Dateien doppelklicken, wird sie in XMLSpy geöffnet, wo sie in der Grid-Ansicht (*Abbildung unten*) oder der Textansicht bearbeitet werden kann.



 © 2016-2022 Altova GmbH

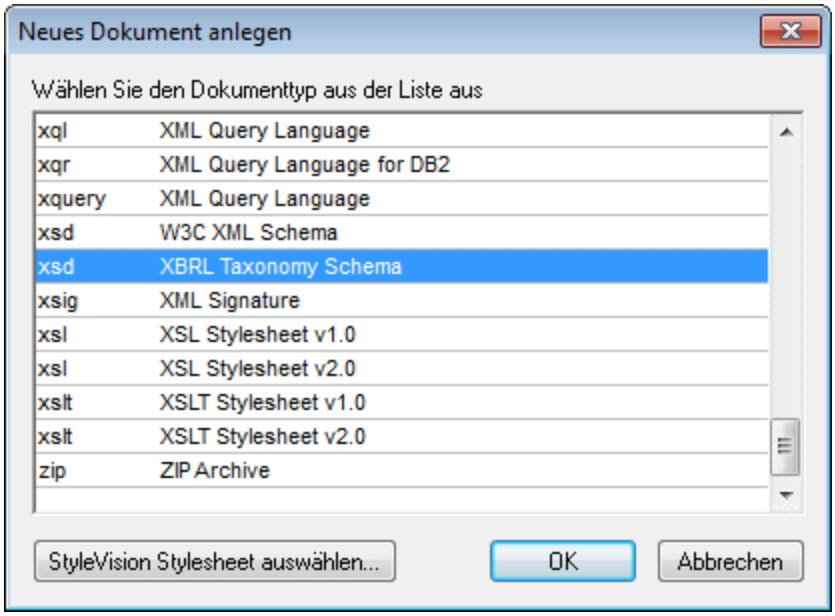
17.2.3 Erstellen einer neuen Taxonomie

Eine neue Taxonomie wird im Normalfall erstellt, um eine oder mehrere Standard-Taxonomien zu erweitern. Wenn eine neue Taxonomie auf einer Standard-Taxonomie oder einer bereits bestehenden Taxonomie aufbaut, muss die bestehende Taxonomie in diese importiert werden. Alternativ dazu kann eine Taxonomie auch von Grund auf neu erstellt werden. In der XBRL-Ansicht von XMLSpy können Sie mit Hilfe des Taxonomieassistenten ganz einfach US-GAAP- und IFRS-Taxonomien in Ihre Taxonomie importieren. Die importierte Taxonomie kann anschließend über die grafische Benutzeroberfläche der XBRL-Ansicht geändert werden.

Der erste Schritt beim Erstellen einer neuen Taxonomie ist die Erstellung einer Concept-Definitionsdatei, die eine XML-Schema (`.xsd`)-Datei ist. Neben Concept-Definitionen sind in dieser Datei der Namespace der neuen Taxonomie, der Pfad der zu importierenden Taxonomien, der Pfad der Beziehungsdateien der Taxonomie und der Namespace importierter Taxonomien sowie andere verwendete Namespaces definiert.

Erstellen der Concept-Definitionsdatei

Um eine neue XBRL-Taxonomie zu erstellen, wählen Sie den Menübefehl **Datei | Neu** aus. Daraufhin wird das Dialogfeld "Neues Dokument anlegen" aufgerufen (*Abbildung unten*).



Wählen Sie den Eintrag *xsd: XBRL Taxonomy Schema* aus und klicken Sie auf **OK**. Daraufhin wird eine neue Taxonomie erstellt.

Am besten speichern Sie die Taxonomie in einem eigenen Ordner, da es höchstwahrscheinlich andere Taxonomiekomponenten geben wird, die am besten in einem gemeinsamen Ordner gespeichert werden sollten.

Übersicht über die Schritte bei der Erstellung einer Taxonomie

Im Folgenden wird in groben Zügen erläutert, wie Sie eine Taxonomie erstellen.

1. Wählen Sie mit dem Menübefehl **XBRL | Importieren/Referenzieren**[852] die Basistaxonomie für Ihre Taxonomie aus. Wenn Sie eine Taxonomie von Grund auf neu erstellen möchten, überspringen Sie diesen Schritt.
2. Geben Sie der Taxonomie mit dem Menübefehl **XBRL | Target Namespace definieren**[856] einen Target Namespace.
3. Die Namespaces der importierten Basistaxonomie werden automatisch in Ihrer Taxonomie deklariert. Sie können gegebenenfalls noch andere Namespaces hinzufügen, wie im Kapitel Einrichten der Taxonomiedateien[859] beschrieben.
4. Anschließend können Sie die Basistaxonomie mit Ihren eigenen Elementen[861] und Beziehungen[865] erweitern.

Beispieldatei: Schritt 1

Erstellen Sie ein neues Taxonomiedokument und speichern Sie es unter dem gewünschten Namen im gewünschten Ordner. Diese Taxonomiedatei ist die Taxonomie-Hauptdatei oder Concept-Definitionsdatei. Es handelt sich um eine XML-Schema-Datei, die eine `.xsd` Dateierweiterung haben muss. Wir wollen die Datei, die wir hier erstellen, `Nanonull.xsd` nennen. Dies ist derselbe Name wie der des Beispiels in `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\XBRLExamples\Nanonull`.

Im nächsten Schritt importieren wir eine Basistaxonomie in unsere Taxonomie.

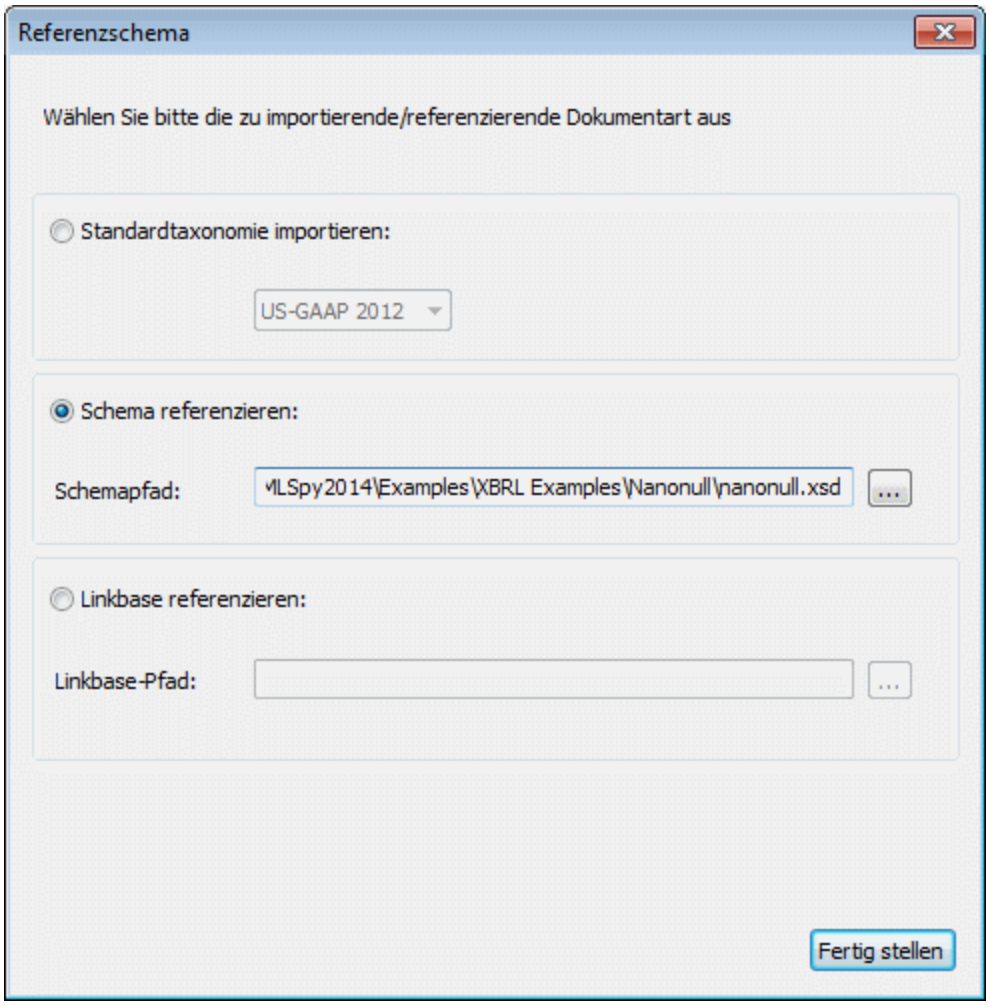
17.2.4 Importieren einer Basis-Taxonomie

Wenn eine neue Taxonomie auf einer bestehenden Taxonomie aufbauen soll, muss die bestehende Taxonomie in die neue Taxonomie importiert werden.

Um eine Taxonomie zu importieren, gehen Sie folgendermaßen vor:

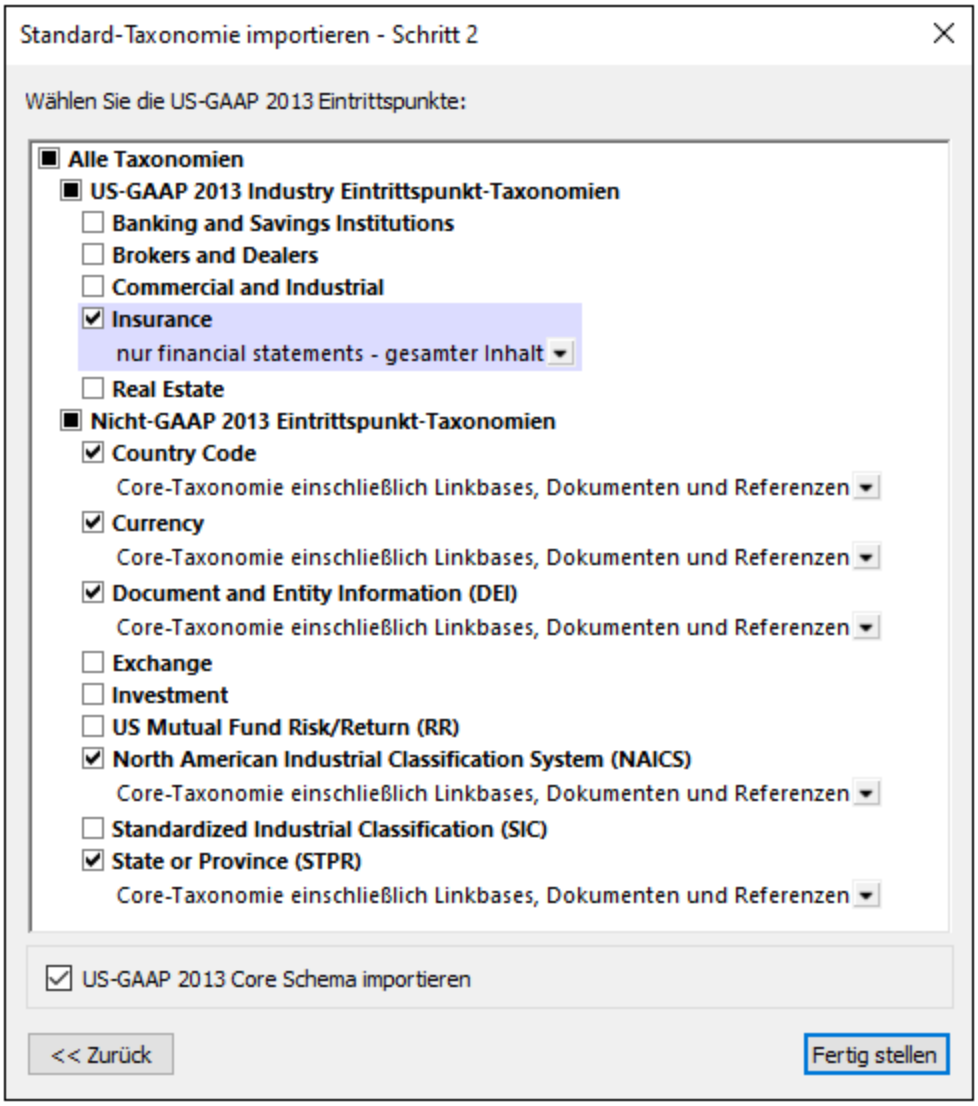
1. Klicken Sie mit der rechten Maustaste in der XBRL-Ansicht in die Eingabehilfe "Übersicht" und wählen Sie den Menübefehl **Importieren/Referenzieren**. Wählen Sie alternativ dazu den Menübefehl **XBRL | Importieren/Referenzieren**[1539].
2. Im Dialogfeld "Standard-Taxonomie importieren" (*Abbildung unten*), das daraufhin angezeigt wird, können Sie eine zu importierende Taxonomie oder eine zu referenzierende Linkbase auswählen. (Der Name des Dialogfelds ändert sich je nach gewählter Option).

© 2016-2022 Altova GmbH



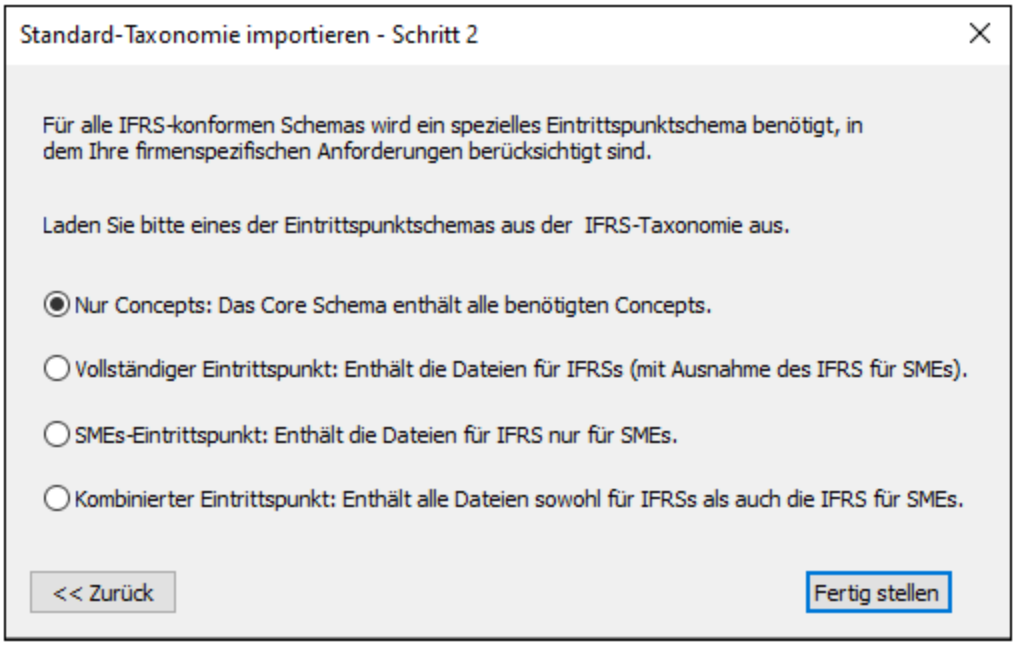
Es gibt drei Import-/Referenzierungsoptionen: (i) eine Standard-Taxonomie (US-GAAP oder IFRS); (ii) jede andere Taxonomie (Referenzschema); und (iii) eine Linkbase. Wenn Sie eine Nicht-Standard-Taxonomie importieren, aktivieren Sie das Optionsfeld *Schema referenzieren*, klicken Sie auf die Schaltfläche **Durchsuchen** des Textfelds "Schemapfad" und navigieren Sie zur gewünschten Schemadatei.

3. Klicken Sie anschließend auf **Fertig stellen**. Die ausgewählte Taxonomie wird importiert und ihre Elemente und Beziehungen werden in der XBRL-Ansicht angezeigt.
4. Wenn Sie eine US-GAAP-Taxonomie ausgewählt haben, wird ein neuer Bildschirm angezeigt, in dem Sie (ii) die Eintrittspunkte, die in die Taxonomie inkludiert werden sollen, auswählen können und (ii) angeben können, ob das US-GAAP Core Schema importiert werden soll (Kontrollkästchen am unteren Rand des Dialogfelds).



Wenn Sie als Basis für Ihre Taxonomie IFRS ausgewählt haben, können Sie einen IFRS-spezifischen Eintrittspunkt auswählen.

© 2016-2022 Altova GmbH



5. Klicken Sie anschließend auf **Fertig stellen**. Die ausgewählten Eintrittspunkt-Schemas werden importiert und von Ihrer Taxonomie referenziert. Die Taxonomie wird in der XBRL-Ansicht geöffnet und kann nun bearbeitet werden.

Beachten Sie die folgenden Punkte:

- In der Eingabehilfe "Übersicht" werden auch Taxonomien aufgelistet, die selbst in die importierte Taxonomie importiert werden, sowie Linkbases, die in der importierten Taxonomie verwendet werden.
- Im Fenster "Globale Elemente" werden Concepts aufgelistet, die in der importierten Taxonomie definiert sind.
- Importierte Concepts werden im Design-Fenster und in der Eingabehilfe "Details" in grauer Schrift angezeigt.
- Sie können eine importierte Taxonomie löschen, indem Sie mit der rechten Maustaste in der Eingabehilfe "Übersicht" darauf klicken und den Befehl **Entfernen** wählen.

Anmerkung: Wenn sich herausstellt, dass die Bearbeitungsgeschwindigkeit durch eine große Taxonomie wie z.B. US-GAAP verlangsamt wird, verwenden Sie den Filter im Hauptfenster 266, um nur Elemente anzuzeigen, die in der neuen erweiternden Taxonomie erstellt wurden. Dadurch wird die Bearbeitungsgeschwindigkeit erheblich gesteigert.

Funktionsweise beim Import

Wenn Sie wie oben beschrieben eine Standard-Taxonomie importieren, wird ein `xs:import` Element zur neuen Taxonomie hinzugefügt. Im `xs:import` Element werden der Namespace und Pfad der importierten Taxonomie definiert (*Liste unten*).

```
<xs:import namespace="http://fasb.org/us-gaap/2013-01-31"
           schemaLocation="http://xbrl.fasb.org/us-gaap/2013/elts/us-gaap-2013-01-31.xsd"/>
```

Im Codefragment oben definiert das `schemaLocation` Attribut, dass die Taxonomie über Internet geladen wird. Diese URI wird jedoch über eine XMLSpy Katalogdatei[463] auf eine lokale Kopie der US-GAAP-Taxonomie (die in Ihrem XMLSpy Paket enthalten ist) gemappt.

Um eine lokal gespeicherte Taxonomie zu verwenden, kann für die Pfadangabe zur Taxonomie auch eine lokale Adresse verwendet werden. Alternativ dazu kann eine Web-Adresse verwendet werden, die dann über eine Katalogdatei[463] auf eine lokale Adresse gemappt wird. Durch Aufrufen von Taxonomien über lokale Pfade lässt sich die Verarbeitungsgeschwindigkeit Ihres Rechners erheblich steigern.

Beispieldatei: Schritt 2

Importieren Sie gemäß den oben beschriebenen Schritten die US-GAAP 2013-Taxonomie als Basistaxonomie. Sehen Sie sich alle importierten Taxonomien und referenzierten Linkbases in der Eingabehilfe "Übersicht" genau an. Wechseln Sie in die Textansicht und suchen Sie nach `xs:import`-Elementen. Beachten Sie, dass importierte Concepts im Hauptfenster der XBRL-Ansicht in grauer Schrift angezeigt werden. Beachten Sie außerdem, dass die Linkbases und die importierten Schemas der US-GAAP-Taxonomie in der Eingabehilfe "Übersicht" aufgelistet sind.

Im nächsten Schritt werden wir den Target Namespace[856] der Taxonomie definieren und erläutern, wie Sie die Namespaces der Taxonomie bearbeiten.

17.2.5 Namespaces

Der Target Namespace

Der Target Namespace einer Taxonomie **ist im** `xs:targetNamespace`-Attribut des `xs:schema`-Elements der Taxonomie (*siehe Liste unten*) definiert. (Das `xs:schema` Element ist das Dokument-Element der Concept-Definitionsdatei.)

```
<xs:schema targetNamespace="http://www.altova.com/XBRL/Taxonomies">
    ...
</xs:schema>
```

Zusätzlich dazu (zur Definition des Target Namespace) muss der Target Namespace im `xs:schema` Element **deklariert** werden, sodass es für die gesamte Länge des Dokuments im Geltungsbereich ist. In der Liste unten wird der Namespace deklariert, der der Target Namespace ist.

```
<xs:schema targetNamespace="http://www.altova.com/XBRL/Taxonomies"
xmlns:ns1="http://www.altova.com/XBRL/Taxonomies" >
    ...
</xs:schema>
```

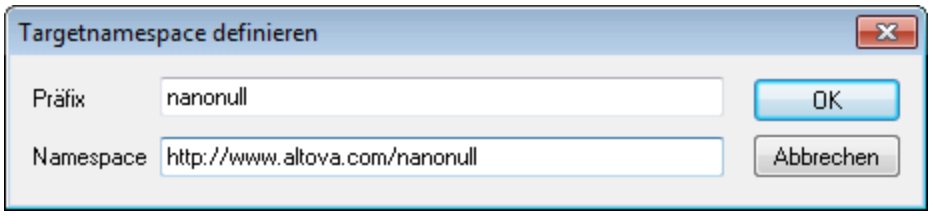
Im Codefragment oben ist der Namespace im Element `xs:schema` deklariert und hat das Präfix `ns1`.

Definieren des Target Namespace

Wenn mit dem Taxonomieassistenten eine neue Taxonomie erstellt wird, werden für die Taxonomie automatisch ein Standard-Target Namespace und ein Präfix generiert. Der Standard-Target Namespace basiert auf den Daten, die Sie im Taxonomieassistenten eingegeben haben. Das Präfix des Standard Target Namespace hat die Form `nX`, wobei `X` für eine Ganzzahl steht. Die Deklaration des Standard Target Namespace und das Präfix können über das Dialogfeld "Target Namespace definieren" (über den Befehl **XBRL | Target Namespaces definieren**) bearbeitet werden (*Abbildung unten*). Mit diesen Bearbeitung wird nicht nur

 © 2016-2022 Altova GmbH

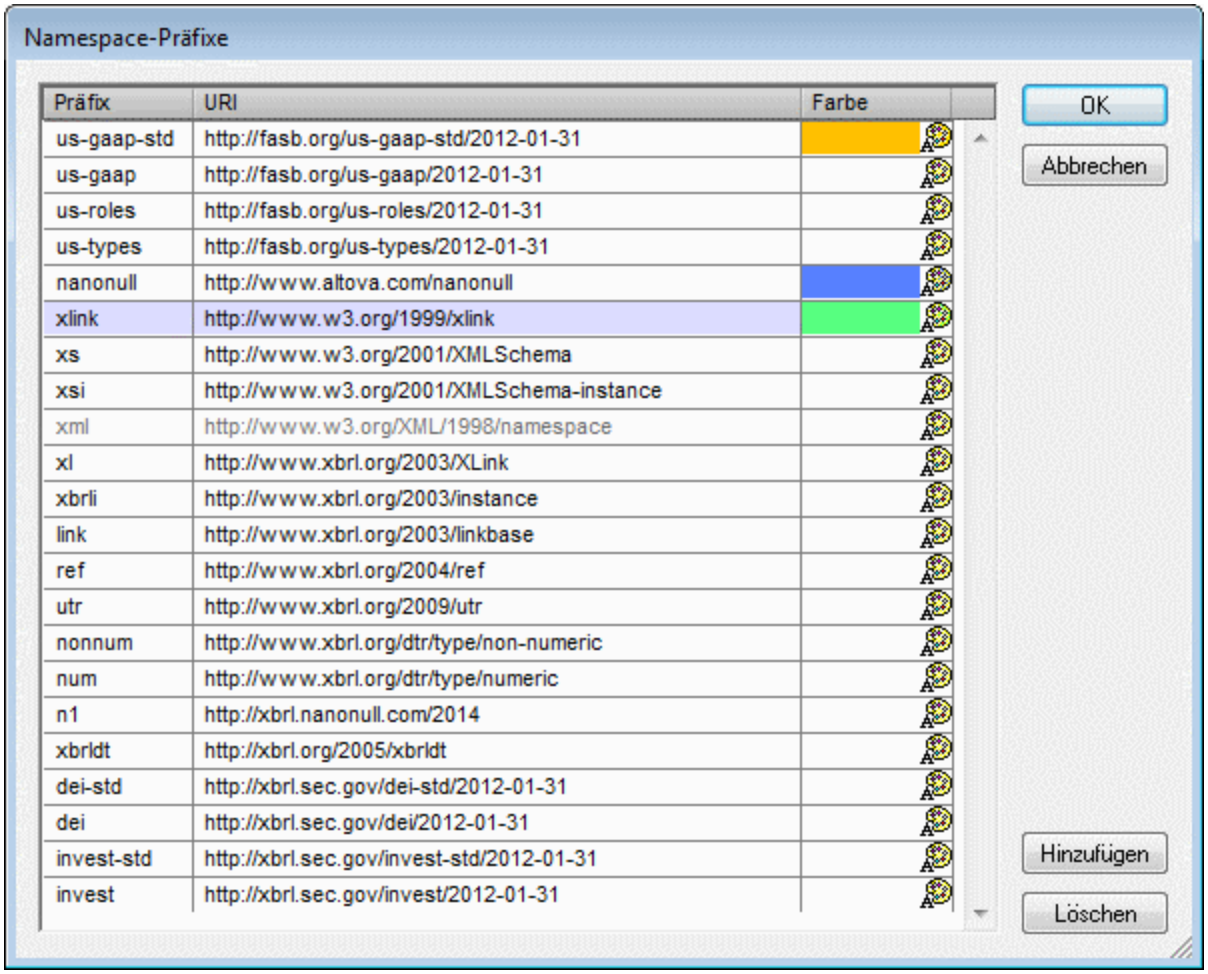
die **Definition** des Target Namespace (der Wert des `targetNamespace` Attributs) geändert, sondern auch die **Deklaration** des Target Namespace.



Um nur die Deklaration des Target Namespace, (nicht aber die Definition) oder die Deklaration jedes beliebigen Namespace zu ändern, bearbeiten Sie das Präfix und den Wert des Namespace im Dialogfeld "Namespace-Präfixe" (Befehl **XBRL | Namespace-Präfixe**).

Taxonomie-Namespaces

Taxonomie-Namespaces können im Dialogfeld "Namespace-Präfixe" (*Abbildung unten*) verwaltet werden, das in der XBRL-Ansicht über den Menübefehl **XBRL | Namespace-Präfixe** aufgerufen wird. Sie können in diesem Dialogfeld Namespaces deklarieren, und den einzelnen Namespaces Präfixe und Hintergrundfarben zuweisen. Bearbeitungen, die in diesem Dialogfeld vorgenommen werden, ändern die Deklarationen von Namespaces in der Taxonomie.



Im Dialogfeld "Namespace-Präfixe" sind alle Namespaces in der Taxonomie aufgelistet.

- Um einen Namespace hinzuzufügen oder zu löschen verwenden Sie den Befehl **Hinzufügen** bzw. **Löschen**. Nachdem Sie einen Namespace hinzugefügt haben, bearbeiten Sie das Standard-Präfix und die Standard-URI durch Doppelklick in das entsprechende Feld und Eingabe der Änderungen.
- Eine Farbe wird einem Namespace über die Farbpalette für diese Namespace zugewiesen. Wenn einem Namespace eine Farbe zugewiesen wurde, werden alle Komponenten in diesem Namespace im Hauptfenster und in den Eingabehilfe der XBRL-Ansicht mit dieser Hintergrundfarbe angezeigt. Beachten Sie, dass eine Farbeinstellung für einen bestimmten Namespace für diesen Namespace in allen Taxonomiedokumenten, die in der XBRL-Ansicht geöffnet sind, gilt.

Nachdem Sie die Bearbeitungen im Dialogfeld "Namespace-Präfixe" abgeschlossen haben, klicken Sie auf **OK** damit Ihre Änderungen wirksam werden.

Beispieldatei: Schritt 3

Öffnen Sie das Dialogfeld "Target Namespace definieren" über den Befehl **XBRL | Target Namespace definieren**. Doppelklicken Sie in die zu bearbeitenden Felder. Wir haben den Namespace `http://www.altova.com/nanonull` verwendet und ihm das Präfix `nanonull` (*siehe Abbildung unten*) zugewiesen. Wenn Sie im Dialogfeld auf **OK** klicken, wird der Target Namespace zugewiesen und mit dem von

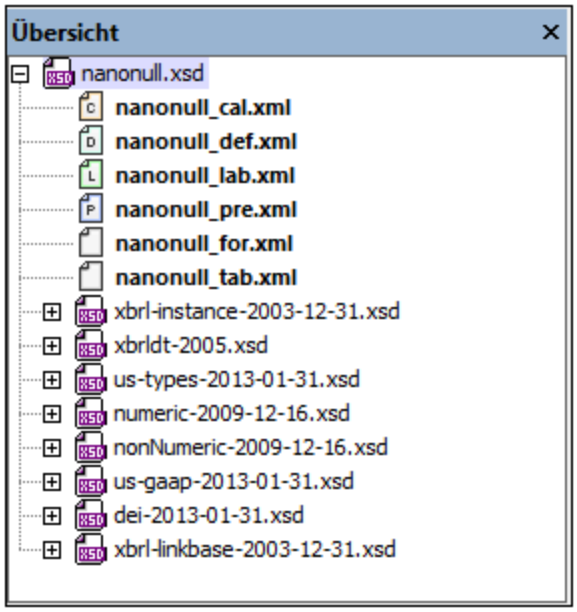
© 2016-2022 Altova GmbH

Ihnen zugewiesenen Präfix deklariert. In unserem Fall lauten der Target Namespace und das Präfix `http://www.altova.com/nanonull` bzw. `nanonull`.

Im nächsten Schritt[859] beschreiben wir Linkbase-Dateien und Referenzierungsmechanismen genauer.

17.2.6 Taxonomiedateien

In der Eingabehilfe "Übersicht" werden die Dateien, aus denen sich die Taxonomie zusammensetzt, in einer Baumstruktur angezeigt (*Abbildung unten*). An der Root der Struktur befindet sich die Taxonomie-Hauptdatei (die Concepts-Definitionsdatei); dies ist die derzeit aktive Datei. Die Dateien auf der nächsten Ebene gehören zwei Arten an: (i) Linkbase-Dateien, in denen die verschiedenen Beziehungen in der Taxonomie definiert sind; diese werden durch farbige Symbole[274] gekennzeichnet; und (ii) importierte Schemas (die `.xsd` Dateien).



Im Abschnitt Importieren einer Taxonomie[852] haben Sie nun gesehen, wie eine Taxonomie über die Eingabehilfe "Übersicht" importiert werden kann. Die importierte Taxonomie wird in der Eingabehilfe "Übersicht" unter den importierten Schemas aufgelistet.

In diesem Abschnitt wird beschrieben, wie Sie Linkbase-Dateien mit Hilfe der Eingabehilfe "Übersicht" verwalten können. Die folgenden vier Operationen zum Verwalten von Linkbases werden alle über das Kontextmenü der Eingabehilfe "Übersicht" aufgerufen:

- Hinzufügen neuer Linkbases[860] und Speichern der Linkbases mit der Taxonomie[860]
- Definieren der Linkbase-Art[860]. In Fällen, in denen XMLSpy den Linkbase-Typ (Calculation, Definition, Presentation, Label oder Reference) nicht kennt, kann der Linkbase-Typ explizit definiert werden.
- Definieren einer Linkbase als Standard-Linkbase[860] für eine bestimmte Art von Beziehungs-Linkbase. Wenn es mehr als eine Linkbase für eine bestimmte Beziehungsart, z.B. Label-Beziehungen gibt, werden die neuen Labels, die Sie im Taxonomie-Editor erstellen, in der Standard-Label-Linkbase erstellt.
- Löschen von Linkbases[860].

Anmerkung: Es gibt die folgenden Arten von Beziehungen: (i) Definition, (ii) Calculation (Berechnung), (iii) Presentation (Darstellung), (iv) Label und (v) Reference. Für jede dieser Beziehungsarten können separate Linkbase-Dateien erstellt werden.

Hinzufügen einer neuen Linkbase

Um eine neue Linkbase hinzuzufügen, gehen Sie folgendermaßen vor:

1. Klicken Sie mit der rechten Maustaste in die Eingabehilfe "Übersicht" und wählen Sie den Befehl **Neue Linkbase hinzufügen | <Beziehungsart>**. Daraufhin wird in der Eingabehilfe "Übersicht" eine neue Linkbase-Datei des ausgewählten Beziehungstyps mit einem Standardnamen erstellt. Beachten Sie, dass die neue Linkbase als Standard-Linkbase ihres Beziehungstyps erstellt wird (dies wird durch den fett gedruckten Dateinamen angezeigt).
2. Klicken Sie mit der rechten Maustaste auf den Standardnamen, wählen Sie den Befehl **Umbenennen** und bearbeiten Sie den Namen.
3. Eine neu erstellte Linkbase wird erst dann physisch in einem bestimmten Ordner gespeichert, wenn die Taxonomie-Hauptdatei das nächste Mal gespeichert wird. Nähere Informationen dazu finden Sie weiter unten.

Speichern von Linkbase-Dateien

Wenn eine Linkbase-Datei noch nicht gespeichert wurde, wird dies durch ein Sternchen hinter dem Namen der Linkbase-Datei gekennzeichnet. Wenn Sie die Taxonomie-Hauptdatei speichern, passiert Folgendes:

1. Das Dialogfeld "Linkbase-Pfade bestätigen" wird angezeigt. Dieses Dialogfeld enthält die Namen und Pfade aller Linkbases in der Taxonomie, einschließlich der neu erstellten Linkbase-Dateien. Eine nicht gespeicherte Linkbase-Datei hat einen Standardpfad zu dem Ordner, in dem die Taxonomie-Hauptdatei gespeichert werden wird bzw. wurde. Wenn Sie eine Linkbase-Datei unter einem anderen Pfad speichern möchten, können Sie ihren Pfad bearbeiten. Sie können auch den Namen der Datei bearbeiten.
2. Klicken Sie abschließend auf **OK**. Die Linkbase-Dateien werden unter den entsprechenden Pfaden gespeichert.

Definieren der Linkbase-Art

Mit diesem Befehl können Sie die Linkbase-Art einer Datei (auch als Linkbase-Typ einer Datei bezeichnet) definieren. Klicken Sie mit der rechten Maustaste auf die Datei, deren Linkbase-Art Sie ändern möchten und wählen Sie im Kontextmenü den Befehl **Linkbase Kind definieren | <Beziehungsart>**. Mit der Option **Alle** können Sie festlegen, dass die Linkbase-Datei mehr als eine Beziehungsart enthalten kann.

Definieren einer Standard-Linkbase

Sie können für jede Beziehungsart eine Standard-Linkbase-Datei definieren. Wenn im Taxonomie-Editor eine Beziehung dieser Art definiert wird, wird diese in der Standard-Linkbase-Datei dieser Beziehungsart gespeichert. Um eine Linkbase-Datei als Standard-Linkbase zu definieren, klicken Sie mit der rechten Maustaste darauf und wählen Sie den Befehl **Standard-Linkbase definieren**. Die Namen der Standard-Linkbases werden fett angezeigt.

Löschen einer Linkbase

Um eine Linkbase aus der Taxonomie zu entfernen, klicken Sie mit der rechten Maustaste darauf und wählen Sie den Befehl **Entfernen**.

© 2016-2022 Altova GmbH

Beispieldatei: Schritt 4

Fügen Sie die Linkbases wie oben beschrieben hinzu. Die Linkbase-Dateien sind:

- Calculation Linkbase: `nanonull_cal.xml`
- Definition Linkbase: `nanonull_def.xml`
- Label Linkbase: `nanonull_lab.xml`
- Presentation Linkbase: `nanonull_pre.xml`

Wenn Sie die Linkbase-Dateien nicht verwenden möchten, können Sie mit der `Nanonull.xsd`-Taxonomie aus dem Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\XBRLExamples\Nanonull`), arbeiten, die bereits über Linkbase-Dateien verfügt.

Um einige der in diesem Abschnitt beschriebenen Befehle zu testen, gehen Sie folgendermaßen vor: Erstellen Sie mit dem Befehl **Neue Linkbase hinzufügen** eine Linkbase-Datei. Wählen Sie eine beliebige Linkbase-Art aus. Benennen Sie die Linkbase um, wie oben beschrieben. Beachten Sie, dass diese neu erstellte Linkbase nun die Standard-Linkbase ihres Beziehungstyps ist (ihr Name wird fett angezeigt). Wählen Sie sie aus und definieren Sie einen anderen Beziehungstyp dafür (mit dem Befehl **Linkbase-Art definieren**). Beachten Sie, dass die Datei **nicht** die Standard-Linkbase ihres neuen Beziehungstyps ist. Löschen Sie die Linkbase nun (mit dem Befehl **Entfernen**). Da nun eine der ursprünglichen Linkbase-Dateien keine Standard-Linkbase mehr ist, definieren Sie eine Datei dieses Beziehungstyps als die Standard-Linkbase ihres Beziehungstyps.

Im nächsten Schritt werden wir neue Elemente zur Taxonomie-Hauptdatei (oder Concept-Definitions-Datei) hinzufügen 861.

17.2.7 Hinzufügen von Elementen zu einer Taxonomie

Um ein Element (Concept) zur Taxonomie hinzuzufügen, klicken Sie im Hauptfenster auf die Schaltfläche **Neues Element hinzufügen** (*Abbildung unten*).

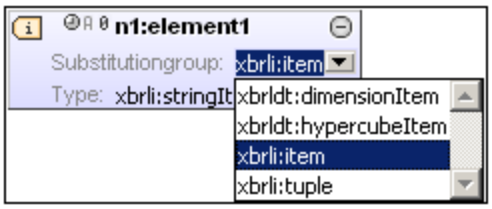


Das neue Element mit der Substitution Group `xbrli:item` und einem Standard-Namen wird zur Liste der Elemente in der Anzeige hinzugefügt (*Abbildung unten*).



Eine Beschreibung des Elementkastens finden Sie unter Hauptfenster: Register "Elemente" 266. Sie können nun im Hauptfenster 254 die Eigenschaften des Elements auf folgende Arten ändern:

- Um den Namen des Elements zu ändern, doppelklicken Sie auf den Standardnamen und geben Sie den gewünschten Namen ein. Beachten Sie, dass Sie das richtige Namespace-Präfix 856 für den Namen eingeben müssen.
- Die Substitution Group des Elements kann durch Erweitern des Elementkastens - klicken Sie dazu auf das Plus-Symbol - und anschließende Auswahl der gewünschten Substitution Group aus der Dropdown-Liste des Felds (*Abbildung unten*) geändert werden.



- Um die Eigenschaften Balance, Zeitraum, Abstract oder Nillable zu ändern, klicken Sie auf das entsprechende Symbol links vom Elementnamen und wählen Sie eine der Optionen aus, die im Popup-Fenster angezeigt werden.
- Um eine Label Linkrole für das Element hinzuzufügen, klicken Sie mit der rechten Maustaste an eine beliebige Stelle im Elementkasten und wählen Sie den Befehl **Label Linkrole hinzufügen**. Daraufhin wird eine Zeile für die Label Linkrole hinzugefügt; in diese Zeile können Sie die Label Linkrole eingeben oder Sie wählen aus der Auswahlliste eine Option aus. Beachten Sie: Wenn keine Label Linkbase mit der Taxonomie verknüpft ist, wird jetzt eine erstellt und in der Eingabehilfe "Übersicht" angezeigt. 274
- Um ein Label für eine Label Linkrole hinzuzufügen, klicken Sie mit der rechten Maustaste auf die Label Linkrole und wählen Sie den Befehl **Label hinzufügen.** Um die Einzelheiten des Label einzugeben, doppelklicken Sie entweder auf das gewünschte Feld und geben Sie den neuen Wert ein oder wählen Sie den neuen Wert aus den entsprechenden Dropdown-Listen aus. Die Änderungen, die Sie an Labels vornehmen, werden in der Label Linkbase gespeichert, wenn die Taxonomie-Hauptdatei gespeichert wird.
- References werden auf dieselbe Art zur Reference Linkbase hinzugefügt, wie Labels zur Label Linkbase hinzugefügt werden. Zuerst wird eine Reference Linkrole für das Element hinzugefügt, anschließend wird eine Reference für eine bestimmte Reference Linkrole hinzugefügt.

Elementeigenschaften können auch in der Eingabehilfe "Details" bearbeitet werden. Eine Beschreibung dazu finden Sie unter Eingabehilfen in der XBRL-Ansicht 274.

Beispieldatei: Schritt 5

In diesem Abschnitt erweitern wir die US-GAAP-Taxonomie durch Erstellen neuer Elemente.

Als erstes erstellen wir das Element `nanonull:OnboardAndOther`, welches für Einnahmen aus dem Verkauf von Artikeln an Bord von Nanonull-Kreuzfahrtschiffen steht. Diese bestimmte Einnahmenüberschrift steht in der US-GAAP-Taxonomie nicht zur Verfügung, weshalb Sie als Erweiterung zu US-GAAP erstellt werden muss. Als neues Element, das speziell für die Nanonull-Taxonomie erstellt wird, muss es im Nanonull Namespace (`http://www.altova.com/nanonull`) erstellt werden, welcher mit dem Präfix `nanonull`deklariert wurde. Indem Sie das Element mit diesem Präfix erstellen, wird es in den Namespace Nanonull platziert.

Um das Element zu erstellen, gehen Sie folgendermaßen vor:

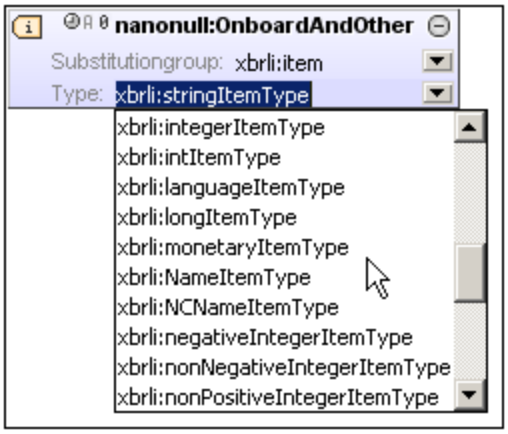
1. Klicken Sie im Hauptfenster auf die Schaltfläche **Neues Element hinzufügen** (*Abbildung unten*):



Zur Liste der Elemente in der Anzeige wird ein neues Element mit einer Substitution Group `xbrli:item` und mit einem Standardnamen hinzugefügt (*Abbildung unten*).



2. Doppelklicken Sie auf den Elementnamen und geben Sie den Namen `nanonull:OnboardAndOther` ein (*Abbildung unten*). Daraufhin wird das Element `OnboardAndOther` im Nanonull Namespace erstellt.
3. Erweitern Sie den Elementkasten. Da das Element einen Geldbetrag enthalten soll, ändern Sie das `Type` Attribute in `xbrli:monetaryItemType` (*Abbildung unten*).



4. Klicken Sie nun auf die Stelle links vom Uhrsymbol und wählen Sie aus dem Popup-Fenster den Eintrag `credit` (*Abbildung unten*).



Daraufhin wird der Wert des `xbrli:balance` Attributs auf `credit` gesetzt.

5. Klicken Sie auf das Uhr-, das `A`- und das `0`-Symbol und setzen Sie die Werte der Attribute `xbrli:duration`, `xs:abstract` und `xs:nillable` auf *duration, NOT Abstract bzw. Nillable.* (in der .xsd-Datei lauten die tatsächlichen Werte dann: `credit`, `duration`, `false`, bzw. `true`).
6. Rechtsklicken Sie auf den Elementkasten und wählen Sie im Menü, das angezeigt wird, den Befehl **Label Linkrole hinzufügen**. Daraufhin wird am unteren Rand des Elementkastens eine Label Linkrole-Zeile erstellt (*Abbildung unten*).



7. Wählen Sie die XBRL Link URI aus.
8. Rechtsklicken Sie auf die Label Linkrole-Zeile und wählen Sie im Popup-Menü den Befehl **Label hinzufügen**. Daraufhin wird in der Label Linkrole ein Label erstellt.
9. Doppelklicken Sie in das language-Feld der neu erstellten Label-Zeile (*Abbildung unten*) und geben Sie `en-us` ein; Wählen Sie im nächsten Feld, dem Linkrole-Feld, die `documentation` Role aus der Dropdown-Liste aus; geben Sie in das Label-Feld den Text ein, der in der Dokumentation zu sehen sein soll. Erstellen Sie anschließend durch Wiederholung von Schritt 9 für die `label` Linkrole eine weitere Label-Zeile. Wenn die Anzeige eines Elements (durch Klicken auf die Pfeilspitze links davon) erweitert wurde, kann die Anzeige der `label` Role aktiviert/deaktiviert werden, indem Sie auf das Plus/Minus-Symbol rechts vom Label klicken (Labels ein-/ausblenden).



Das Element `nanonull:OnboardAndOther` wurde nun erfolgreich erstellt.

Beachten Sie, dass `OnboardAndOther` einen `xbrli:balance` Wert "credit" hatte. Der Grund dafür ist, dass es sich um ein Einnahmenelement handelt: Es kommt Geld herein. Da die an Bord verkauften Artikel mit Kosten verbunden sind, d.h. ihr Erwerb kostet die Firma Geld, werden wir auch auf der Soll-Seite ein Element namens `nanonull:CostOfOnboardAndOther` erstellen. Erstellen Sie dieses Element auf dieselbe Art wie `nanonull:OnboardAndOther`, jedoch mit einem Unterschied: Setzen Sie den Wert von `xbrli:balance` auf `debit` anstelle von `credit`.

Des Weiteren müssen auch die Kosten für Kommissionen für Reisebüroagenten berücksichtigt werden. Erstellen Sie dazu ein Soll-Element namens `nanonull:CruiseCommissionsTransportationAndOther`. Erstellen Sie dieses Element genau wie `nanonull:CostOfOnboardAndOther`.

Schließlich fügen wir noch drei abstract Elements, nämlich `Asia`, `Europe` und `US` hinzu, damit die Concepts nach Region gruppiert werden können. Da die Elemente nur zum Gruppieren dienen und selbst keine Werte haben, werden sie als abstract Elements oder abstrakte Elemente bezeichnet. Welchen Typ ein solches Element hat, spielt daher keine Rolle. Am besten Sie geben einem abstract Element einen Typ, der der Semantik entspricht. So haben wir z.B. den abstract Elementen `Asia`, `Europe` und `US` den Typ `stringItemType` zugewiesen. Erstellen Sie die Elemente `nanonull:Asia`, `nanonull:Europe` und `nanonull:USA` genau wie die vorhergehenden Elemente. Der einzige Unterschied ist diesmal, dass der Wert des Attributs "Abstract" auf `true` gesetzt werden muss und dass es kein `xbrli:balance` Attribut gibt.

Anmerkung: Wenn in einem abstract Element ein `xbrli:balance` Attribut vorhanden ist, muss dieses abstract Element den Typ `monetaryItemType` haben, andernfalls ist die Taxonomie ungültig. Es empfiehlt sich, das optionale Attribut `xbrli:balance` in allen abstract Elementen wegzulassen.

 © 2016-2022 Altova GmbH

Im nächsten Schritt definieren wir Linkroles [865] für die neue Taxonomie. Diese Linkroles werden beim Erstellen neuer Beziehungen benötigt.

17.2.8 Beziehungen und Linkroles

Wenn eine Gruppe von Beziehungen erstellt wird, so werden diese Beziehungen innerhalb eines Elements erstellt. Wenn z.B. eine Definition Relationship erstellt wird, werden die Elemente, die Definition Relationships definieren (die Locators und Definition Arcs) alle innerhalb eines `definitionLink` Elements erstellt, das in etwa folgendermaßen aussieht:

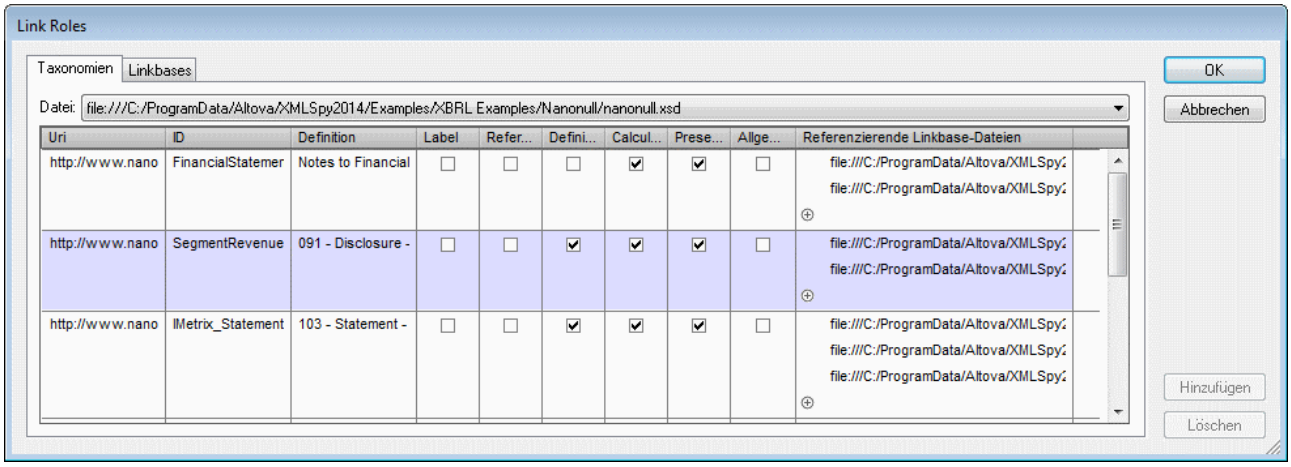
```
<link:definitionLink xlink:type="extended"
xlink:role="http://www.nanonull.com/taxonomy/role/SegmentRevenueAndOperatingIncome">
```

Der Wert des `xlink:role` Attributs im Definition Link muss (wie im oben aufgelisteten Definition Link) der Wert des `roleURI` Attributs einer der Linkroles sein, die zur Verwendung in Definition Relationships (*siehe Auflistung unten*) definiert wurden. Eine Linkrole (*wie in der Auflistung unten*) ist im `appinfo` Element der Taxonomie enthalten.

```
<xs:appinfo>
   <link:roleType id="SegmentRevenueAndOperatingIncome"
   roleURI="http://www.nanonull.com/taxonomy/role/SegmentRevenueAndOperatingIncome">
      <link:definition>006091 - Disclosure - Segment Revenue and Operating
Income</link:definition>
      <link:usedOn>link:calculationLink</link:usedOn>
      <link:usedOn>link:definitionLink</link:usedOn>
      <link:usedOn>link:presentationLink</link:usedOn>
   </link:roleType>
</xs:appinfo>
```

Linkroles können nicht nur in `definitionLink` Elementen, sondern auch in den umschließenden Elementen anderer Beziehungsarten (z.B. in `calculationLink` und `presentationLink` Elementen) verwendet werden. Beachten Sie, dass die Auflistung oben `usedOn` Elemente enthält, die definieren, in welcher Art von Beziehungen diese Linkrole verwendet werden darf.

Um Linkroles in einer Concept Definitions-Datei (Taxonomie-Hauptdatei) zu erstellen, klicken Sie in der XBRL-Ansicht auf den Menübefehl **XBRL | Linkroles**. Daraufhin wird das Dialogfeld "Link Roles" angezeigt (*Abbildung unten*).



Wählen Sie die Taxonomie-Datei auf dem Register "Taxonomien" aus der Dropdown-Liste in der Auswahlliste *Datei* aus und klicken Sie auf **Hinzufügen**, um eine Linkrole hinzuzufügen und definieren Sie anschließend die URI und ID der Linkrole (*siehe Liste oben*). Definieren Sie nun, für welche Arten von Beziehungen diese Linkrole zur Verfügung stehen soll; aktivieren Sie dazu die entsprechenden Kontrollkästchen der gewünschten Beziehungsarten *(siehe Abbildung oben*).

Beispieldatei: Schritt 6

Erstellen Sie über das Dialogfeld "Link Roles" (**XBRL | Linkroles**) zwei Linkroles, wie oben beschrieben und in der Abbildung oben gezeigt:

1. `id="SegmentRevenueAndOperatingIncome"` `URI="http://www.nanonull.com/taxonomy/role/SegmentRevenueAndOperatingIncome"` (zur Verwendung in Definition, Calculation und Presentation-Beziehungen)
2. `id="FinancialStatements"` `URI="http://www.nanonull.com/taxonomy/role/FinancialStatements"` (zur Verwendung in Calculation- und Presentation-Beziehungen)

Im nächsten Schritt erstellen wir Beziehungen 866 für die neue Taxonomie.

17.2.9 Erstellen von Beziehungen : Teil 1

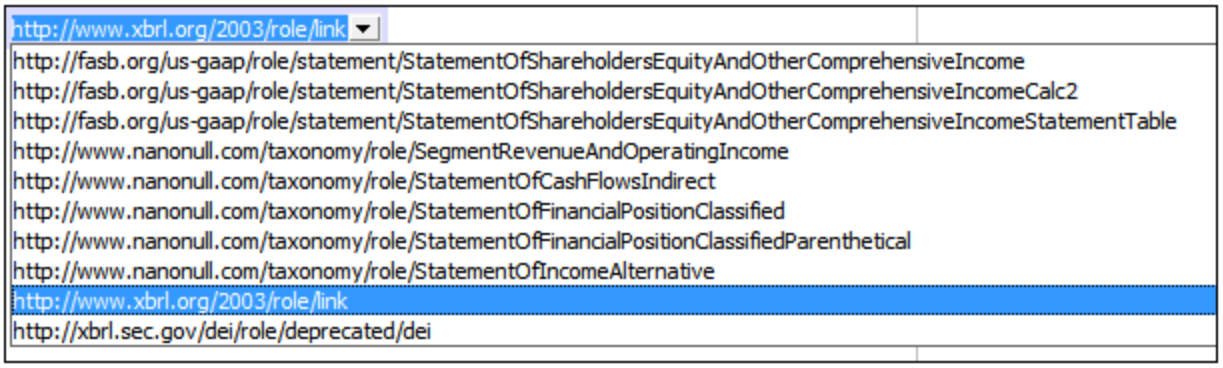
Beziehungen werden auf ihren jeweiligen Registern erstellt: Definitionen, Presentation und Calculation. Alle diese drei Arten von Beziehungen werden auf ähnliche Art erstellt. Der größte Unterschied ist, dass Definitionsbeziehungen (Definition Relationships) Arcroles haben, während Darstellungsbeziehungen (Presentation Relationships) und Berechnungsbeziehungen (Calculation Relationships) keine Arcroles haben. In diesem Abschnitt wird anhand von Definitionsbeziehungen beschrieben, wie man Beziehungen erstellt. Im nächsten Abschnitt 869 wird beschrieben, inwiefern sich Darstellungs- und Berechnungsbeziehungen unterscheiden. Außerdem werden weitere Eigenschaften im Zusammenhang mit Beziehungen beschrieben.

Am besten Sie öffnen eine fertige Taxonomie in der XBRL-Ansicht, wenn Sie die folgende Beschreibung lesen. Sie finden die Nanonull-Taxonomie (`nanonull.xsd`) im Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\XBRLExamples\Nanonull`.

© 2016-2022 Altova GmbH

Hinzufügen der Linkrole

Klicken Sie im Hauptfenster auf das gewünschte Beziehungsregister (Definitionen, Presentation, Calculation). Klicken Sie anschließend mit der rechten Maustaste in das Hauptfenster und wählen Sie den Befehl **Extended Link Role hinzufügen.** Daraufhin wird eine Zeile hinzugefügt, die die URI einer Standard-Linkrole enthält (*siehe Abbildung unten*). Klicken Sie auf den Dropdown-Pfeil rechts von dieser Zeile, um eine Liste verfügbarer Linkroles anzuzeigen und wählen Sie die gewünschte Linkrole aus.

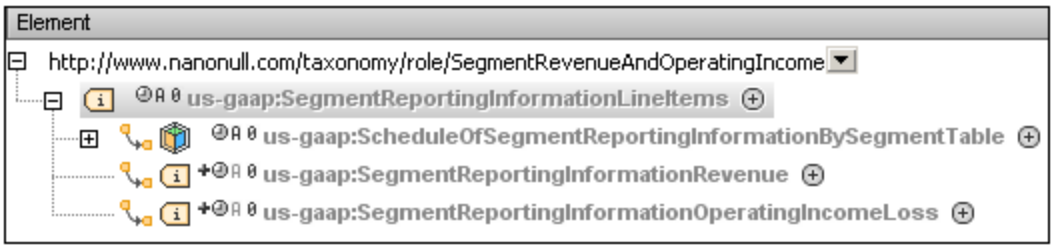


Wenn die gewünschte Linkrole nicht zur Verfügung steht, liegt das daran, dass sie entweder in der Taxonomie nicht definiert wurde oder nicht für die aktuelle Beziehungsart definiert wurde. Nähere Informationen zu Linkroles und wie man sie erstellt, finden Sie unter Beziehungen und Linkroles 865.

Es können beliebig viele Linkroles hinzugefügt werden.

Einfügen von Elementreferenzen und Arcs in eine Linkrole

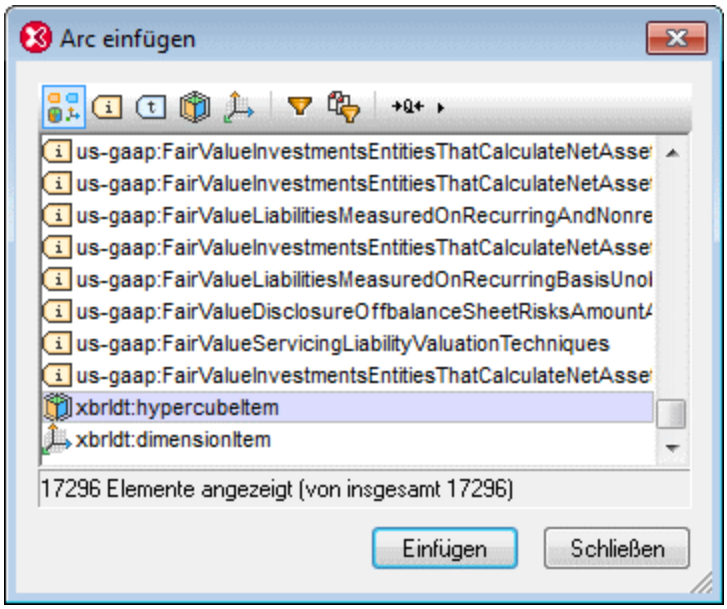
Das erste Element, das in einer Linkrole erstellt werden muss, ist ein Element, **von dem aus** eine Beziehung zu einem anderen Element erstellt wird (*siehe Abbildung unten*). Dabei handelt es sich normalerweise um ein abstraktes Element, in dem andere Elemente gruppiert werden (z.B. ein Element für eine Bilanz). Dieses Element verfügt in der Arcrole-Spalte über keinen Eintrag, da es sich am from Ende eines Arc befindet. Arcroles werden in den Elementen am to Ende eines Arc aufgelistet.



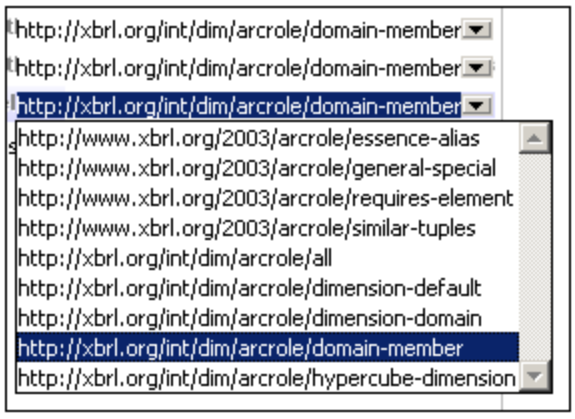
In der Abbildung oben ist das markierte Element die eingefügte Elementreferenz. Es hat drei Arcs, einen zu einem Hypercube-Element und zwei zu Item-Elementen. Diese drei Elemente befinden sich am to End ihrer jeweiligen Arcs und die from-to Beziehung wird durch die entsprechenden Arcroles, die in der Arcrole-Spalte angezeigt werden, definiert.

Um einen Arc in eine Elementreferenz oder ein Element einzufügen, rechtsklicken Sie auf das from Element und wählen Sie im Kontextmenü den Befehl **Arc einfügen.** Daraufhin erscheint das Dialogfeld "Arc einfügen" (*Abbildung unten)*. Wählen Sie das Element aus, das am to Ende des Arc erstellt werden soll. Um

die Ansicht in diesem Dialogfeld zu filtern, aktivieren Sie den Filter und definieren Sie eine Bedingung für den Filter (Eine Beschreibung dazu finden Sie unter Eingabehilfe in der XBRL-Ansicht[276]).



Das Element wird mit einer Standard-Arcrole eingefügt. Sie können die Arcrole ändern, indem Sie eine andere aus der Dropdown-Liste der Arcrole auswählen (*Abbildung unten*).

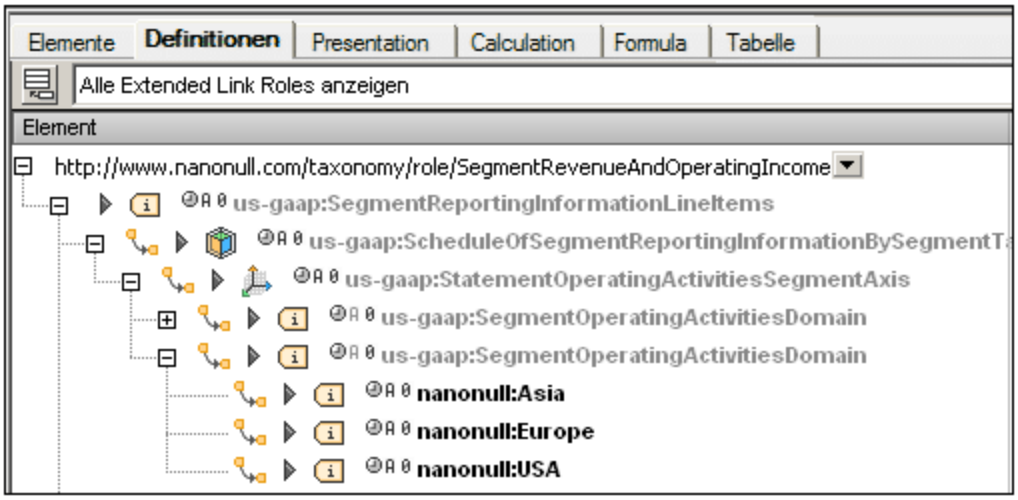


Hinweis: Elemente mit Arcs können auch durch Ziehen der Elemente aus der Eingabehilfe "Globale Elemente" hinzugefügt werden.

Beispieldatei: Schritt 7

Erstellen Sie mit der oben beschriebenen Methode Definitionsbeziehungen wie in der Abbildung unten gezeigt.

© 2016-2022 Altova GmbH



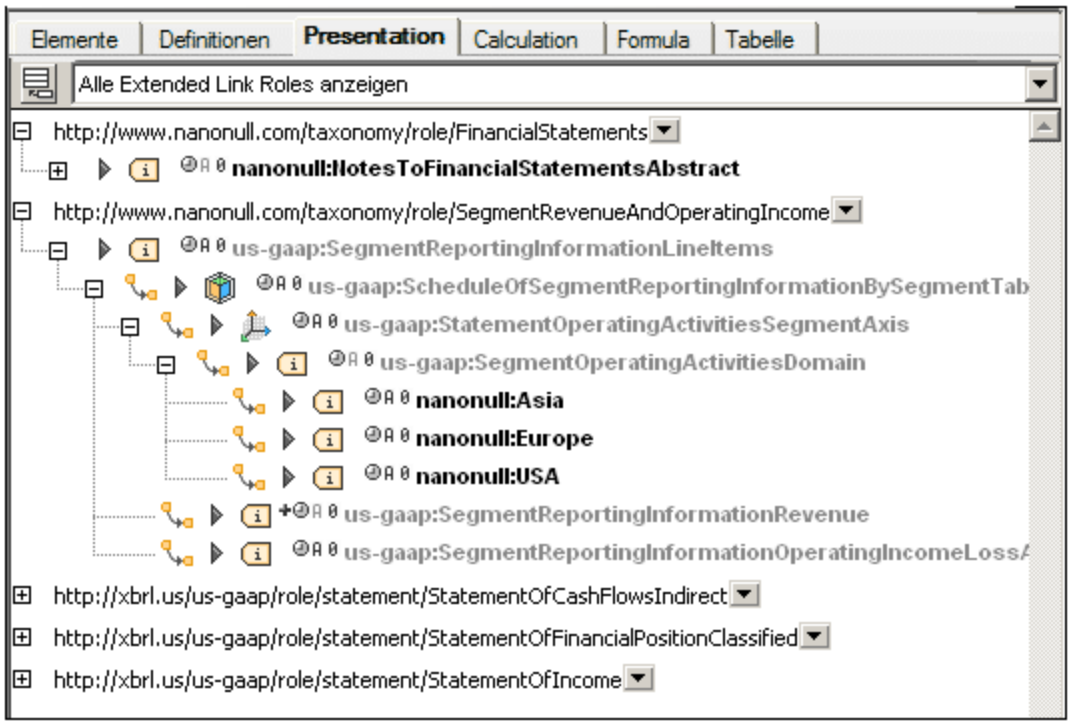
In der Abbildung oben sehen Sie die Elemente, die mit Arcs hinzufügt werden sollen. In der Abbildung unten sehen Sie die Arcroles der neu hinzugefügten Elemente.



Sie können die von Ihnen erstellte Taxonomie mit der aus dem XMLSpy Paket vergleichen. Diese (`nanonull.xsd`) befindet sich im Ordner `C:\Dokumente und Einstellungen\<Benutzer>\Eigene Dokumente\Altova\XMLSpy2023\Examples\XBRLExamples\Nanonull`.

17.2.10 Erstellen von Beziehungen: Teil 2

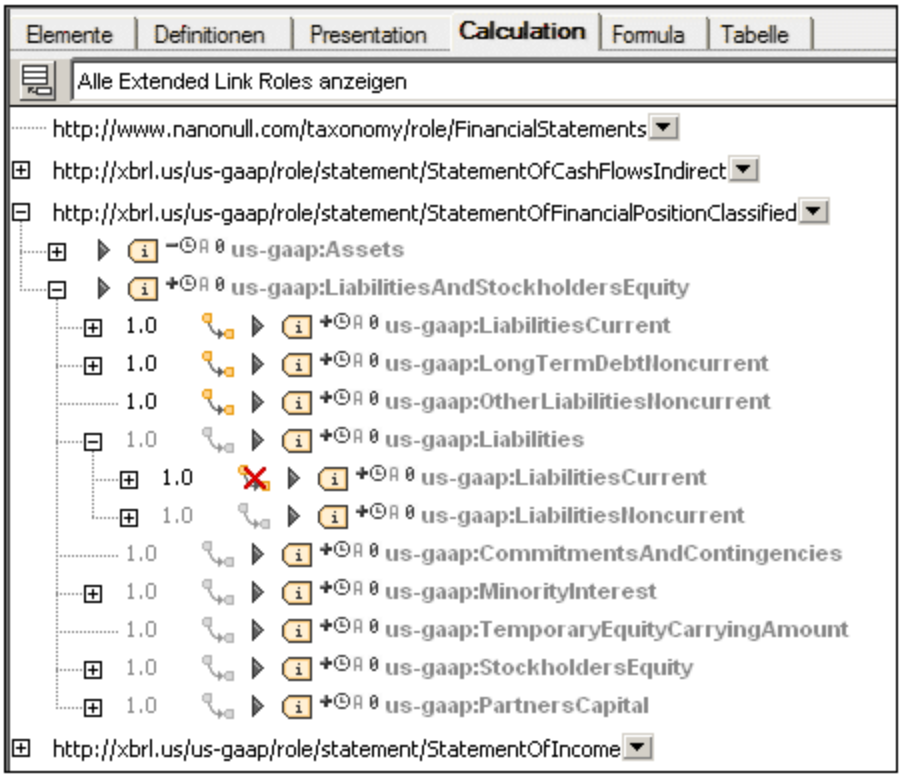
Im vorherigen Abschnitt, Erstellen von Beziehungen, Teil 1 866, wird die Funktionsweise der Erstellung von Beziehungen anhand von Definitionsbeziehungen erklärt. Darstellungsbeziehungen (Presentation Relationships) (*Abbildung unten*) und Calculation Relationships (Berechnungsbeziehungen) werden auf ähnliche Art erstellt. Der einzige Unterschied ist, dass es in Darstellungs- und Berechnungsbeziehungen keine Arcroles gibt.



Beachten Sie bitte die folgenden Punkte:

- Darstellungs- und Berechnungsbeziehungen können als einfacher Arc zwischen zwei Elementen in der Art Parent-Child-Beziehungen betrachtet werden. Die Arc-Symbole bezeichnen diese Beziehungen. Wenn Sie also einen Arc in einem Element einfügen, bedeutet dies, dass in der grafischen Darstellung ein Sub-Eintrag erstellt wird. Mit Hilfe von Arcs kann daher eine Hierarchie erstellt werden.
- Elemente können auch aus der Eingabehilfe "Globale Elemente" in die Struktur gezogen werden. Diese Elemente werden immer an die to Position eines Arc gezogen. Wenn das Element sich an einer zulässigen Position befinden, erscheint ein Pfeilsymbol.
- Calculation Arcs haben `weight` Attribute, die anzeigen, wie der Wert des to Elements im Arc summiert werden soll (*siehe Abbildung unten*). So gibt z.B. ein weight-Wert `+1.0` an, dass 100% des Elementwerts zum Wert des from (oder Summierungs-) Elements addiert werden sollen. Ein Wert `-1.0` gibt an, dass 100% des Werts vom Wert des Summierungselements subtrahiert werden sollen. Wenn Sie auf den Wert des `weight` Attributs doppelklicken, können Sie einen optionalen Wert eingeben.

© 2016-2022 Altova GmbH



Das `weight` Attribut kann auch in der Eingabehilfe "Details" geändert werden (*siehe unten*).

Verbieten der Verwendung eines Arc

Alle Arcs, ob Definition, Presentation oder Calculation, haben ein `use` Attribut, das den Wert `optional` oder `prohibited` erhalten kann. Wenn der Wert `prohibited` verwendet wird, wird der Arc unterbunden.

Farbe und Kontextmenü

Wenn Elemente in der aktuellen Taxonomie erstellt wurden und bearbeitet werden können, werden sie schwarz angezeigt. Andernfalls (wenn sie aus importierten Taxonomien, die bearbeitet werden müssen, stammen), werden Elemente grau angezeigt.

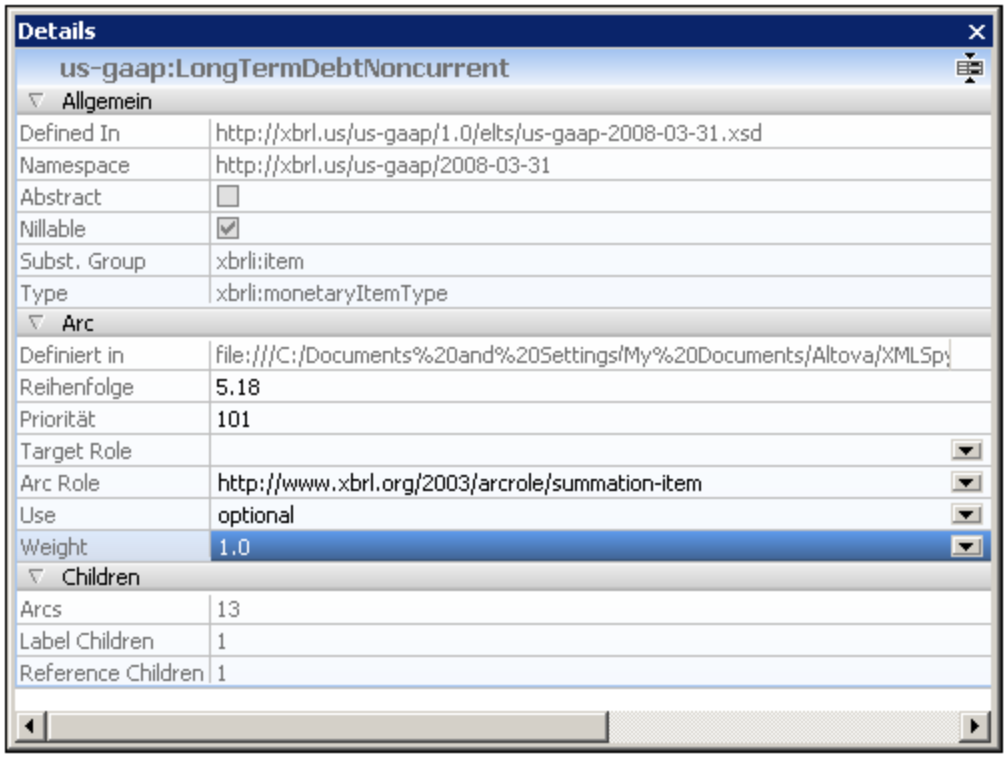
Kontextmenüs im Beziehungsregisterhauptfenster enthalten die folgenden Befehle.

- *Element Reference einfügen:* Steht bei Extended Linkroles zur Verfügung. Fügt unter der Linkrole ein Element hinzu; das hinzugefügte Element befindet sich immer am from Ende von Arcs.
- *Element Reference löschen:* Verfügbar für Element References unmittelbar unter einer Linkrole.
- *Arc einfügen:* Verfügbar bei Elementen. Fügt einen Arc ein und ruft ein Dialogfeld auf, in dem ausgewählt werden kann, welches Element am to Ende des Arc stehen soll.
- *Target Role definieren:* Definiert eine Target Role für das ausgewählte Element.
- *Label Linkrole hinzufügen:* Fügt eine Label Linkrole zum ausgewählten Element hinzu.
- *Reference Linkrole hinzufügen:* Fügt eine Reference Linkrole zum ausgewählten Element hinzu.
- *Arc überschreiben:* Ersetzt den (impliziten) `optional` Wert des `use` Attributs des Arc durch den `prohibited` Wert, so dass der Arc verboten wird.
- *Arc entfernen:* Entfernt den ausgewählten Arc.

- *In globalen Elementen anzeigen:* Markiert das ausgewählte Element in der Eingabehilfe "Globale Elemente"

Eingabehilfe "Details"

Wenn ein Element in einer Beziehung ausgewählt wird, können Arc-Attribute in der Eingabehilfe "Details" bearbeitet werden (*Abbildung unten*).



In der Eingabehilfe "Details" können Attribute, die in der grafischen Anzeigen des Hauptfensters nicht bearbeitet werden können - wie z.B. `order` und `priority` - bearbeitet werden.

© 2016-2022 Altova GmbH

17.3 Zusätzliche Verfahren

Im Abschnitt "Zusätzliche Verfahren" werden verschiedene nützliche Zusatzfunktionalitäten beschrieben:

- Bevorzugte Labels 873
- Typed Domains 875
- Ermittlung und Eliminierung von Duplikaten 876
- Inline XBRL 876

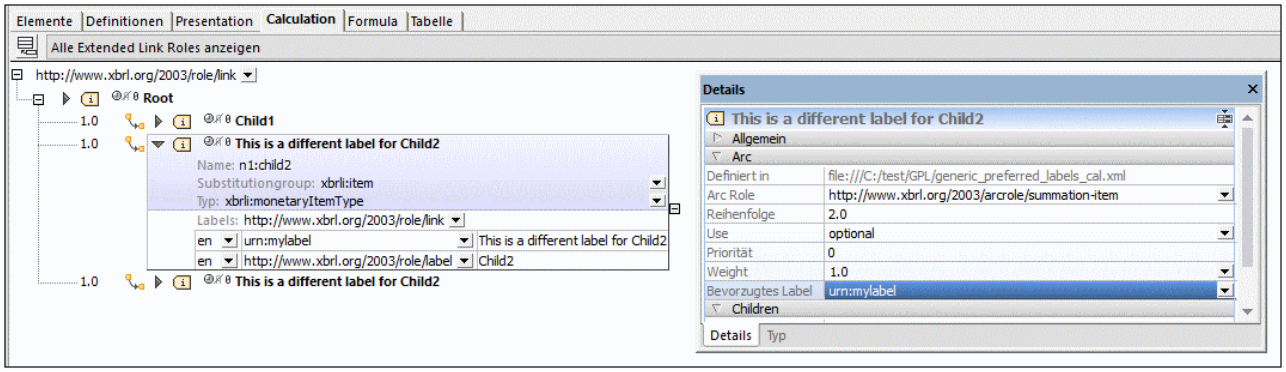
17.3.1 Bevorzugte Labels

Einem Concept oder einer generischen Ressource (Formulas, Tabellen, usw.) können mehrere Labens zugewiesen werden. Das gewünschte Label des Ziel/Child-Node in einem Relationship Arc wird über die Label Role des Labels ausgewählt. Dabei wird folgendermaßen vorgegangen:

- Wenn für den Relationship Arc ein bevorzugtes Label definiert ist, so wird dieses bevorzugte Label verwendet.
- Wenn für eine Beziehung kein bevorzugtes Label definiert wurde, so wird das Standard-Label verwendet.

Durch die Zusatzspezifikation Generic Preferred Label 1.0 Recommendation kann für jeden Arc in Definitionen sowie in Calculation, Formula und Tabellen-Linkbases das Attribut **`gpl:preferredLabel`** definiert werden. Den Wert des Attributs bildet die Label Role, anhand welcher das Label des Ziel-Node ausgewählt werden soll. Aus diesem Grund werden bevorzugte Labels nicht nur für Presentation-Beziehungen (aufgrund früherer Spezifikationen), sondern auch für andere Beziehungen (wie z.B. Calculation-Beziehungen) unterstützt.

Bevorzugte Labels
Bevorzugte Labels werden durch Auswahl der Beziehung auf dem Hauptregister und anschließende Auswahl der gewünschten Label Role als Wert der Eigenschaft *Bevorzugtes Label* (*siehe Abbildung unten*) definiert. Beachten Sie in der Abbildung unten, dass für die zweite Calculation-Beziehung zwei Labels definiert sind. In der Eingabehilfe "Details" wurde als bevorzugtes Label `urn:mylabel` definiert, d.h. dies ist das Label, das als Beschriftung für die Beziehung verwendet wird. (Sie können in den XBRL-Ansichtseinstellungen 1545 von XMLSpy festlegen, dass in der Anzeige von Concepts und/oder Ressourcen anstelle von Namen das Label angezeigt wird.)

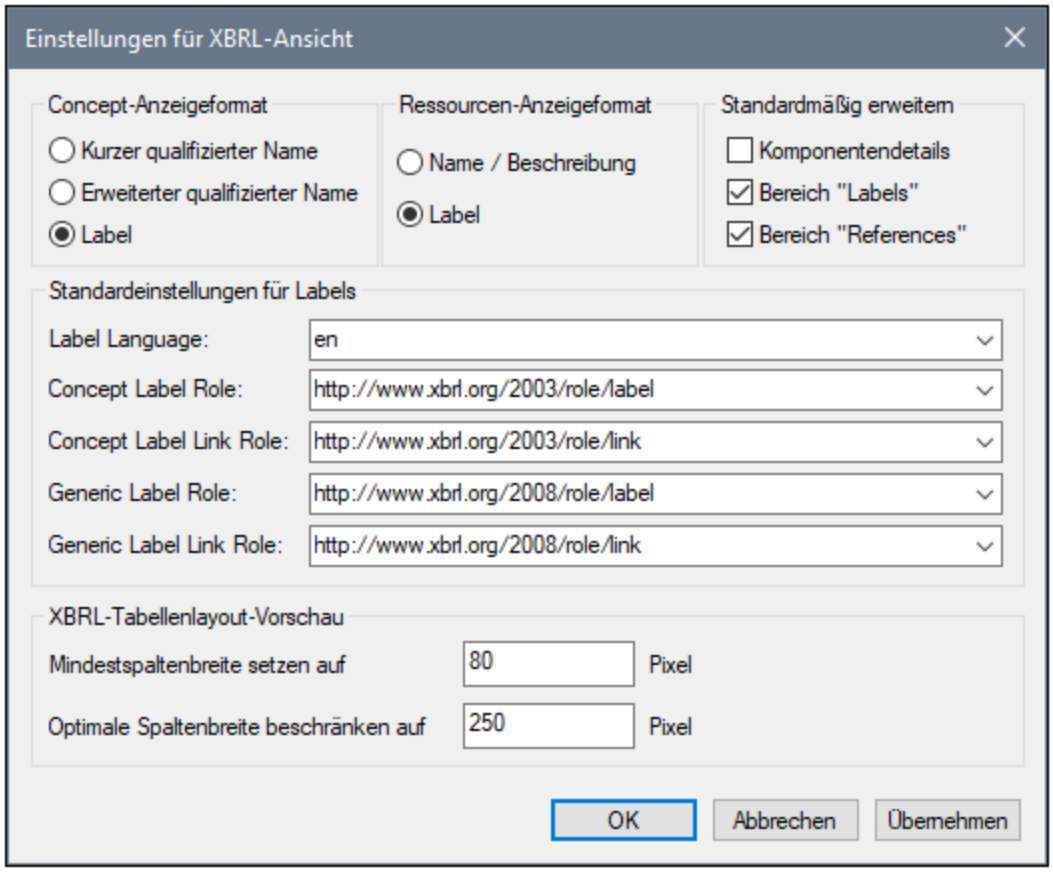


Anmerkung: In Tabellen-Linkbases wird die Eigenschaft *Bevorzugtes Label* für Beziehungs-Nodes, Aspect Nodes und Zusammengeführte Regel-Nodes nicht angezeigt, da für diese Nodes keine Labels definiert werden können.

Standard-Labels
Wenn die Eigenschaft *Bevorzugtes Label* eines Concept oder einer Ressource nicht definiert ist, wird das Standard-Label verwendet. Die Standard-Labels von Concepts und Ressourcen werden im Dialogfeld Einstellungen für XBRL-Ansicht[1545] definiert.

Label-Einstellungen in den XBRL-Ansichtseinstellungen

Das Dialogfeld "Einstellungen für XBRL-Ansicht" (*Abbildung unten*) wird über den Menübefehl **XBRL | Ansichtseinstellungen** aufgerufen.



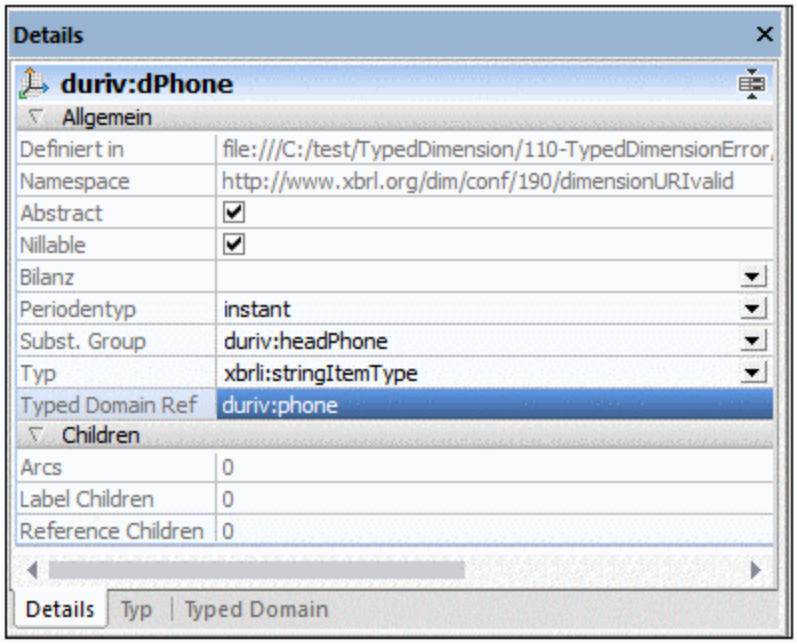
Dieses Dialogfeld enthält zwei für Labels relevante Einstellungen:

- Sie können festlegen, dass Concepts und/oder Ressourcen statt mit ihren Namen mit ihren Labels angezeigt werden.
- Sie können für Concept Lables und Generic Lables (die für Ressourcen verwendet werden) Standardeinstellungen vornehmen.

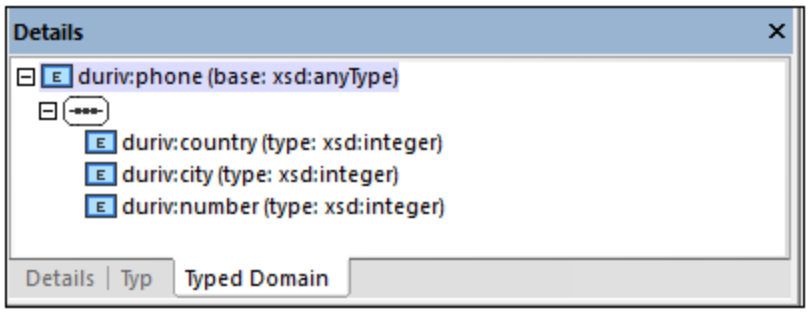
© 2016-2022 Altova GmbH

17.3.2 Typed Domains

Eine Typed Domain ist die von einer Typed Dimension referenzierte Elementdeklaration. So sehen Sie etwa in der Abbildung unten in der Eingabehilfe "Details", dass die Typed Dimension **`duriv:dPhone`** die Typed Domain **`duriv:phone`** referenziert.



Um Informationen über die referenzierte Typed Domain zu sehen, klicken Sie auf das Register "Typed Domain" (*siehe Abbildung unten*). Sie können Items auf diesem Register über die Befehle des Kontextmenüs in der Schema-Ansicht öffnen oder in die Zwischenablage kopieren.



Anmerkung: Das Register "Typed Domain" wird nur dann angezeigt, wenn eine Typed Dimension ausgewählt ist, die eine Typed Domain referenziert.

17.3.3 Ermittlung und Eliminierung von Duplikaten

Zur Behandlung von doppelt vorhandenen Facts in XBRL-Instanzdokumenten stehen die folgenden Funktionalitäten zur Verfügung:

- In XBRL-Instanzdokumenten kann eine Liste doppelt vorhandener Facts ermittelt und im Fenster "Meldungen" angezeigt werden. Verwenden Sie dazu den Menübefehl **XBRL | Duplikate ermitteln** oder **XBRL | Duplikate am Server ermitteln (Hochleistungsausführung)**. Nähere Informationen dazu finden Sie unter der Beschreibung zu den Befehlen. 1551
- Wenn Sie ein XBRL-Instanzdokument validieren, können Sie festlegen, dass doppelte Facts, die sich auf Berechnungen auswirken, bei der Validierung ignoriert werden sollen. Sie können diese Einstellung in den XBRL-Validierungsoptionen 1639 vornehmen.
- Bei der Ausführung von Formulas, der Generierung von Tabellen oder der Transformation von Inline XBRL können vorhandene Duplikate ignoriert werden. Dies kann im Dialogfeld XBRL-Verarbeitungsoptionen 1555 definiert werden.

Die Ermittlung von Duplikaten erfolgt auf Basis der Regeln in der Spezifikation Handling Duplicate Facts in XBRL and Inline XBRL 1.0.

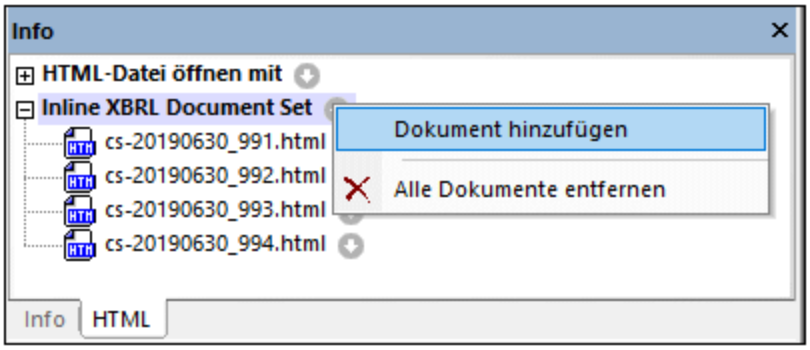
17.3.4 Inline XBRL

Inline XBRL-Dokumente sind HTML-Dokumente, die XBRL-Daten enthalten (d.h. Daten, die mit XBRL-Markup versehen sind). Sie können das aktive Inline XBRL-Dokument (mit dem Befehl **XML | Validieren**) validieren und (mit dem Befehl **XBRL | Inline XBRL transformieren**) verarbeiten. Bei der Verarbeitung des Dokuments werden die XBRL-Daten aus dem HTML-Dokument extrahiert.

Mehrere Inline XBRL-Dokumente

Sie können auch mehrere Inline XBRL-Dokumente validieren/verarbeiten. Gehen Sie dazu folgendermaßen vor:

1. Öffnen Sie das Inline XBRL-Hauptdokument, sodass es das aktive Dokument ist. Im Fenster "Info" 120 wird das Register "HTML" angezeigt *(siehe Abbildung unten)*.



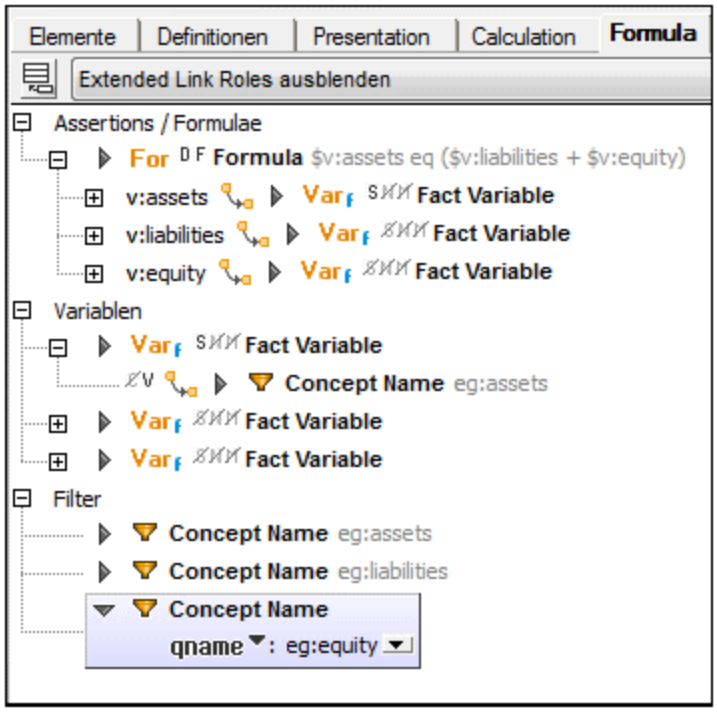
2. Klicken Sie auf dem Register "HTML" auf die Menüschaltfläche von *Inline XBRL Document Set* und anschließend auf **Dokument hinzufügen** *(siehe Abbildung)* und navigieren Sie zu den Inline XBRL-Dateien, die Sie hinzufügen möchten..
3. Führen Sie den Validierungs- oder Verarbeitungsbefehl aus.

© 2016-2022 Altova GmbH

17.4 XBRL Formula Editor

Die XBRL Formula-, Variable- und Filter-Spezifikation stellt eine Syntax zur Verfügung, mit der Regeln zum Ableiten neuer Fact-Werte anhand der Daten in XBRL-Geschäftsberichten ausgedrückt werden können. Die allgemeinen Label- und Reference-Spezifikationen unterstützen die Bezeichnung aller Arten verschiedener XBRL-Konstrukte. Im Zusammenhang mit XBRL Formula kann mit Hilfe dieser Bezeichnungen und Referenzierungen manuell verfasste Dokumentation mit Formeln, ihren Variablen und den Filtern verknüpft werden, die festlegen, welche Facts in einem XBRL-Geschäftsbericht durch eine Variable für die Verwendung in der Auswertung einer Formula ausgewählt werden. Die Validierungsspezifikation und die drei Assertion-Spezifikationen definieren eine Syntax zum Ausdrücken von Regeln zum erwarteten Inhalt von Geschäftsberichten. Diese werden in Form von Variablen, Variablengruppen und Formeln ausgedrückt. Im Arbeitsentwurf von XBRL Formula Overview 1.0 finden Sie eine Einführung in die Syntax und Semantik von XBRL Formula.

Der XBRL Formula Editor von XMLSpy wurde im Rahmen des XBRL-Taxonomie-Editors der Applikation implementiert. Er steht in der XBRL-Ansicht auf dem Register "Formula" zur Verfügung (*siehe Abbildung unten*).



Das Register "Formula" wird zusammen mit den Eingabehilfen "Übersicht" und "Details" verwendet und dient zum Erstellen und Bearbeiten von Formeln. In der Eingabehilfe "Übersicht" wird die Standard-Linkbase für XBRL Formulas festgelegt (die Datei, in der die Formeln standardmäßig gespeichert werden), während in der Eingabehilfe "Details" die Eigenschaften und der Inhalt der Formula-Komponenten bearbeitet werden können. (Diese Bearbeitung kann allerdings auch direkt auf dem Register "Formula" durchgeführt werden).

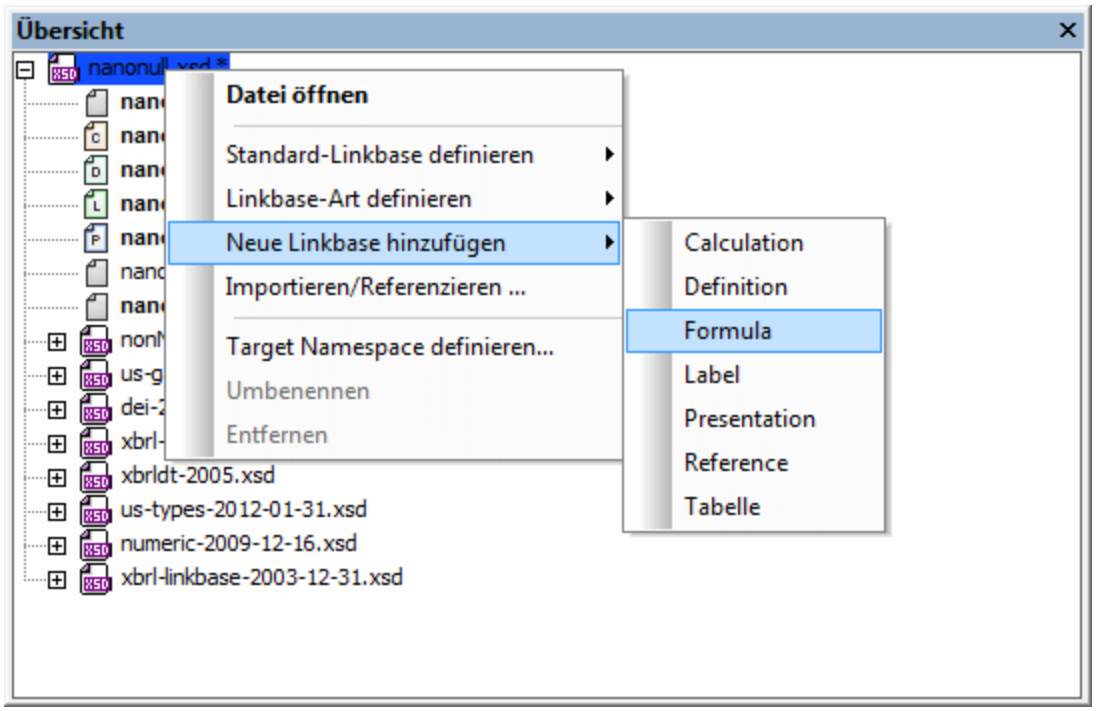
© 2016-2022 Altova GmbH

17.4.1 Formula Linkbases und Link Roles

Während Standard XBRL Linkbases (Definitions, Presentations, Calculations) über Locators und Standard Arcs in Standard Extended Links Beziehungen zwischen Concepts definieren, definiert eine Formula Linkbase Formula-Komponenten (Formulae, Variablen, Filter, Assertions usw.) und ihre Beziehungen. Diese Definitionen sind in allgemeinen Extended Links mittels Ressourcen und allgemeine Arcs definiert.

Hinzufügen einer Formula Linkbase

Klicken Sie in der Eingabehilfe "Übersicht" (*Abbildung unten*) mit der rechten Maustaste auf die Taxonomiedatei oder eine vorhandene Linkbase und wählen Sie den Befehl **Neue Linkbase hinzufügen | Formula**. Die hinzugefügte Linkbase wird die Standard Formula Linkbase-Datei. Die Standard Formula Linkbase-Datei ist die Datei, in der beim Speichern der Taxonomie-Datei neue Formula-Definitionen gespeichert werden. Wenn Sie eine andere Formula Linkbase-Datei zur Standard Formula Linkbase machen möchten, klicken Sie mit der rechten Maustaste darauf und wählen Sie den Befehl **Standard-Linkbase definieren | Formula** (*siehe Abbildung unten*).



Beachten Sie, dass Standard-Linkbases fett formatiert angezeigt werden und dass Linkbases, die geändert, aber noch nicht gespeichert wurden, mit einem Sternchen markiert sind.

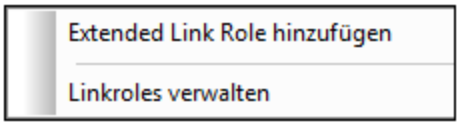
Die Formula Linkbase wird auf dem Registger "Formula" angezeigt.

Anmerkung: Wenn zu einem Zeitpunkt, zu dem in der Taxonomie noch keine Formula Linkbase vorhanden ist, eine Formula-Komponente hinzugefügt wird[880], wird automatisch eine Formula Linkbase erstellt.

Link Roles

Wie bei Standard Extended Links (für Definitions, Presentations, Calculations) müssen allgemeine Links einen Extended Link-Wert definieren, der Beziehungen desselben Typs in separate Netzwerke aufteilt. Alle allgemeinen Extended Links mit derselben Link Role werden im Diagramm auf dem Register "Formula" unter ein und demselben Link Role Node kombiniert, *selbst wenn sie sich in verschiedenen Linkbase-Dateien befinden*.

Allgemeine Link Roles können im Diagramm über das Kontextmenü (Rechtsklick auf den leeren Bereich) (*Abbildung unten*) erstellt werden. Beachten Sie allerdings, dass dieses Kontextmenü nur angezeigt wird, wenn in der Auswahlliste der Ansichtsoptionen des Registers "Formula" die Option *Alle Extended Link Roles anzeigen* ausgewählt wurde.



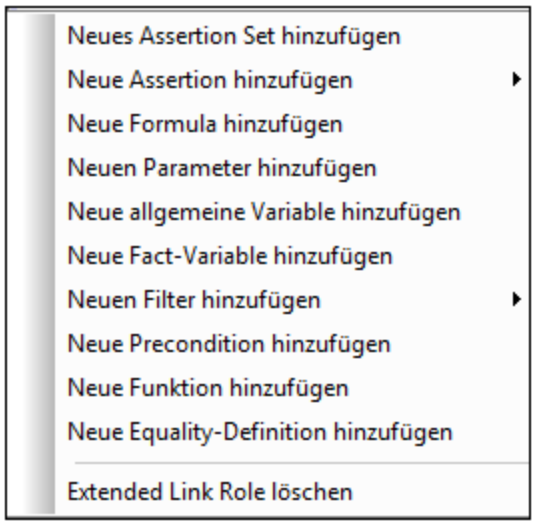
Dieses Menü steht auch über die Symbolleisten-Schaltfläche  *Extended Link Role hinzufügen / Linkroles verwalten* zur Verfügung. Da Beziehungsnetzwerke bei einer Formula Linkbase keine große Rolle spielen, ist die Standardansicht auf dem Register "Formula"*Extended Link Roles ausblenden.* Mit dieser Option werden die Link Roles ausgeblendet. Statt dessen werden die Formula-Komponenten ohne ihre Link Roles angezeigt.

Wenn bei der Erstellung der Extended Link Role keine Standard-Formula Linkbase-Datei vorhanden ist, wird automatisch eine Standard-Formula Linkbase-Datei erstellt und wenn bei der Erstellung der Link Role in der Standard Linkbase-Datei keine Link Role vorhanden ist, so wird automatisch eine Link Role in der Standard-Linkbase-Datei erstellt.

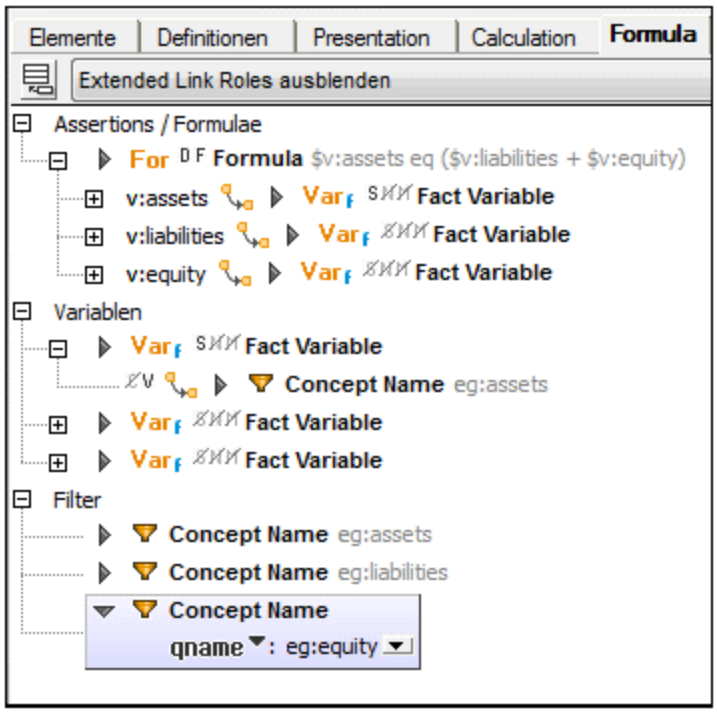
17.4.2 Formula-Komponenten

Neue Formula-Komponenten werden über das Kontextmenü eines Link Role Node (*Abbildung unten*) oder bei ausgewählter Ansichtsoption *Extended Link Roles ausblenden* über die Symbolleisten-Schaltfläche  *Neue Formula-Komponente hinzufügen* erstellt.

© 2016-2022 Altova GmbH



Eine genauere Beschreibung zum Hinzufügen einzelner Komponenten finden Sie in den Unterabschnitten dieses Abschnitts. Nachdem eine Formula-Komponente hinzugefügt wurde, wird sie im Diagramm auf dem Register "Formula" (*siehe Abbildung unten*) angezeigt.



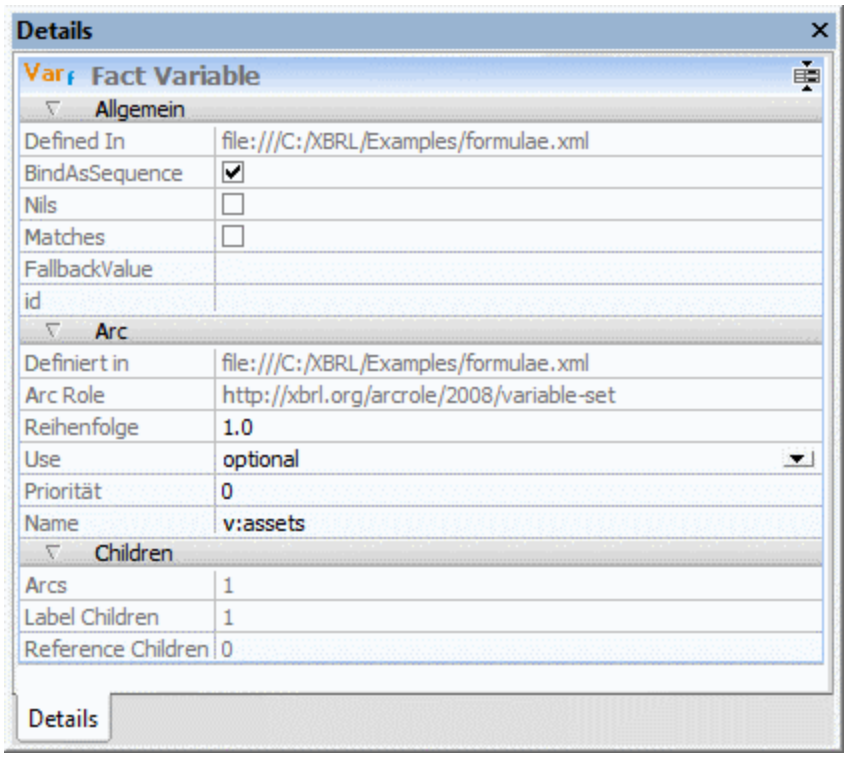
Aus Gründen der Übersichtlichkeit sind Formula-Komponenten in Abschnitte gegliedert, wobei die Beziehungen zu anderen Komponenten (die Arcs) in einer Baumstruktur angezeigt werden (*siehe Abbildung oben*).

Die Eigenschaften von Komponenten und Beziehungen (Arcs) werden im Diagramm als Symbole links von der Komponente bzw. dem Arc angezeigt (*siehe Abbildung unten*).



So hat z.B. die Fact Variable-Komponente in der Abbildung oben drei Eigenschaften, `BindAsSequence` (gekennzeichnet durch das Symbol `S`), `Nils` (Symbol `N`) und `Matches` (Symbol `M`). Es handelt sich bei allen drei Eigenschaften um Boolesche Eigenschaften. Die erste (`BindAsSequence`) hat den Wert `true`. Im Diagramm ist dies daran zu erkennen, dass das Symbol nicht durchgestrichen ist. Die anderen beiden Eigenschaften haben den Wert `false` (gekennzeichnet durch ein durchgestrichenes Symbol). Der Arc (unterhalb der Variablen) hat zwei Eigenschaften: die erste ist der Boolesche Wert `false`, die zweite der Boolesche Wert `true`.

In der Eingabehilfe "Details" der Fact Variable (*Abbildung unten*) sind die Eigenschaften der Variablen unter dem Abschnitt *Allgemein* aufgelistet. Die Werte von Booleschen Eigenschaften sind durch ein Häkchen für `true` und kein Häkchen für `false` gekennzeichnet.



Um die Eigenschaften eines Arc in der Eingabehilfe "Details" zu sehen, wählen Sie im Diagramm die `to` (Ziel)-Komponente aus; die Eigenschaften des Arc werden daraufhin im Abschnitt *Arc* aufgelistet.

Kontextmenüs im Formula Editor

Welches Kontextmenü für Formula-Komponenten angezeigt wird, hängt vom Typ der Komponente ab. Die Menüeinträge sind die die folgenden Abschnitte gegliedert:

- Änderung von Inhalt (bei Formulas, einigen Filtern, benutzerdefinierten Funktionen): z.B., *Aspect-Regel anhängen/einfügen.*
- Änderungen von Beziehungen (nur bei Subeinträgen): *Arc überschreiben/entfernen*

© 2016-2022 Altova GmbH

- *Labels/References hinzufügen*
- Erstellung neuer Child-Komponenten (einschließlich Beziehungen): z.B. *Neuen Filter hinzufügen*
- Löschen einer Komponente (einschließlich Beziehungen)
- *Weitersuchen/Vorheriges suchen* (Komponenten)

Anmerkung: Inhaltselemente, die über das Kontextmenü erstellt oder entfernt werden können, werden in der Eingabehilfe "Details" in zusätzlichen Abschnitten wie z.B. *Concept Aspect-Regel* angezeigt.

17.4.2.1 Assertions und Assertion Sets

Es gibt drei Arten von Assertions:

- Value Assertions
- Existence Assertions
- Consistency Assertions

Value Assertions

Value Assertions sind das am häufigsten verwendete Formula Linkbase Feature. Sie bieten die Möglichkeit, Facts einer XBRL-Input-Instanz anhand eines XPath-Ausdrucks zu überprüfen. Dazu stehen die Eigenschaften *Aspect Model* und *Implicit Filtering* in Form von Symbolen zur Verfügung. Der Wert der Eigenschaft `test` ist ein XPath-Ausdruck.

Existence Assertions

Mit einer Existence Assertion wird das statische Vorhandensein überprüft, z.B. um sicherzustellen, dass die beschreibenden Facts eines Dokuments wie z.B. Form Type, Company Identification und Filing Identification vorhanden sind. Dazu stehen die Eigenschaften *Aspect Model* und *Implicit Filtering* als Symbole zur Verfügung. Der Wert der Eigenschaft `test` ist ein XPath-Ausdruck.

Consistency Assertions

Mit einer Consistency Assertion wird definiert, wie festgestellt wird, ob ein mit der verknüpften Formula erzeugtes Output Fact in der XBRL-Input-Instanz mit allen Facts konsistent ist, die mit dem Aspect übereinstimmen. Dazu steht die Boolesche Eigenschaft `strict` als Symbol zur Verfügung. Die Werte der Eigenschaften *Absolute Acceptance Radius* und *Proportional Acceptance Radius* sind XPath-Ausdrücke.

Assertion erfüllt/nicht erfüllt-Meldungen

Diese Unterkomponenten von Assertions ermöglichen das Verknüpfen von Meldungen mit Auswertungen von Assertions: bei erfolgreicher Auswertung die Meldung "Erfüllt", bei fehlgeschlagener Auswertung die Meldung "nicht erfüllt". Diese Meldungen können über das Kontextmenü der einzelnen Assertions hinzugefügt werden.

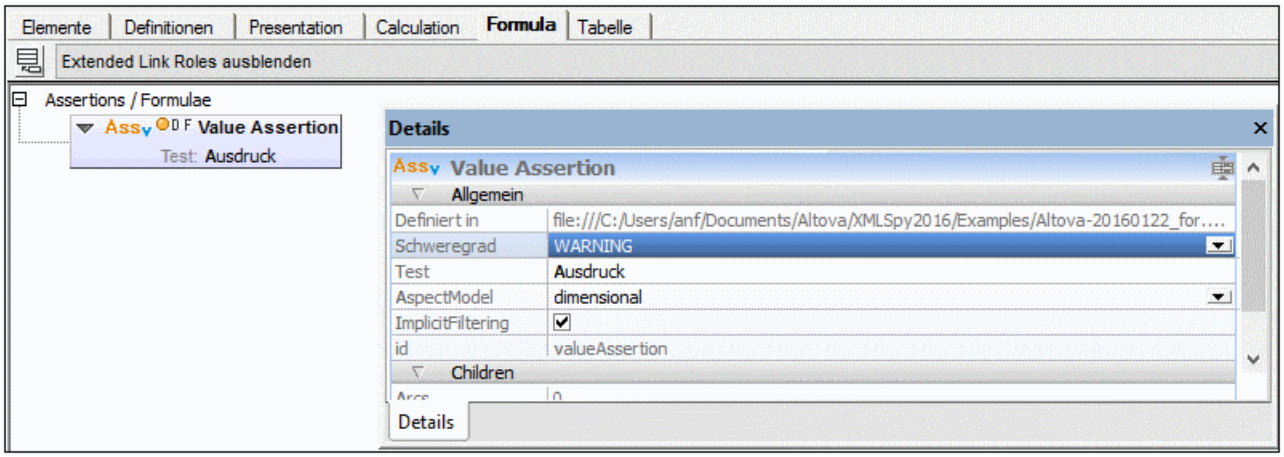
Assertion-Unsatisfied-Severity-Beziehungen

Eine Assertion wird entweder erfüllt (satisfied) oder nicht erfüllt (unsatisfied). Da es für Assertions jedoch Regeln unterschiedlicher Prioritätsstufe gibt, werden nicht erfüllte Assertions nach dem Schweregrad (Severity) dieser Nichterfüllung eingeteilt. Es gibt drei Standardschweregrade: `ERROR`, `WARNING` und `OK`. Der

Standardschweregrad ist `ERROR`. Er wird aufgerufen, wenn eine Assertion nicht mit einem bestimmten Schweregrad verknüpft ist.

Die Assertion-Unsatisfied-Severity-Beziehung besteht zwischen einer Assertion und einer der definierten Schweregradressourcen. Sie wird durch einen XLink Ark mit: (i) einem Arcrole-Wert von `http://xbrl.org/arcrole/PR/2015-11-180/assertion-unsatisfied-severity`, (ii) einer Assertion als ihrer Ausgangsressource und (iii) einer Schweregradressource als Endressource definiert.

Die Schweregradbeziehung kann im Taxonomie-Editor durch Klick auf das `Schweregrad`-Symbol der Assertion-Komponente im Diagramm (siehe Abbildung unten) und anschließende Auswahl des Schweregrads aus dem Popup-Menü definiert werden. Alternativ dazu kann der Schweregrad in der Eingabehilfe "Details" der Assertion ausgewählt werden (*siehe Abbildung*).



Assertion Sets

Ein Assertion Set enthält eine oder mehrere Assertions. Über das Kontextmenü eines Assertion Set können einzelne Assertions zum Assertion Set hinzugefügt werden.

17.4.2.2 Formulas

Mit einer Formula wird eine Gruppe von Regeln für die Konstruktion eines XBRL-Output Fact durch Transformation der Werte, zu denen die Variablen in der Variablengruppe der Formula ausgewerten wurden, ausgedrückt. Die Werte der Variablen stammen aus einer XBRL-Input-Instanz und ihrem zugrunde liegenden DTS oder aus der Applikation, die die Formula verarbeitet.

Die Value-Regel ist ein XPath-Ausdruck, dessen Ergebnis der Wert ist, der dem Fact zugewiesen werden soll. Die Value-Regel kann ein einfacher Ausdruck wie z.B. eine Konstante sein oder Ausdrücke enthalten, die sich auf Variablen und Parameter der Variablengruppe, verkettete Werte aus anderen Variablengruppen und/oder berechnete Werte aus benutzerdefinierten und vordefinierten Funktionen beziehen.

In XBRL werden ganzzahlige numerische Facts mit Informationen über ihre Präzision in Form eines precision/decimals-Attributs ausgegeben. Daher können Formulae Accuracy-Regeln enthalten, mit denen festgelegt wird, wie die Genauigkeit bestimmt wird, die für ein Output Fact behauptet werden soll.

© 2016-2022 Altova GmbH

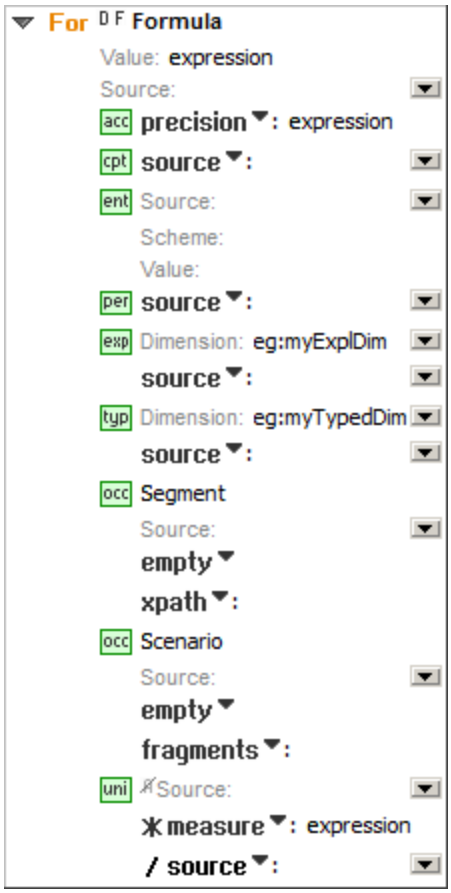
Neben Regeln zum Ermitteln von Output Fact-Werten und deren Präzision werden mit Formulae Aspect-Regeln definiert oder impliziert, die Werte für alle für die Interpretation der Output-Werte erforderlichen Output Aspects ermitteln. Regeln zum Festlegen des Output Concept, des Output-Kontexts und der Output-Maßeinheiten (für numerische Facts) sind alle unterschiedliche Typen von Aspect-Regeln.

Ein Aspekt kann durch Angabe einer Quelle (teilweise oder zur Gänze) aus einer gebundenen Variablen der Auswertung stammen. Die Quelle kann in einer Regel definiert werden oder aus einer Quelle im Formula (oder Tuple) Element geerbt werden. Bei Vorhandensein mehrerer Quellen hat die der Aspect-Regel nächstgelegene Vorrang.

Wenn eine Formula eingefügt wird, hat sie keine Accuracy oder Aspect-Regel (*siehe Abbilung unten*).



Accuracy- und Aspect-Regeln werden im Inhalt der Formula definiert und über das Kontextmenü hinzugefügt (oder entfernt). In der Abbildung unten sehen Sie eine Formula mit allen möglichen Accuracy und Aspect-Regeln.



In der Eingabehilfe "Details" werden Accuracy und Aspect-Regeln in zusätzlichen Abschnitten angezeigt.

Accuracy-Regel

Art: precision oder decimals
Wert: XPath-Ausdruck

Aspect-Regeln

Aspect-Regeln werden nach Art gruppiert.

Concept-Regeln
Art: qname, expr oder source
Wert: QName des Concept, XPath-Ausdruck oder Quellvariable (oder der freigelegte QName)

Entity Identifier-Regeln
Quelle: Quellvariable (oder der freigelegte QName)
Scheme/Walue: XPath-Ausdruck

Zeitraum-Regeln
Art: instant, duration, forever oder source
Wert: XPath-Ausdruck des Werts, Beginn/Ende/Quelle, kein Wert oder Quellvariable (oder der freigelegte QName)

Explicit Dimension-Regeln
Dimension: QName der Dimension, wird durch die Explicit Dimension Rule beeinflusst.
Art: qname, exp, omit oder source
Wert: QName des Mitglieds, XPath-Ausdruck des Mitglieds, kein Wert oder Quellvariable (oder der freigelegte QName)

Typed Dimension-Regeln
Dimension: QName der Dimension, wird durch die Typed Dimension Rule beeinflusst.
Art: xpath, value, omit oder source
Wert: XPath-Ausdruck, XML-Element, kein Wert oder oder Quellvariable (oder der freigelegte QName)

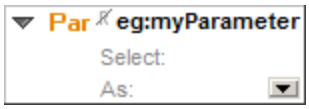
Open Context Component-Regeln
OCC-Regeln sind nach Art gruppiert, d.h. nach Segment OCC-Regeln und Scenario OCC-Regeln.
Quelle: in der ersten OCC Rule definierte Quellvariable.
Für jede OCC Rule:
Art: empty, fragments oder xpath
Wert: Kein Wert, XML-Elemente oder XPath-Ausdruck

Unit-Regeln
Das Boolesche Flag *Augment* definiert, ob der Aspect-Quellwert verwendet werden muss oder nicht.
Für jede Unit Multiplication/Division Rule:
Art: *measure, /measure oder *source, /source
Wert: XPath-Ausdruck der Maßeinheit oder Quellvariable (oder der freigelegte QName)

© 2016-2022 Altova GmbH

17.4.2.3 Parameter

Ein Parameter kann in XPath-Ausdrücken referenziert werden. Er stellt ein *Required* Flag zur Verfügung. Ist dieses Flag gesetzt, ist der Parameter obligatorisch, d.h. sein Wert muss von der verarbeitenden Applikation zur Verfügung gestellt werden. Ist der Parameter nicht obligatorisch und wird von der verarbeitenden Applikation kein Wert bereitgestellt, kann der bereitgestellte Wert mit Hilfe des in der *Select* Eigenschaft angegebenen XPath-Ausdrucks berechnet werden. Die optionale Eigenschaft *As* definiert den für den Parameter erforderlichen Datentyp.

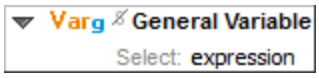


17.4.2.4 Variablen

Mit Variablen wird eine Methode deklariert, wie Input-Daten, im Allgemeinen Fact Items, an einen Namen gebunden werden, der von einem Variablennamen wie z.B. von einer Assertion oder einem Formula-Ausdruck aus referenziert werden kann. Variablen, die an Input Fact Items gebunden werden, sind Fact-Variablen und verwenden Filter, um zu deklarieren, woran sie im Input gebunden werden können. Allgemeine Variablen werden für Zwischenergebnisse von Ausdrücken und andere Arten der Verarbeitung verwendet.

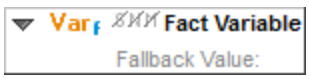
Allgemeine Variable

Eine allgemeine Variable stellt die Boolesche Eigenschaft "Bind As Sequence" als Symbol bereit. Der Wert der Eigenschaft "Select" ist ein XPath-Ausdruck.



Fact Variable

Eine Fact Variable stellt die Booleschen Eigenschaften "Bind As Sequence", "Nils" und "Matches" als Symbole zur Verfügung. Der Wert der Eigenschaft "Fallback Value" ist ein XPath-Ausdruck.



17.4.2.5 Filter

Filter dienen zur Definition von Auswahlkriterien für Facts in der XBRL-Input-Instanz, also der XBRL-Instanz, anhand welcher Variablen ausgewertet werden. Filter drücken Kriterien aus, die auf Input Facts angewendet werden können. Bei einigen Filtern können in Unterzeilen XML-Inhalte angezeigt werden.

Aspect Cover
Diese Filter führen keine "Filterung" durch und haben daher keinen impliziten XPath-Ausdruck. Sie werden nach anderen Filtern (wie z.B. Concept und Dimension) verarbeitet oder angewendet und setzen den Cover-Status von Aspects, die aus der Anwendung der anderen Filter resultieren, außer Kraft.

Ein oder mehrere Aspect Items
Art: aspect, dim-qname/excl-dim-qname oder dim-exp/excl-dim-exp
Wert: Aspect-Art (enum), QName oder XPath-Ausdruck der Dimension
Die Items werden in der Eingabehilfe "Details" in zusätzlichen Abschnitten angezeigt.

Boolesche Filter
Boolesche Filter stehen mit Sub-Filtern in Zusammenhang.
Der and-Filter findet Facts auf Basis der Kriterien, die in allen seinen Sub-Filtern ausgedrückt sind.



Der or-Filter findet Facts auf Basis von Kriterien, die in einem beliebigen seiner Sub-Filter ausgedrückt sind.

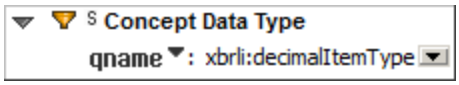


Concept Name
Der Concept Name-Filter findet Facts auf Basis der Namen ihrer Concepts.



Ein oder mehrere Concepts:
Art: qname oder exp
Wert: QName oder XPath-Ausdruck des Concept
Die Concepts werden in der Eingabehilfe "Details" in zusätzlichen Abschnitten angezeigt.

Concept Data Type
Der Concept Data-Type-Filter dient zum Finden von Facts auf Basis ihres XML-Schema-Datentyps.



Das Boolesche Flag: “strict” definiert, ob es sich beim Datentyp des Fact um einen nicht abgeleiteten handeln muss oder nicht.
Art: qname oder exp
Wert: QName oder XPath-Ausdruck des Datentyps

© 2016-2022 Altova GmbH

Concept Substitution Group
Der Concept Substitution-Group-Filter dient zum Finden von Facts auf Basis Ihrer XML-Schema-Ersetzungsgruppe.

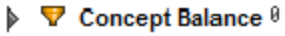


Das Boolesche Flag: “strict” legt fest, ob das Concept des Fact das Element in seinem @substitutionGroup-Attribut direkt definieren muss oder nicht.
Art: qname oder exp
Wert: QName oder XPath-Ausdruck der Ersetzungsgruppe

Concept Period Type
Der Concept Period-Type-Filter dient zum Finden von Facts auf Basis dessen, ob Facts Werte für Concepts vom Typ Duration oder Instant ausgeben. Dies ist im Attribut @xbrli:periodType festgelegt.



Concept Balance
Der Concept Balance-Filter dient zum Finden von Facts auf Basis dessen, ob diese ein @xbrli:balance Attribut haben und ob dieses Attribut den Wert "debit" oder "credit" hat.

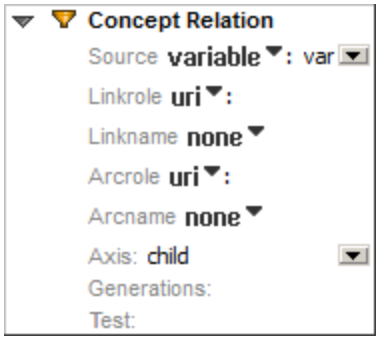


Concept Custom Attribute
Der Concept Custom-Attribute-Filter dient zum Finden von Facts auf Basis des Vorhandenseins oder Werts eines custom-Attributs in der Deklaration der einzelnen Concepts.



Art: qname oder exp
Wert: QName oder XPath-Ausdruck des Attributs

Concept Relation
Der Concept Relation-Filter dient zum Finden von Facts auf Basis der effektiven Beziehungen ihrer Concepts zum Quell-Concept, in einem definierten Linkrole URI-Netzwerk effektiver Beziehungen, einer angegebenen Arcrole URI, in einer bestimmten Achse, einschließlich definierter Generierungen und in Übereinstimmung mit einem optionalen Testausdruck.



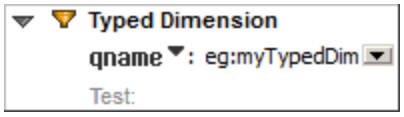
Quelle: Art = variable, qname oder exp
Linkrole: Art = uri oder exp
Linkname: Art = none, qname oder exp
Arcrole: Art = uri oder exp
Arcname: Art = none, qname oder exp

Explicit Dimension
Eine Explicit Dimension Domain ist im Kontext eines angegebenen DTS als die Gruppe aller Domain-Mitglieder in der Vereinigung aller Domains gültiger Mitglieder der Filter Dimension definiert. Mit dem Explicit Dimension Filter werden Facts gefunden, die mit einem beliebigen der Domain-Mitglieder in einer Explicit Dimension Domain als Wert für diese Explicit Dimension übereinstimmen.



Dimension-Art: qname oder exp
Ein oder mehrere Mitglieder:
Art: variable, qname oder exp
Die Mitglieder werden in der Eingabehilfe "Details" in zusätzlichen Abschnitten angezeigt.

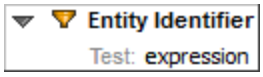
Typed Dimension
Der Typed Dimension-Filter dient zum Finden von Facts auf Basis des Werts für eine Typed Dimension.



Dimension-Art: qname oder exp

© 2016-2022 Altova GmbH

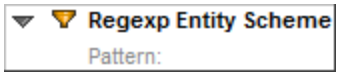
Entity Identifier Filter
Der Entity Identifier Filter dient zum Finden von Facts auf Basis der Charakteristiken des Entity Identification-Schemas und/oder des Entity Identification-Werts.



Specific Entity Scheme
Der Specific Entity-Scheme-Filter dient zum Finden von Facts auf Basis dessen, ob diese Facts Werte für das vom Filter identifizierte Schema ausgeben.



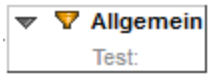
Regular Expression Entity Scheme
Der Regular-Expression Entity-Scheme-Filter dient zum Finden von Facts auf Basis von Mustern von regulären Ausdrücken im Text des Entity-Schemas.



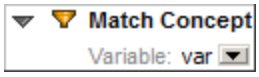
Specific Entity Identifier
Der Specific Entity-Identifier-Filter dient zum Finden von Facts auf Basis dessen, ob diese Facts Werte ausgeben, die den vom Filter gelieferten Entity Identifier-Wert verwenden.

Regular Expression Entity Identifier
Der Regular Expression Entity-Identifier-Filter dient zum Finden von Facts auf Basis von Regular Expression-Mustern im Text des Entity Identifier-Werts.

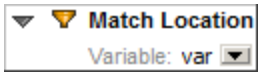
General
Der General-Filter deckt keine Aspects ab.



Match Concept
Der Concept Matching-Filter dient zum Auswählen von Facts, die Werte für dasselbe Concept ausgeben.



Match Location
Der Location Matching-Filter dient zum Auswählen von Facts, die dasselbe Parent-Element haben.



Match Unit
Der Unit Matching-Filter dient zum Finden von Facts, die dieselbe Unit haben.

Match Entity Identifier
Der Entity-Identifier Matching-Filter dient zum Auswählen von Facts mit demselben Entity Identifier.

Match Period
Der Period Matching-Filter dient zum Auswählen von Facts, die denselben Zeitraum haben.

Match Dimension
Der Dimension Matching-Filter dient zum Auswählen von Facts, die denselben Wert für eine bestimmte XBRL Dimension haben.



Match Complete Segment
Der Complete-Segment Matching-Filter dient zum Auswählen von Facts, die dasselbe Segment haben, dessen Inhalt nicht auf Basis der XBRL Dimensions-Spezifikation interpretiert wird.

Match Non-XDT Segment
Der Non-XDT Segment-Matching-Filter dient zum Auswählen von Facts, die dasselbe Segment haben, nachdem alle Inhalte der XBRL Dimensions-Spezifikation vom Vergleich ausgeschlossen wurden.

Match Complete Scenario
Der Complete-Scenario Matching-Filter dient zum Auswählen von Facts, die dasselbe Scenario haben, wobei das Scenario nicht auf Basis der XBRL Dimensions Specification interpretiert wird.

Match Non-XDT Scenario
Der non-XDT Scenario Matching-Filter dient zum Auswählen von Facts, die dasselbe Scenario haben, nachdem alle Inhalte der XBRL Dimensions-Spezifikation vom Vergleich ausgeschlossen wurden.

© 2016-2022 Altova GmbH

Zeitraum
Der Zeitraum-Filter dient zum Finden von Facts auf Basis einer breiten Palette von Kriterien im Zusammenhang mit dem Zeitraum, über den oder in dem diese Facts gemessen wurden.



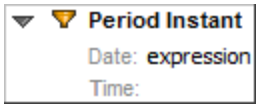
Zeitraumbeginn
Der Zeitraumbeginn-Filter dient zum Finden von Facts auf Basis des Beginns des Zeitraums, in dem diese Facts gemessen wurden.



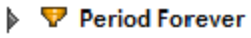
Zeitraumende
Der Zeitraumende-Filter dient zum Finden von Facts auf Basis des Endes des Zeitraums, in dem diese Facts gemessen wurden.



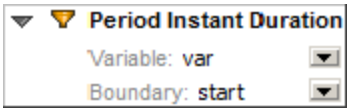
Period Instant
Der Period-Instant-Filter dient zum Finden von Facts auf Basis des Zeitpunkts, zu dem sie gemessen wurden.



Period Forever
Der Forever-Filter dient zum Finden von Facts, die mit einem Forever-Zeitraum ausgegeben werden.



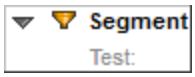
Period Instant Duration
Der Instant-Duration-Filter dient zum Finden von Facts, die zu einem Zeitpunkt stattfinden, der mit dem Beginn oder dem Ende der Zeitdauer eines anderen Fact übereinstimmt.



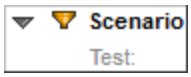
Relative
Der Relative-Filter dient zum Auswählen von Facts, deren vom Relative-Filter erfasste Aspects Werte haben, die mit den entsprechenden Aspects eines anderen Fact übereinstimmen. Das vom relative-Filter gefundene Fact muss das Auswertungsergebnis einer anderen Fact-Variablen in der ausgewerteten Variablengruppe sein.



Segment
Der Segment-Filter dient zum Finden von Facts, die Nicht-XDT-Inhalt haben, der die definierten Constraints erfüllt. Nicht-XDT-Inhalt bezieht sich auf Segmentinhalt, der nicht auf den in der XBRL Dimensions-Spezifikation definierten Explicit oder Typed Dimensions basiert.



Scenario
Der Scenario-Filter dient zum Finden von Facts, die Nicht-XDT-Inhalt haben, der die definierten Constraints erfüllt. Nicht-XDT-Inhalt bezieht sich auf Scenario-Inhalt, der nicht auf den in der XBRL Dimensions-Spezifikation definierten Explicit oder Typed Dimensions basiert

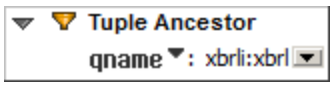


Tuple Parent
Der Parent-Filter dient zum Auswählen von Facts, die ein bestimmtes Parent-Element haben.



Art: qname oder exp

Tuple Ancestor
Der Ancestor-Filter dient zum Auswählen von Facts, die ein bestimmtes Vorfahren-Element haben.



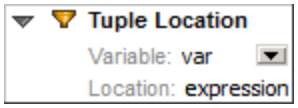
Art: qname oder exp

Tuple Sibling
Der Sibling-Filter dient zum Auswählen von Facts, die gleichrangige Facts eines anderen Fact sind.

© 2016-2022 Altova GmbH



Tuple Location
Der Location-Filter dient zum Auswählen von Facts, die eine bestimmte Position relativ zur Position eines anderen Fact haben.

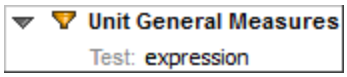


Unit Single Measure
Der Single-Measure Unit-Filter dient zum Auswählen von Facts, die mit einer Unit (Maßeinheit) ausgegeben werden, die durch eine einzige Maßeinheit definiert ist.



Art: qname oder exp

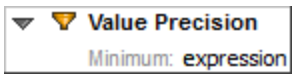
Unit General Measures
Der General Unit-Filter dient zum Auswählen von Facts auf Basis von Kriterien, die eine Reihe von Maßeinheiten beinhalten.



Value Nil
Der Nil-Filter dient zum Finden von Facts, die als Null ausgegeben werden.

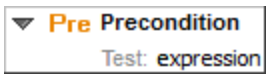


Value Precision
Der Precision-Filter dient zum Finden von Facts auf Basis dessen, ob diese eine tatsächliche oder abgeleitete Mindestpräzision haben, wobei berücksichtigt wird, dass die Präzision vom Wert des @decimal Attributs abgeleitet werden kann. Beachten Sie, dass mit dem Precision-Filter keine Facts ausgewählt werden, wenn dabei eine unendliche Mindestpräzision vorausgesetzt wird. Auch nicht numerische Facts oder Facts, die mit einem Null-Wert ausgegeben werden, werden vom Filter nicht ausgewählt.



17.4.2.6 Preconditions

Preconditions bieten eine Methode, um festzustellen, ob eine Gruppe gebundener Variablen einen Formula-Wert und ein Output Fact oder einen Assertion-Werttest oder einen Existence Count aktivieren kann.

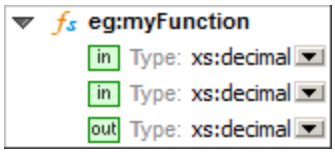


17.4.2.7 Funktionen

Eine benutzerdefinierte Funktion ist eine XPath-Funktion, die nicht in der XPath- und XQuery Functions-Spezifikation definiert ist und auch nicht in der XBRL Functions Registry enthalten ist. Benutzerdefinierte Funktionen können in XPath-Ausdrücken verwendet werden.

Funktionssignatur

In der Abbildung unten sehen Sie eine Funktionssignatur.



Die Child-Elemente einer benutzerdefinierten Funktionssignatur, definieren, falls vorhanden, die Datentypen der Input-Parametr der benutzerdefinierten Funktion. Die Reihenfolge der Input-Parameter der benutzerdefinierten Funktion entspricht der Dokumentreihenfolge der Child-Elemente der benutzerdefinierten Funktionssignatur.

Die Inputs werden in der Eingabehilfe "Details" in zusätzlichen Abschnitten angezeigt.

Funktionsimplementierung

In der Abbildung unten sehen Sie eine Funktionsimplementierung.



Eine benutzerdefinierte Funktionsimplementierung (Custom Function Implementation = CFI) enthält eine Sequenz von Child-Elementen, mit denen Namen für die Funktions-Inputs definiert, XPath-Ausdrücke, die die CFI enthalten, ausgedrückt werden und der Output der benutzerdefinierten Funktoin definiert wird.

Eine Funktionsimplementierungsbeziehung ist eine Beziehung zwischen einer benutzerdefinierten Funktionssignatur und einer benutzerdefinierten Funktionsimplementierung. Da eine Funktionsimplementierung das Ziel einer Funktionsimplementierungsbeziehung sein muss, wird sie immer unter der entsprechenden

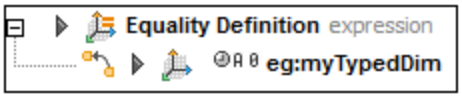
© 2016-2022 Altova GmbH

Funktionssignatur angezeigt. Falls die Beziehung fehlt (oder die Signatur unter einer anderen Linkrole definiert ist), wird die Implementierung direkt unter dem Abschnitt *Funktionen* angezeigt.

Die Inputs und Schritte werden in der Eingabehilfe "Details" in zusätzliche Abschnitten angezeigt.

17.4.2.8 Equality-Definitionen

Eine Equality-Definition ist eine Gleichheitsdefinition zwischen zwei beliebigen Werten in einer Typed-Dimension Domain-Definition. Eine Typed-Dimension Domain-Definition ist das Element in XML-Schema, das das Content Model für eine Typed Dimension definiert und das durch ein `@xbrldt:typedDomainRef` Attribut in dem XML-Schema-Element, in dem die Typed Dimension deklariert ist, als solches identifiziert wird. Eine Equality-Definition-Beziehung, die die Beziehung zwischen einer Typed-Dimension Domain-Definition und einer Equality-Definition is, wird als umgekehrte Beziehung zwischen der Equality-Definition und der entsprechenden Typed Dimension angezeigt.

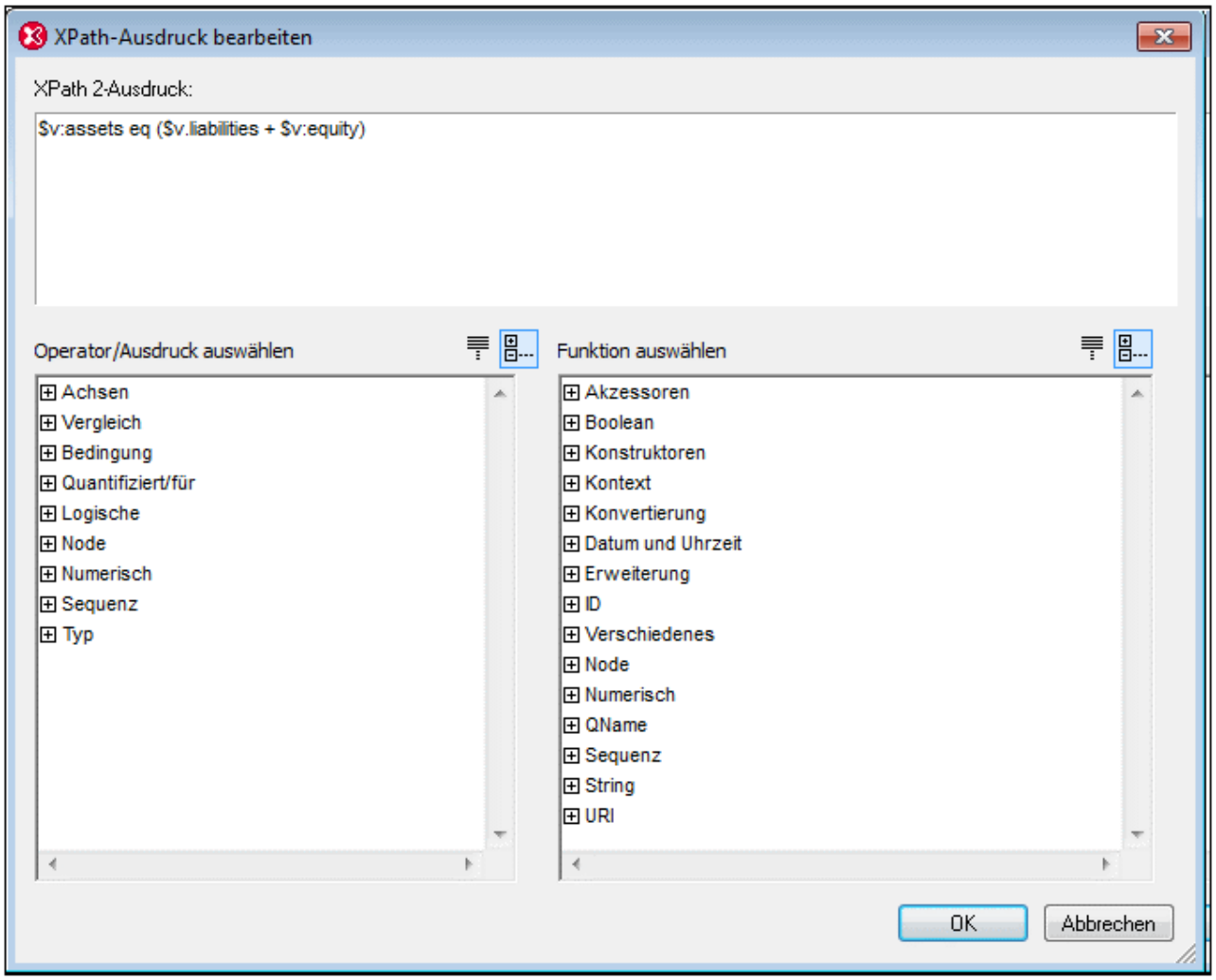


Eine Equality-Definition-Beziehung wird durch Ziehen einer Typed Dimension aus der Eingabehilfe "Globale Elemente" auf eine Equality-Definition-Komponente erzeugt. Beachten Sie, dass weder die Equality-Definition noch die Typed Dimension zu diesem Zeitpunkt an einer vorhandenen Equality-Definitionsbeziehung beteiligt sein müssen.

17.4.3 Bearbeiten der Eigenschaften von Formula-Komponenten

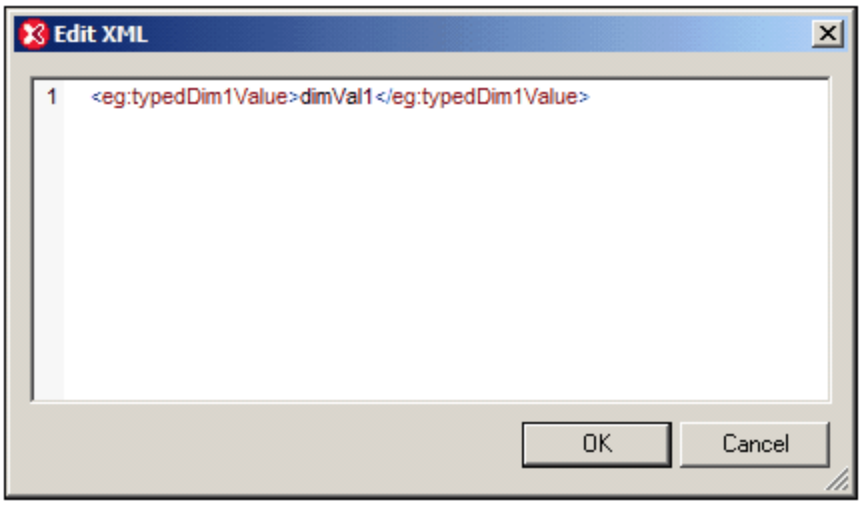
Die Eigenschaften von Formula-Komponenten können direkt im Diagramm oder in der Eingabehilfe "Details" bearbeitet werden.

Wenn im Diagramm eine Komponente eingeklappt ist, wird neben dem Beschreibungstext der Komponente entweder ihr Name (falls vorhanden) oder der Wert der entsprechenden Standardeigenschaft grau angezeigt. Durch Doppelklick auf die Komponente wird diese erweitert. Wenn Sie auf eine Eigenschaft doppelklicken wird diese im Bearbeitungsmodus angezeigt. Wenn eine Eigenschaft oder Inhalt einen XPath-Ausdruck enthält, wird das Dialogfeld "XPath-Ausdruck bearbeiten" angezeigt.



Wenn XML-Inhalt, wie z.B. der Inhalt von *Value* in der Typed Dimension Aspect-Regel einer Formula oder der Inhalt von *Fragments* in der OCC Aspect-Regel einer Formula bearbeitet wird, wird das Dialogfeld "XML bearbeiten" (*Abbildung unten*) angezeigt. (Klicken Sie mit der rechten Maustaste auf eine Formula, um eine Aspect-Regel hinzuzufügen.)

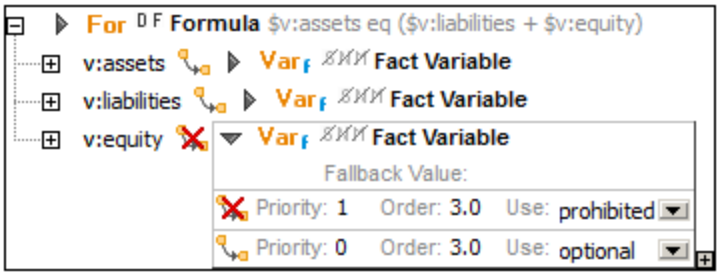
© 2016-2022 Altova GmbH



Nachdem Sie den XML-Text eingegeben und auf OK geklickt haben, wird der XML-Text als Inhalt der Eigenschaft eingegeben. Wenn der XML-Text nicht wohlgeformt ist, erscheint eine entsprechende Meldung und der Text wird nicht zugelassen.

17.4.4 Formula-Komponentenbeziehungen

Mit Hilfe von Drag-and-Drop kann eine Beziehung zwischen zwei Formula-Komponenten erstellt werden. Die Beziehungen werden im Diagramm mittels Bögen (Arcs) angezeigt (*siehe Abbildung unten*).



Es stehen die folgenden Ansichts- und Bearbeitungsoptionen zur Verfügung:

- Die Reihenfolge der Children einer Komponente hängt von den Werten der Arc-Eigenschaft *Order* ab, die durch Ziehen der Children mit der Maus geändert werden kann. (*siehe Abbildung oben*).
- Eine Child-Komponente kann mit der Maus auf oder unter eine andere Parent-Komponente gezogen werden, um die Beziehung (und ihre Eigenschaften) zu kopieren oder zu verschieben.
- Wenn eine neue Komponente über das Kontextmenü einer bestehenden (Parent)-Komponente erstellt wird, wird auch die Beziehung (der Bogen) automatisch generiert.
- Mit Hilfe der Befehle **Arc überschreiben** und **Arc entfernen** im Kontextmenü einer Child-Komponente können Sie die Beziehung zwischen der Komponente und ihrem Parent überschreiben bzw. entfernen.
- Wie bei Concept-Beziehungen werden mehrere Arcs von überschriebenen Beziehungen in Subzeilen angezeigt (*siehe Abbildung oben*).

Anmerkung: Die Arcrole von Formula-Komponentenbeziehungen kann nicht geändert werden.

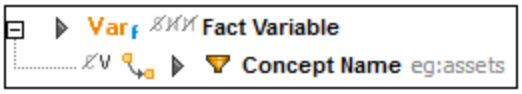
Variablengruppen-Beziehungen

Eine Variablengruppen-Beziehung ist eine Beziehung zwischen (i) einer Variablengruppen-Ressource (einer Value Assertion, Existence Assertion oder Formula) und (ii) einer Variablen (Fact Variable oder allgemeine Variable) oder einem Parameter. Der *Name* eines Variablenparameters wird vor dem Arc-Symbol angezeigt (Abbildung unten).



Variablenfilterbeziehungen

Eine Variablenfilterbeziehung ist eine Beziehung zwischen einer Fact Variable und einem Filter. Wenn das Boolesche Flag *Complement* (Symbol C im Diagramm) gesetzt ist, ist die Beziehung eine Variablenfilterbeziehung mit einem Komplement. Wenn das Boolesche Flag *Cover* (Symbol V im Diagramm) gesetzt ist, handelt es sich bei der Beziehung um eine Variablenfilterbeziehung mit Cover (*siehe Abbildung unten*). In diesem Fall deckt der Filter Aspects der gefilterten Facts ab.



Variablengruppen-Filterbeziehung

Eine Variablengruppen-Filterbeziehung (*siehe Abbildung unten*) ist eine Beziehung zwischen einer Variablengruppen-Ressource und einem Filter. Ein Filter, der Teil einer Variablengruppen-Filterbeziehung ist, ist per definitionem mit jeder der Fact-Variablen in der Variablengruppe, die durch die Ressource, mit der sie in Beziehung steht, definiert ist, verknüpft. Das Boolesche Flag *Complement* definiert, ob Variablen das Filter-Komplement verwenden. Alle Filter, die mittels Variablengruppen-Filterbeziehungen mit Fact Variablen verknüpft sind, decken definitionsgemäß keine Aspects ab.



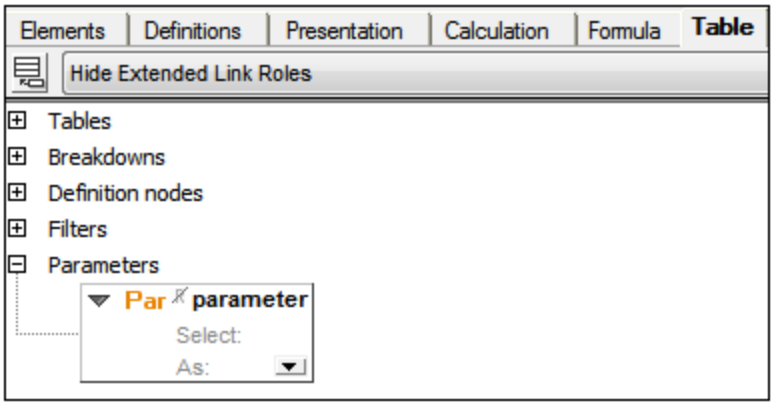
Visuelles Erstellen von Formulas in der Tabellenlayoutvorschau

Mit Hilfe der XBRL-Tabellenvorschau können XBRL-Taxonomieentwickler XBRL-Formulas mit einfachen Mausklicks erstellen. Näheres zu dieser Funktionalität finden Sie im Abschnitt Erstellen von Formulas in der Tabellenlayoutvorschau 935.

© 2016-2022 Altova GmbH

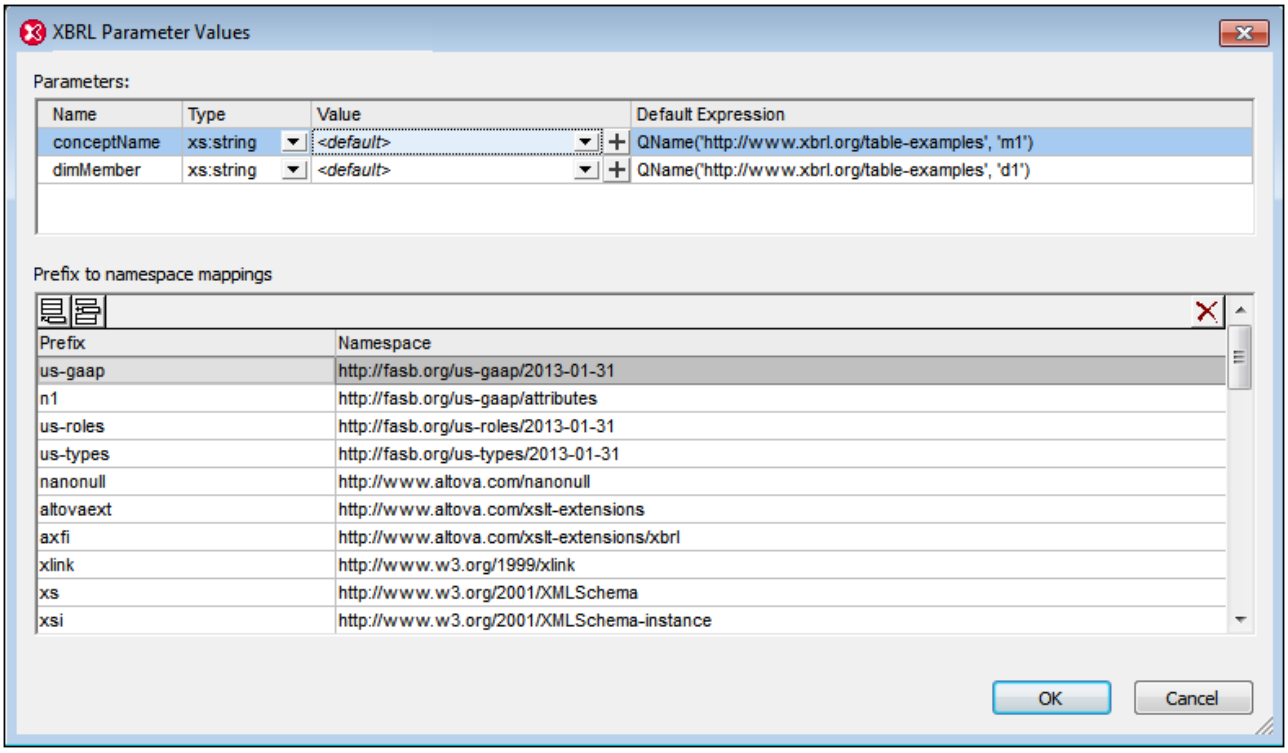
17.4.5 Formula-Parameter

XBRL-Parameter können in XPath-Ausdrücken in Formulas und Tabellendefinitionen verwendet werden. Parameter, die als Formula-Parameter (aus der Formula Linkbase) verwendet werden sollen, werden im Abschnitt "Formula" erstellt, während Tabellenparameter (aus der Table Linkbase) auf dem Register "Tabelle" erstellt werden. Sowohl Formula- als auch Tabellenparameter können lokal oder global sein. Bei lokalen Parametern handelt es sich im Grunde um globale Parameter, die zum Zeitpunkt ihrer Erstellung mit der entsprechenden Komponente (Formula oder Tabelle) verknüpft werden. Lokale Parameter werden durch Rechtsklick auf die Komponente (Formula oder Tabelle) und Auswahl des Befehls **Neuen Parameter hinzufügen** erstellt, während globale Parameter durch Rechtsklick in einen leeren Bereich des entsprechenden Registers und Auswahl des Befehls **Neuen Parameter hinzufügen** erstellt werden. Daraufhin wird ein neuer Parameter namens `parameter` in das Diagramm eingefügt (*in der Abbildung unten sehen Sie einen globalen Parameter*). Um den Parameternamen zu ändern, doppelklicken Sie auf den Namen und bearbeiten Sie ihn.



Jeder Parameter hat ein *Required* Flag. Wenn das Flag gesetzt ist, ist der Parameter obligatorisch, d.h. sein Wert muss von der verarbeitenden Applikation bereitgestellt werden. Wenn der Parameter nicht obligatorisch ist und kein Wert von der verarbeitenden Applikation bereitgestellt wird, kann der Wert mit Hilfe des in der Eigenschaft *Select* angegebenen XPath-Ausdrucks berechnet werden. Doppelklicken Sie in das *Select*-Feld, um einen XPath-Ausdruck einzugeben. Dieser Wert wird als Standardwert für den Parameter verwendet. In der optionalen Eigenschaft *As* wird der vom Parameter benötigte Datentyp definiert. Wählen Sie einen Datentyp aus der Dropdown-Liste der Auswahlliste aus.

Bei Parametern, die als Tabellenparameter verwendet werden, können Sie den Datentyp des Parameters bearbeiten und einen Parameterwert angeben, der den Standardwert außer Kraft setzt. Klicken Sie dazu auf **XBRL | Parameterwerte**. Geben Sie anschließend im Dialogfeld, das angezeigt wird (*Abbildung unten*), einen Parameterwert ein. Dieser Wert setzt den Standardwert außer Kraft. Da Parameter, die als Tabellenparameter verwendet werden, mehrere Werte haben können, können Sie durch Klicken auf das **+** Symbol in der Spalte *Wert* zusätzliche Parameterwerte definieren.



Die in diesem Dialogfeld zugewiesenen Werte von globalen Parametern werden nur für Tabellenparameter ausgewertet. Werte von in Formulas verwendeten Parametern können in diesem Dialogfeld nicht bearbeitet werden.

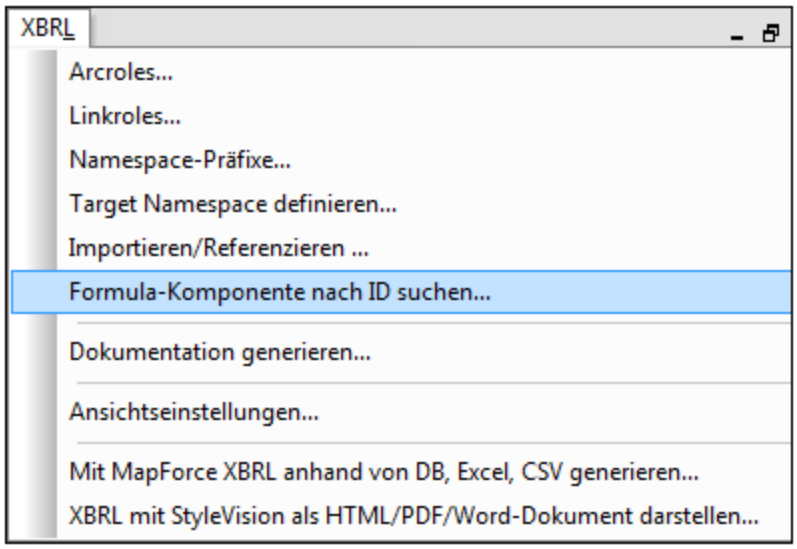
17.4.6 Suchen von Formula-Komponenten

Sie können anhand der ID nach einer Formula-Komponente suchen oder durch die Instanzen der Komponente im Dokument navigieren.

Suchen von Formula-Komponenten nach ihrer ID

In Taxonomien mit großen Formula-Linkbases, die mehrere Komponenten derselben Art enthalten, ist es hilfreich, eine Komponente nach der ID zu suchen. Dies erfolgt mit dem Menübefehl **XBRL | Formula-Komponente nach ID suchen**.

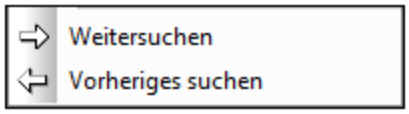
© 2016-2022 Altova GmbH



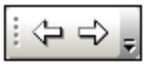
Wenn Sie auf den Befehl klicken, wird ein Dialogfeld angezeigt, in dem Sie nach der ID gefragt werden.

Suchen von Komponenteninstanzen

Die meisten Formula-Komponenten werden mehrmals im Formula Linkbase-Diagramm angezeigt: (i) Die Definition, die sich direkt unter dem entsprechenden Abschnitts-Node befindet und (ii) alle Referenzen zur Komponente (über Beziehungen). Mit Hilfe der Befehle **Weitersuchen** und **Vorheriges suchen** des Komponenten-Kontextmenüs (*Abbildung unten*) können Sie zu allen Stellen navigieren, wo diese Formula-Komponente referenziert wird.



Sie können diese Befehle auch über Symbolleisten-Schaltflächen aufrufen (*Abbildung unten*).



Wenn die Definition einer Komponente erreicht wurde, wird eine entsprechende Meldung angezeigt.

17.5 XBRL-Tabellendefinitionseditor

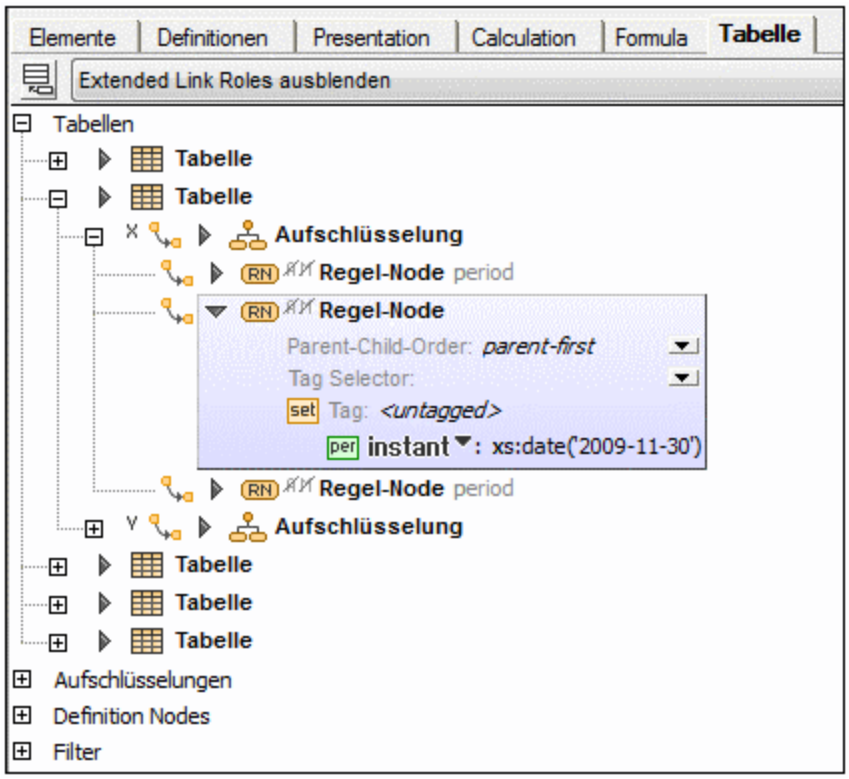
Als Ergänzung zur Presentation Linkbase ist in den XBRL-Spezifikationen auch eine Table Linkbase (Tabellen-Linkbase) beschrieben. Tabellen bieten eine alternative Möglichkeit, um Ansichten von in XBRL-Taxonomien definierten Concepts zu definieren. Anstatt Concepts wie im Fall einer Presentation Linkbase in Form einer Hierarchie anzuzeigen, können hier Tabellen mit mehreren Achsen definiert werden. Dabei müssen die Komponenten von Achsen nicht auf einzelne Elemente beschränkt werden, sondern es kann eine Kombination von Dimensions, Zeitdauerreferenzen, Units, Entities oder beliebigen anderen Eigenschaften definiert werden, anhand derer durch Taxonomien dargestellte Finanz-Facts beschrieben werden können. Im XBRL Table Linkbase Overview 1.0 und in der Table Linkbase 1.0 Recommendation vom 18. März 2014 finden Sie eine Einführung in die Syntax und Semantik von XBRL Tabellen-Linkbases.

XMLSpy entspricht der Table Linkbase 1.0 Recommendation vom 18. März 2014, und verwendet den Namespace `http://xbrl.org/2014/table`.

Während in den XBRL-Standard-Linkbases (Presentation, Calculation, Definition) Beziehungen zwischen Concepts mittels Locators und Standard Arcs in Standard Extended Links definiert sind, enthält eine Tabellen-Linkbase Komponenten (Tabellen, Aufschlüsselungen, Definition Nodes, usw.) und deren Beziehungen in Form von Ressourcen und Generic Arcs in generischen Extended Links. In der Table Linkbase-Spezifikation ist eine Sequenz von drei Modellen und Prozessen beschrieben, um das jeweilige Modell in das nächste zu transformieren. Bei den drei Modellen handelt es sich um das Definition Model (Definitionsmodell), das Structural Model (Strukturmodell) und das Layout (oder Rendering Model). Das Definitionsmodell ist ein Modell des semantischen Inhalts der Tabellen-Linkbase. Tabellen werden durch ihre Achsen definiert und Achsendefinitionen wiederum bestehen aus Definition Node-Strukturen.

Der XBRL-Tabellendefinitionseditor von XMLSpy wurde als Teil des XBRL-Taxonomie-Editors der Applikation implementiert. Er steht auf dem Register "Tabelle" der XBRL-Ansicht (*siehe Abbildung unten*) zur Verfügung.

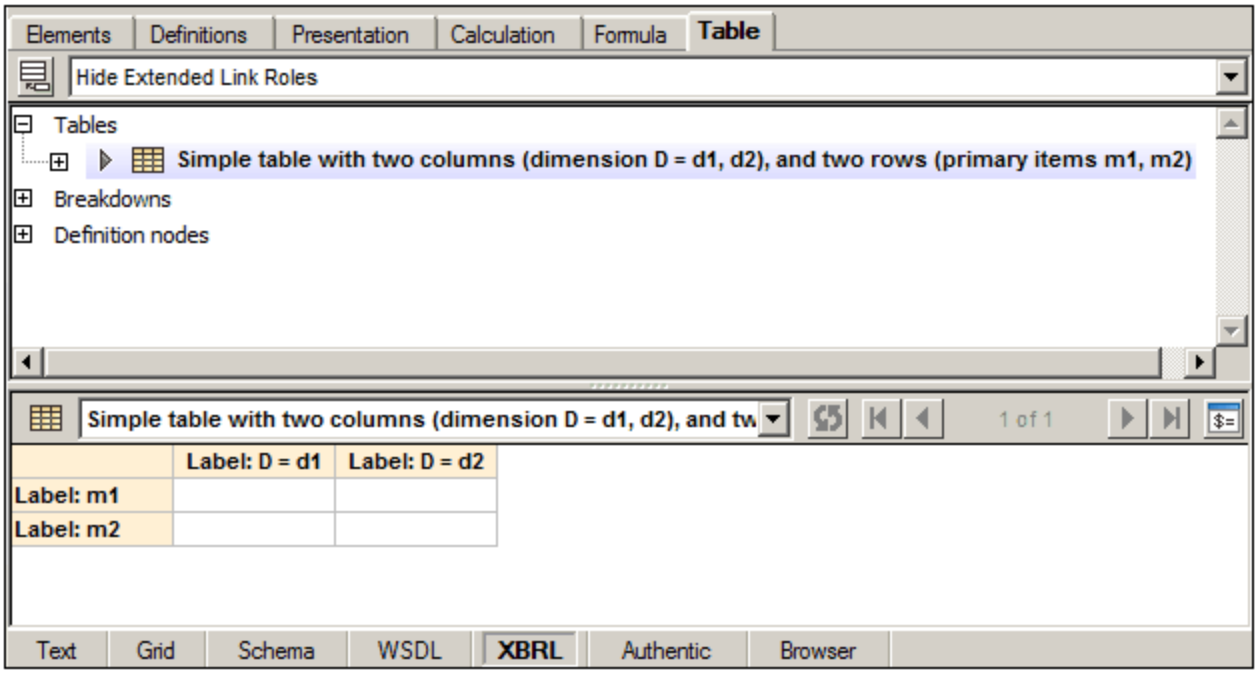
© 2016-2022 Altova GmbH



Über das Register "Tabelle" können Sie zusammen mit den Eingabehilfen "Übersicht" und "Details" Tabellendefinitionen erstellen und bearbeiten. In der Eingabehilfe "Übersicht" können Sie die Standard-Linkbase für XBRL-Tabellen (die Datei, in der die Tabellendefinitionen standardmäßig gespeichert werden) definieren, während Sie in der Eingabehilfe "Details" die Eigenschaften und den Inhalt der Tabellenkomponenten bearbeiten können. Sie können Tabellendefinitionen auch direkt über das Register "Tabelle" bearbeiten.

XBRL-Tabellenlayoutvorschau

Um eine Vorschau des Layout einer Tabellendefinition zu sehen, steht im XBRL-Taxonomie-Editor in der XBRL-Ansicht auf dem Register "Tabelle" ein XBRL-Tabellenlayoutvorschau-Fenster zur Verfügung (*siehe Abbildung unten*). Wenn eine Tabelle oder Tabellenkomponente im Diagramm ausgewählt wird, wird im Tabellenlayoutvorschau-Fenster unterhalb des Diagramms eine Vorschau der Tabelle angezeigt (*siehe Abbildung unten*). Alternativ dazu können Sie in der Dropdown-Auswahlliste des Vorschaubereichs eine Tabelle auswählen. Dies ist eine Liste von Tabellen in der Table Linkbase.



Nähere Informationen zur Vorschaufunktion finden Sie in den folgenden Abschnitten:

- Tabellenstruktur 908
- Tabellenparameter 929
- Tabellenlayoutvorschau 933

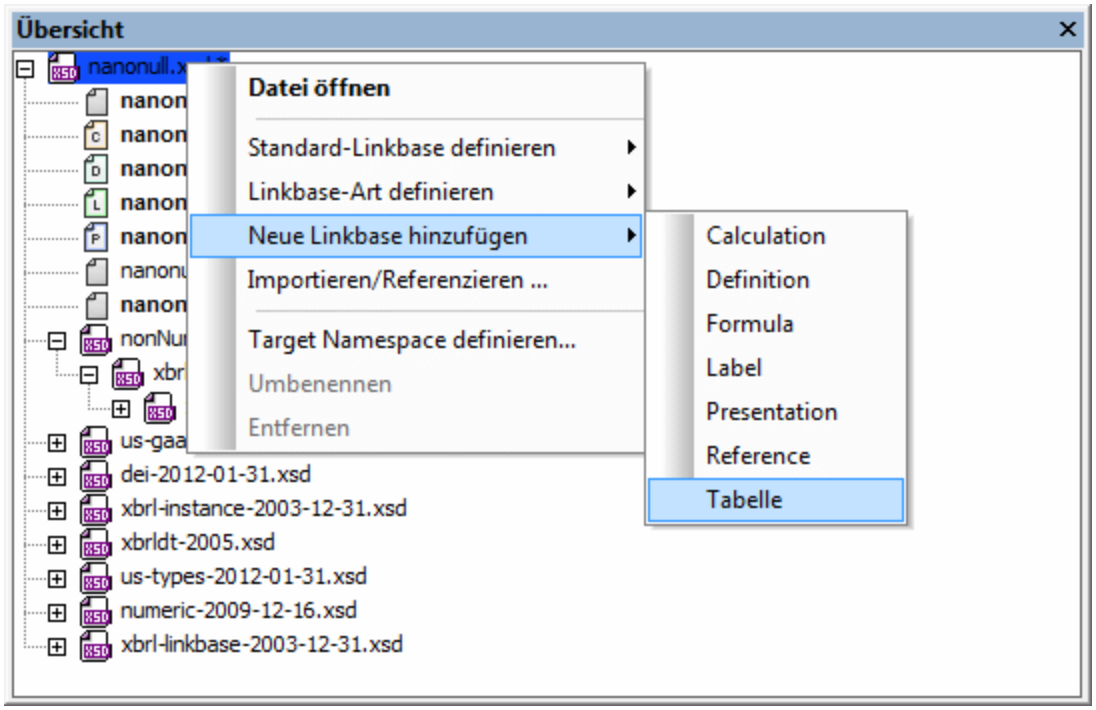
17.5.1 Tabellen-Linkbases und Link Roles

Während in den XBRL-Standard-Linkbases (Presentation, Calculation, Definition) Beziehungen zwischen Concepts mittels Locators und Standard Arcs in Standard Extended Links definiert sind, enthält eine Tabellen-Linkbase Komponenten (Tabellen, Aufschlüsselungen, Definition Nodes, usw.) und deren Beziehungen. Diese Beziehungen sind in Form von Ressourcen und Generic Arcs in generischen Extended Links definiert.

Hinzufügen einer Tabellen-Linkbase

Klicken Sie in der Eingabehilfe "Übersicht" (*Abbildung unten*) mit der rechten Maustaste auf die Taxonomiedatei oder eine bestehende Linkbase und wählen Sie den Befehl **Neue Linkbase hinzufügen | Tabelle**. Die hinzugefügte Linkbase wird zur Standard-Tabellen-Linkbase-Datei. Die Standard-Tabellen-Linkbase-Datei ist die Datei, in der die neuen Tabellendefinitionen beim Speichern der Taxonomiedatei gespeichert werden. Falls Sie eine andere Tabellen-Linkbase-Datei zur Standard-Tabellen-Linkbase machen möchten, klicken Sie mit der rechten Maustaste darauf und wählen Sie den Befehl **Standard-Linkbase definieren | Tabelle** (*siehe Abbildung unten*).

© 2016-2022 Altova GmbH



Beachten Sie, dass Standard-Linkbases fett angezeigt werden und dass geänderte Linkbases, die noch nicht gespeichert wurden, mit einem Sternchen gekennzeichnet sind.

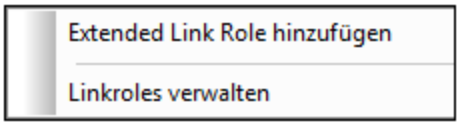
Die Tabellen-Linkbase wird auf dem Register "Tabelle" angezeigt.

Hinweis: Wenn zu einem Zeitpunkt, zu dem noch keine Tabellen-Linkbase vorhanden ist, eine Tabellenkomponente zur Taxonomie hinzugefügt wird, 922 wird automatisch eine Tabellen-Linkbase erstellt.

Link Roles

Wie bei Standard Extended Links (für Definitions, Presentations, Calculations), müssen generische Links einen Extended Link Role-Wert definieren, der Beziehungen desselben Typs in voneinander getrennte Netzwerke aufteilt. Im Diagramm auf dem Register "Tabelle" sind alle generischen Extended Links mit derselben Link Role unter einem einzigen Link Role Node zusammengefasst, *selbst wenn sie sich in unterschiedlichen Linkbase-Dateien befinden.*

Generische Link Roles können im Diagramm über das Kontextmenü des Hintergrundbereichs (*Abbildung unten*) erstellt werden. Beachten Sie allerdings, dass dieses Kontextmenü nur dann angezeigt wird, wenn auf dem Register "Tabelle" in der Auswahlliste für die Ansichtsoptionen die Option *Alle Extended Link Roles anzeigen* ausgewählt ist.



Dieses Menü steht auch über die Symbolleistenschaltfläche *Extended Link role hinzufügen / Link roles verwalten* zur Verfügung. Da Relationship Networks (Beziehungsnetzwerke) für eine Tabellen-Linkbase nicht von großer Bedeutung sind, ist die Standardansicht des Registers "Tabelle" *Extended Link Roles ausblenden*, sodass die Link Roles ausgeblendet und die Tabellenkomponenten ohne ihre Link Roles angezeigt werden.

Wenn zum Zeitpunkt der Erstellung der Extended Link Role keine Standard-Tabellen-Linkbase-Datei vorhanden ist, so wird automatisch eine erstellt. Und falls die Standard-Linkbase-Datei zum Zeitpunkt der Erstellung einer Link Role keine Link Role enthält, wird automatisch eine Link Role in der Standard-Linkbase-Datei erstellt.

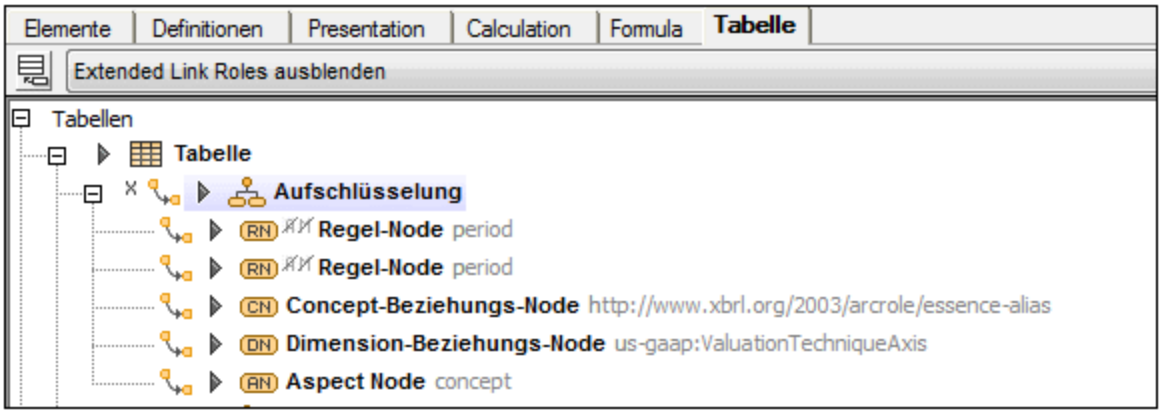
17.5.2 Tabellenstruktur

Die Struktur einer Tabelle in der Tabellendefinition wird durch die Achsen der Tabelle (X,Y,Z) definiert. Jede dieser Achsen entspricht einer oder mehreren Aufschlüsselungskomponenten (Breakdowns) (*siehe Abbildung unten*).

- Die X- und Y-Achse entsprechen den Spalten bzw. Zeilen der generierten Tabelle. Eine Beschreibung dazu finden Sie unter X- und Y-Achse 909.
- Wenn eine Z-Achse definiert ist, wird diese als separate Tabelle dargestellt. Siehe dazu Abschnitt Z-Achse 920.
- Jede Aufschlüsselungskomponente kann mehrere Tabellendefinitionsnodes enthalten (*Abbildung unten*). Es gibt verschiedene Arten von Definitionsnodes:

  – Regel-Nodes (Symbol RN in der Abbildung unten)
  – Concept-Beziehungs-Nodes (CN)
  – Dimension-Beziehungs-Nodes (DN) und
  – Aspect Nodes (AN).

  Eine Beschreibung der Struktureigenschaften dieser Definitionsnodes finden Sie im Abschnitt Definition Nodes 912.



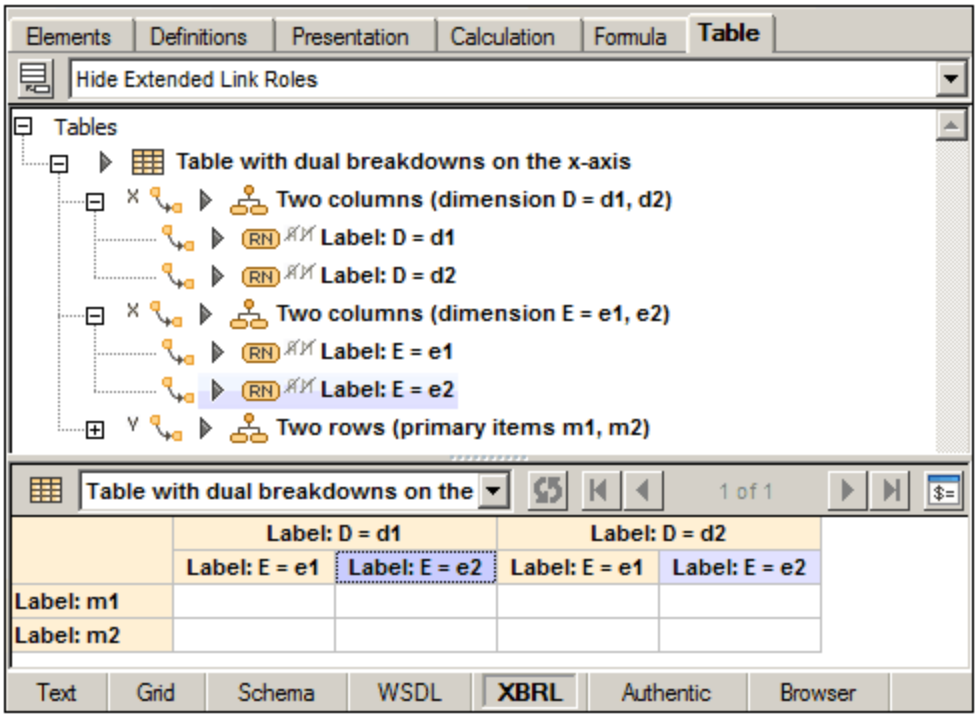
Projektionen für mehrere Aufschlüsselungen

Mit einer einzigen Tabellenachse können mehrere unabhängige Aufschlüsselungen verknüpft sein. Der Mechanismus, wie mehrere Aufschlüsselungen zu einer einzigen "effektiven" Aufschlüsselung kombiniert

© 2016-2022 Altova GmbH

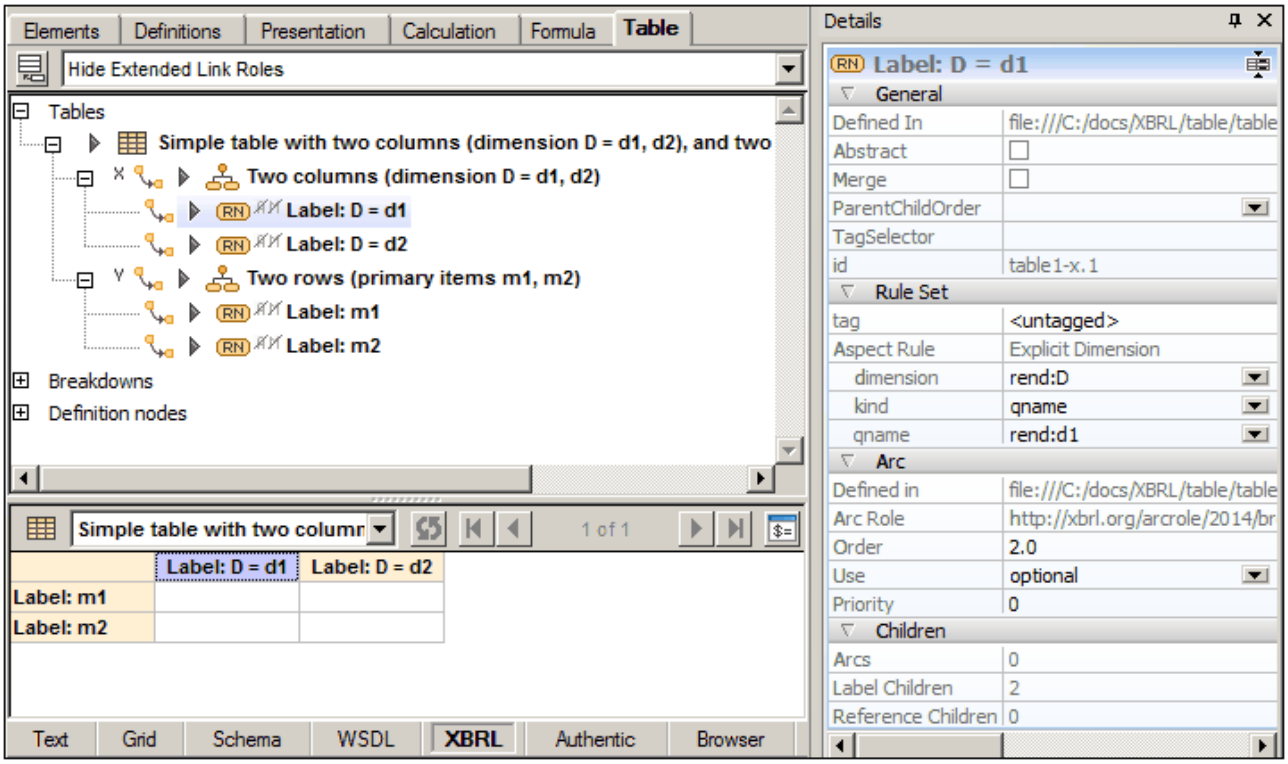
werden, wird als Projektion bezeichnet. Die relative Vorrangigkeit mehrerer Aufschlüsselungen für eine einzige Achse wird durch das `@order` Attribut der einzelnen Aufschlüsselungen definiert. Die Aufschlüsselungen werden in Form von Baumstrukturen dargestellt. Für jedes Blatt der ersten Aufschlüsselung wird die gesamte zweite Aufschlüsselung angehängt. Dies setzt sich rekursiv weiter fort.

So sind z.B. in der Abbildung unten zwei Aufschlüsselungen für die X-Achse vorhanden: **`dimension D`** ist in der Vorrangigkeit höher gereiht als **`dimension E`**. Dadurch wird für jedes Blatt von **`dimension D`** (**`d1`** und **`d2`**) die gesamte Baumstruktur von **`dimension E`** angehängt. Da die X-Achse Spalten generiert, bilden diese Aufschlüsselungen eine Projektion für die Spaltenstruktur der Tabelle. In der Abbildung unten sehen Sie dazu die Tabellenlayoutvorschau.



17.5.2.1 X- und Y-Achse

Die X- und Y-Achse legen die Spalten bzw. Zeilen einer Tabelle fest. Für jede Achse sind eine oder mehrere hierarchische Aufschlüsselungen definiert (*siehe Abbildung unten*). Die Aufschlüsselung(en), die einer einzigen Achse entsprechen, werden zu einer einzigen "effektiven" Aufschlüsselung aufgelöst. Wenn für eine Achse nur eine Aufschlüsselung vorhanden ist, so bildet diese Aufschlüsselung die effektive Aufschlüsselung. Eine Beschreibung der Auflösungsmethode für eine Achse mit mehreren Aufschlüsselungen finden Sie weiter unten unter Projektionen für mehrere Aufschlüsselungen 910.



Beachten Sie die folgenden Eigenschaften und Bearbeitungsfunktionen im Zusammenhang mit Achsen:

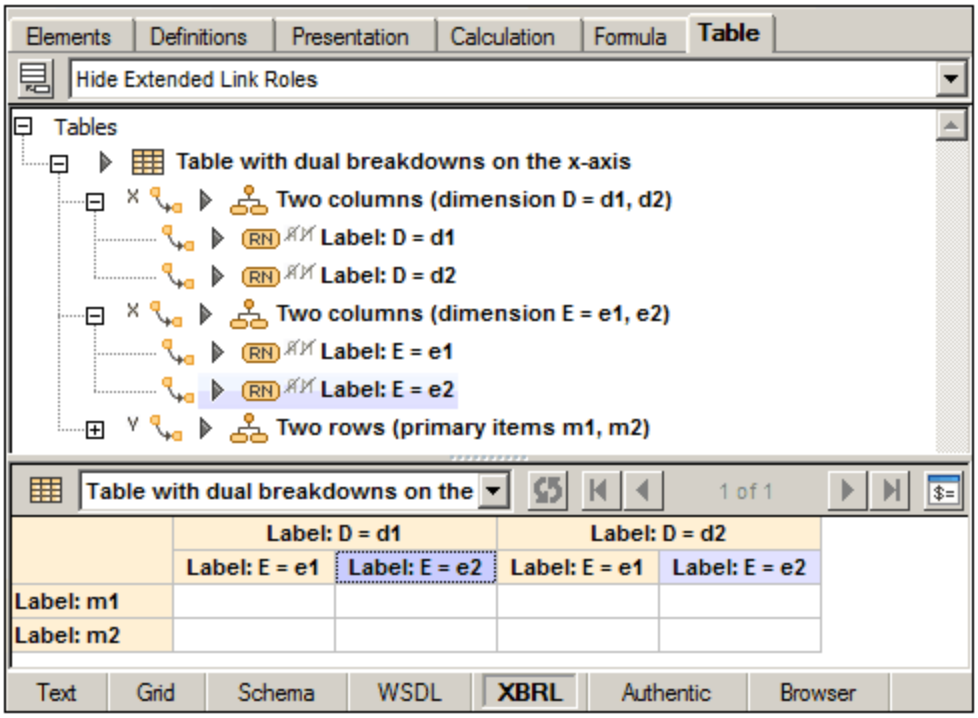
- In Tabellendefinitionen entspricht die X-Achse den Spalten der generierten Tabelle, während die Y-Achse den Tabellenzeilen entspricht (siehe *Abbildung der Tabellenlayoutvorschau unten*).
- Jede Achse kann eine oder mehrere Aufschlüsselungen haben (siehe Projektionen für mehrere Aufschlüsselungen 910 unten).
- Jede Zelle in der generierten Tabelle hat einen orange-gelben Hintergrund. In der Tabellendefinition entspricht eine Zelle einem Definition Node in einer Aufschlüsselung der Achse.
- Wenn eine Zelle ausgewählt wird, wird auch der dazugehörige Definition Node ausgewählt und umgekehrt. Die Hintergrundfarbe von Zellen ausgewählter Komponenten ist lila.
- Wenn eine Komponente ausgewählt ist, werden ihre Eigenschaften in der Eingabehilfe "Details" angezeigt und können dort bearbeitet werden (*siehe Abbildung oben*).
- Datenzellen haben keine Hintergrundfarbe. Sie sind immer leer, da die Taxonomie selbst keine Facts enthält.
- Zellen-Constraints werden (über Tag-Selektoren, falls vorhanden) anhand der Achsen berechnet und auf dem Register "Constraints" der Eingabehilfe 'Details" angezeigt. Siehe Abbildung im Abschnitt Z-Achse 920.

Projektionen für mehrere Aufschlüsselungen

Mit einer einzigen Tabellenachse können mehrere unabhängige Aufschlüsselungen verknüpft sein. Der Mechanismus, wie mehrere Aufschlüsselungen zu einer einzigen "effektiven" Aufschlüsselung kombiniert werden, wird als Projektion bezeichnet. Die relative Vorrangigkeit mehrerer Aufschlüsselungen für eine einzige Achse wird durch das `@order` Attribut der einzelnen Aufschlüsselungen definiert. Die Aufschlüsselungen werden in Form von Baumstrukturen dargestellt. Für jedes Blatt der ersten Aufschlüsselung wird die gesamte zweite Aufschlüsselung angehängt. Dies setzt sich rekursiv weiter fort.

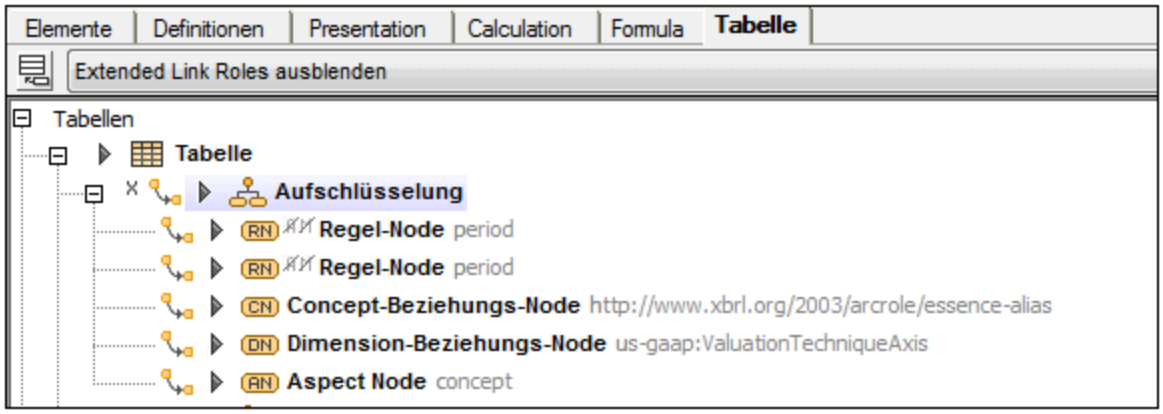
© 2016-2022 Altova GmbH

So sind z.B. in der Abbildung unten zwei Aufschlüsselungen für die X-Achse vorhanden: **`dimension D`** ist in der Vorrangigkeit höher gereiht als **`dimension E`**. Dadurch wird für jedes Blatt von **`dimension D`** (**`d1`** und **`d2`**) die gesamte Baumstruktur von **`dimension E`** angehängt. Da die X-Achse Spalten generiert, bilden diese Aufschlüsselungen eine Projektion für die Spaltenstruktur der Tabelle. In der Abbildung unten sehen Sie dazu die Tabellenlayoutvorschau.



17.5.2.2 Definition Nodes

Jede Aufschlüsselungskomponente kann mehrere Tabellen-Definition-Nodes enthalten (*siehe Abbildung unten*).



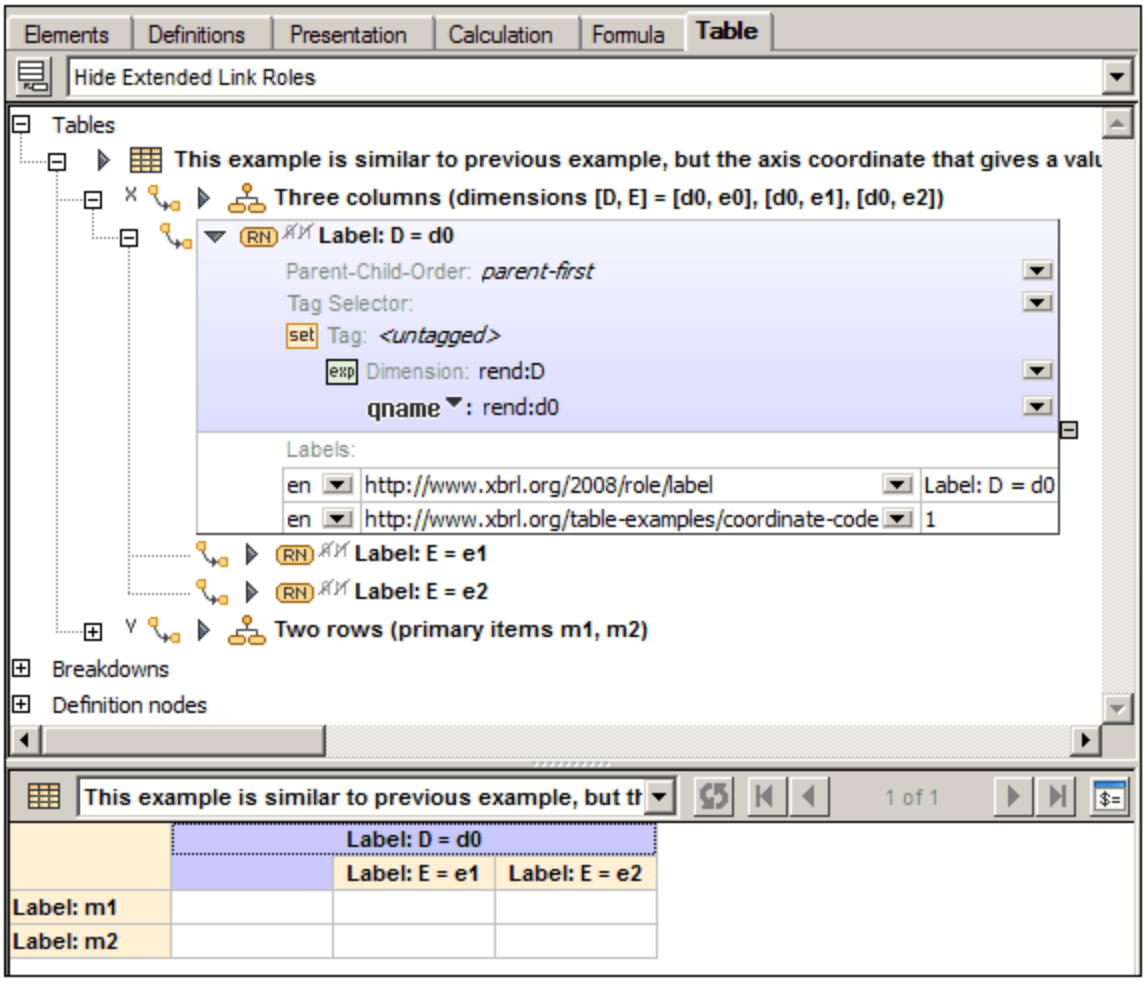
Es gibt unterschiedliche Arten von Definition Nodes:

- Regel-Nodes 912 (Symbol `RN` in der Abbildung oben)
- Concept-Beziehungs-Nodes 916 (**CN**)
- Dimension-Beziehungs-Nodes 916 (**DN**)
- Aspect Nodes 920 (**AN**)

17.5.2.2.1 Regel-Nodes

Ein Regel-Node definiert Aspect-Regeln für einen oder mehrere Aspects: Concept, Zeitraum (Period), Unit, Entity Identifier, Dimension oder Open Content Aspect. Die Komponente in der Definitionsstruktur entspricht genau einer Zelle im Layout, wenn der Regel-Node abstrakt ist oder keine Children hat. Andernfalls enthält das Layout eine weitere Roll-up-Zelle, deren Position durch den effektiven Wert der Regel-Node-Eigenschaft `parentChildOrder` festgelegt wird:

© 2016-2022 Altova GmbH

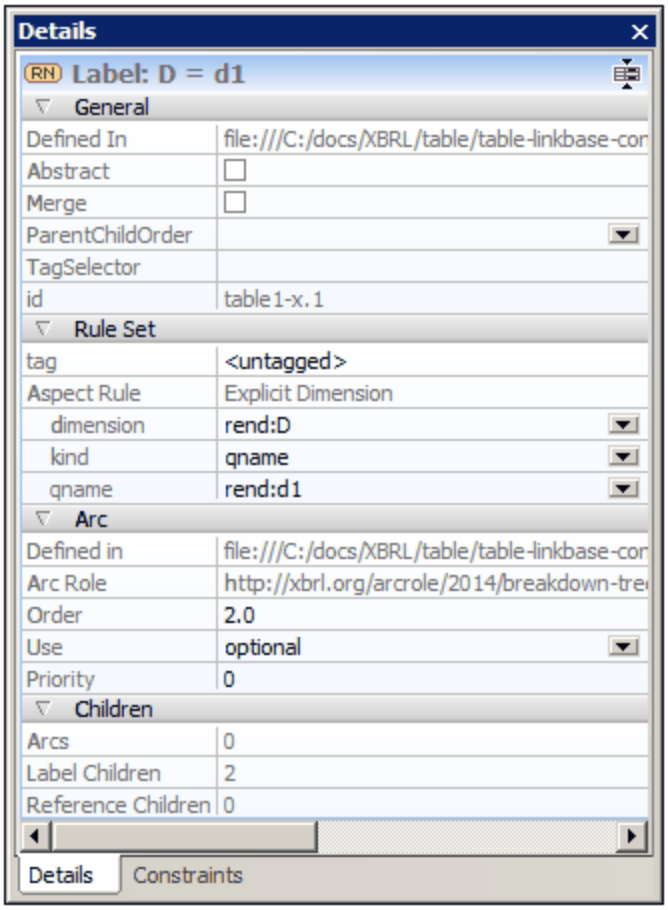


Die Überschrift der Layout-Zelle wird folgendermaßen anhand des Regel-Node berechnet:

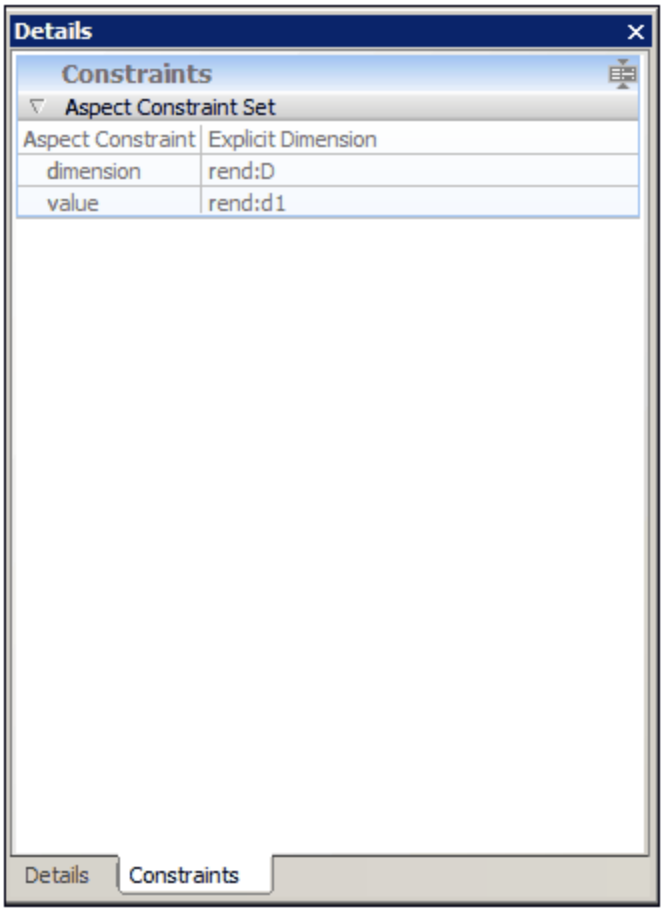
- Wenn der Node mit einem benutzerdefinierten Label verknüpft ist, wird dieser Label-Text angezeigt.
- Wenn kein Label vorhanden ist, der Node aber einen einzigen Aspect Constraint (Concept, Dimension, Unit, Entity-Identifier oder Zeitraum) definiert, wird dessen Wert angezeigt (z.B. der qualifizierte Name des Concept).
- Andernfalls wird der statische Text `Regel-Node` verwendet.

Eingabehilfe "Details"

Die Eigenschaften des Definition Node werden in der Eingabehilfe "Details" auf dem Register "Details" (*Abbildung unten links*) angezeigt. Auf dem Register "Constraints" (*Abbildung unten rechts*) sehen Sie eine schreibgeschützte Ansicht des/der Aspect Constraint Set/s, die anhand der Aspect-Regeln des Regel-Node berechnet werden.

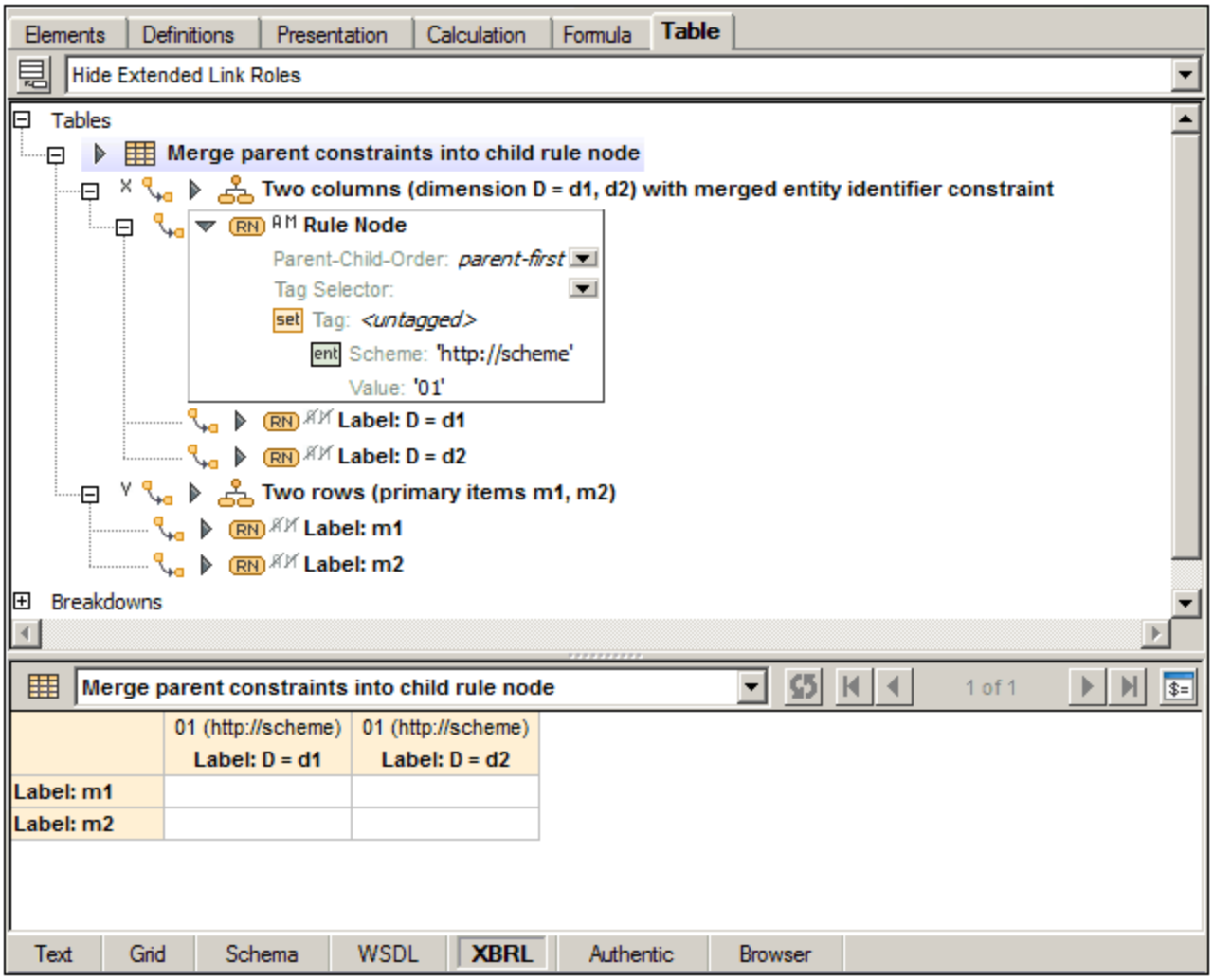


© 2016-2022 Altova GmbH



Zusammengeführte Regel-Nodes

Ein zusammengeführter Regel-Node weist auf zusätzliche Eigenschaften hin, die für alle seine Children gelten, d.h. alle seine Constraints bilden Teil der durch seine Children erzeugten einzelnen Constraint Sets (*siehe Abbildung unten*).

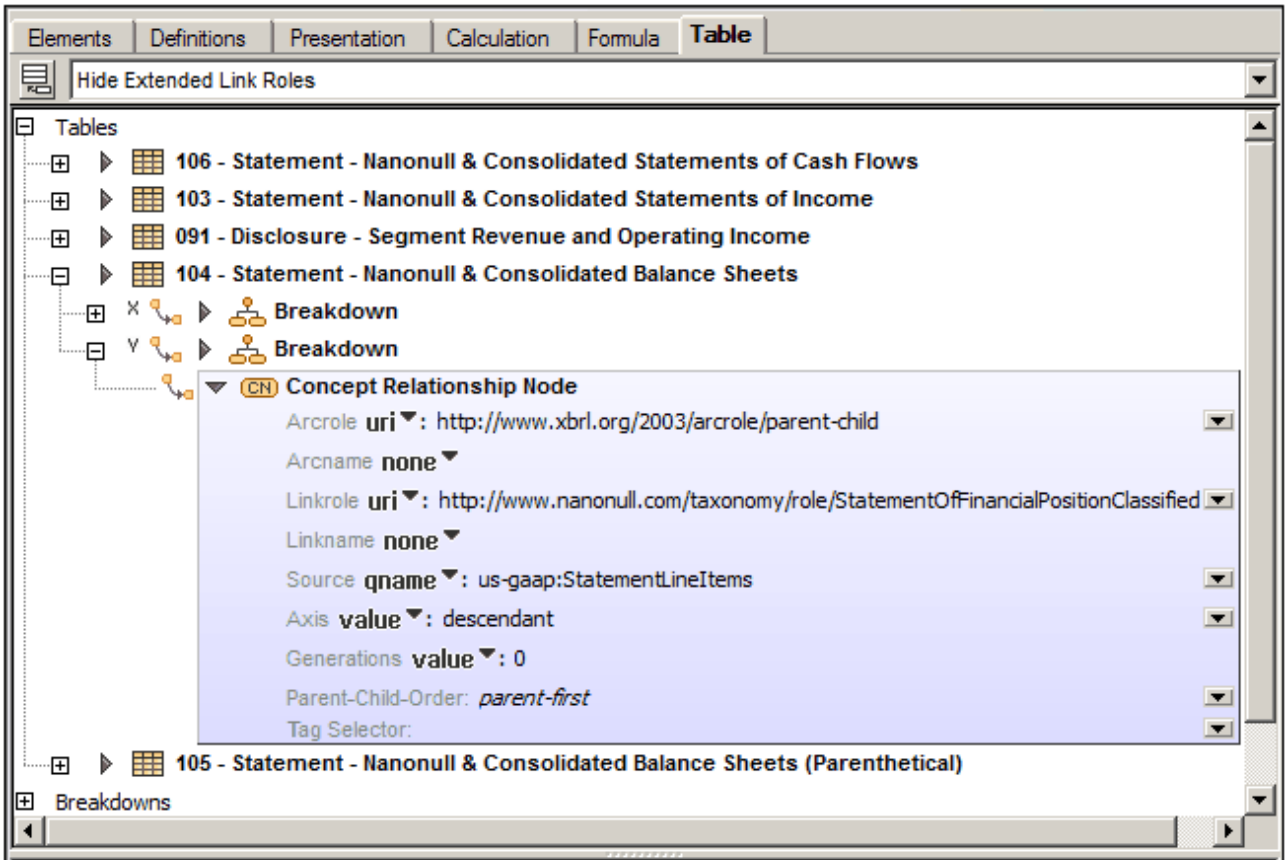


17.5.2.2.2 Beziehungs-Nodes

Ein Concept oder Dimension-Beziehungs-Node wird zu einer Struktur bestehend aus Struktur-Nodes, die durch Concept-Netzwerke oder Explicit Dimension Members im DTS definiert sind, aufgelöst. Daher entspricht die Komponente in der Definitionsstruktur einem Zellenblock im Layout.

In der Abbildung auf der linken Seite sehen Sie eine Tabellendefinition, die einen Concept-Beziehungs-Node enthält. In der Abbildung auf der rechten Seite sehen Sie das dazugehörige Concept-Netzwerk, das in diesem Fall in der Presentation Linkbase definiert ist.

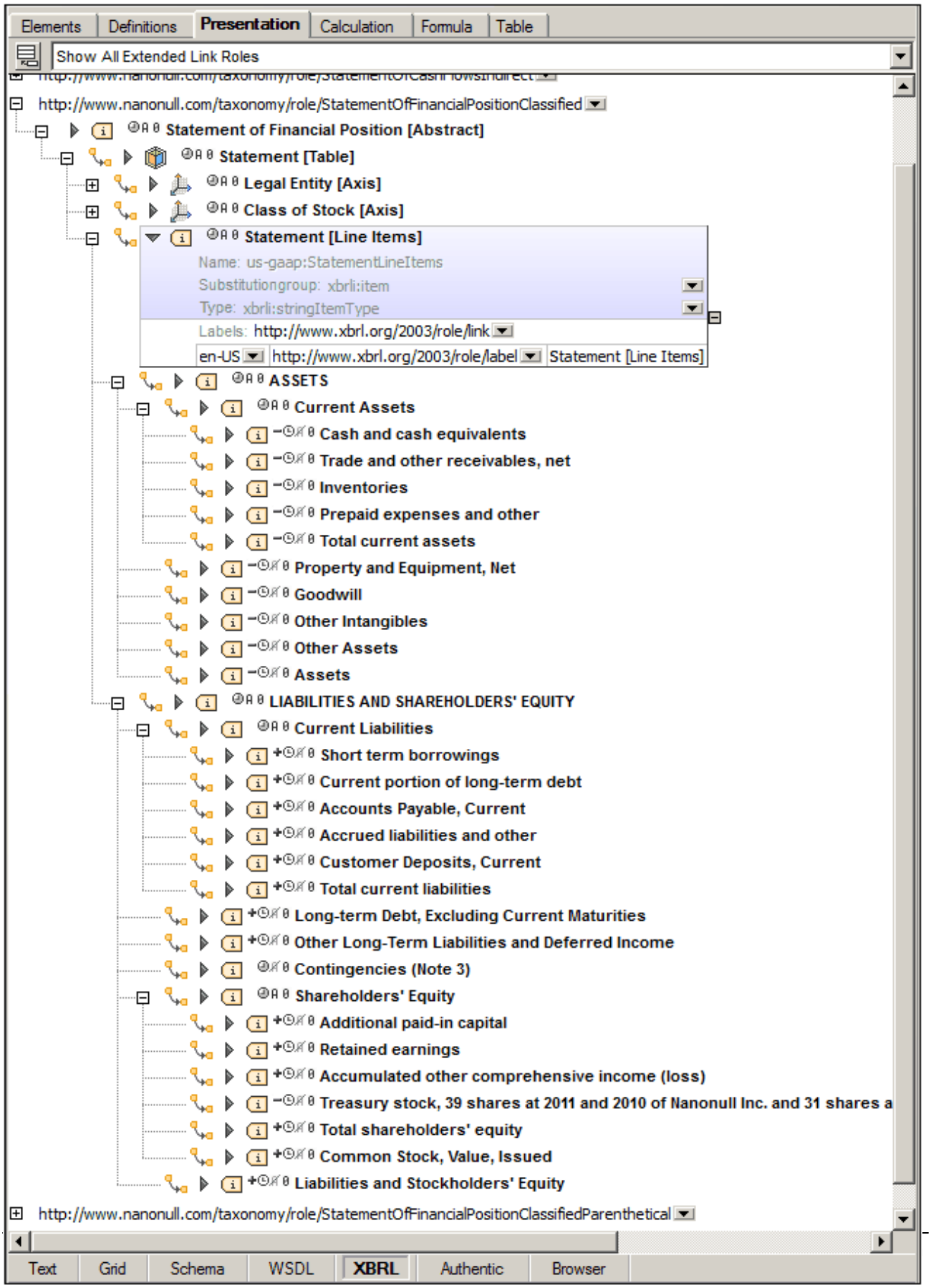
© 2016-2022 Altova GmbH



104 - Statement - Nanonull & Consolidated Balance Sheets — 1 of 1

			2010-08-31	2009-11-30	2
ASSETS	Current Assets	Cash and cash equivalents			
		Trade and other receivables, net			
		Inventories			
		Prepaid expenses and other			
		Total current assets			
	Property and Equipment, Net				
	Goodwill				
	Other Intangibles				
	Other Assets				
	Assets				
LIABILITIES AND SHARE	Current Liabilities	Short term borrowings			
		Current portion of long-term debt			
		Accounts Payable, Current			
		Accrued liabilities and other			
		Customer Deposits, Current			
		Total current liabilities			
	Long-term Debt, Excluding Current Maturities				
	Other Long-Term Liabilities and Deferred Income				
	Contingencies (Note 3)				
	Shareholders' Equity	Additional paid-in capital			
		Retained earnings			
		Accumulated other comprehensive			
		Treasury stock, 39 shares at 2011 an			
		Total shareholders' equity			
		Common Stock, Value, Issued			
	Liabilities and Stockholders' Equity				

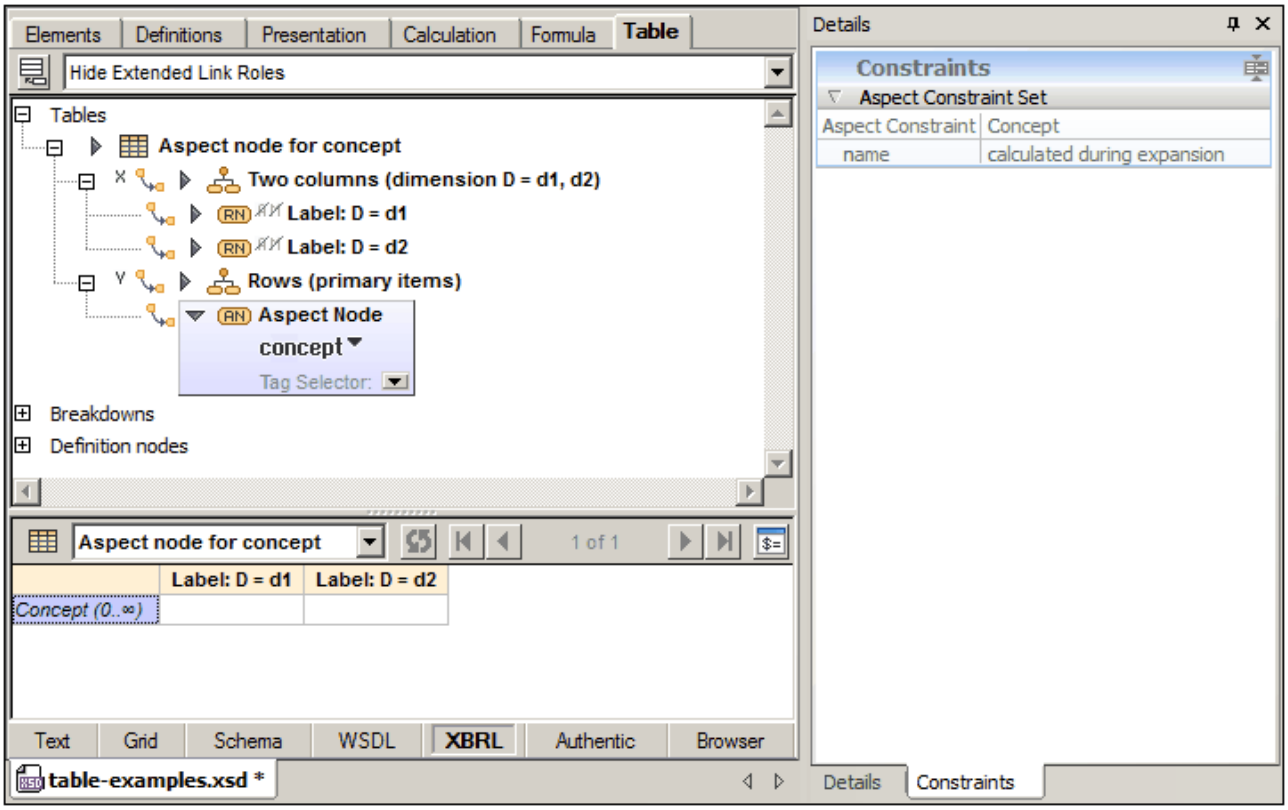
© 2016-2022 Altova GmbH


Elements
Definitions
Presentation
Calculation
Formula
Table
Show All Extended Link Roles
http://www.nanonull.com/taxonomy/role/StatementOfFinancialPositionClassified
Statement of Financial Position [Abstract]
Statement [Table]
Legal Entity [Axis]
Class of Stock [Axis]
Statement [Line Items]
Name: us-gaap:StatementLineItems
Substitutiongroup: xbrli:item
Type: xbrli:stringItemType
Labels: http://www.xbrl.org/2003/role/link
en-US
http://www.xbrl.org/2003/role/label
Statement [Line Items]
ASSETS
Current Assets
Cash and cash equivalents
Trade and other receivables, net
Inventories
Prepaid expenses and other
Total current assets
Property and Equipment, Net
Goodwill
Other Intangibles
Other Assets
Assets
LIABILITIES AND SHAREHOLDERS' EQUITY
Current Liabilities
Short term borrowings
Current portion of long-term debt
Accounts Payable, Current
Accrued liabilities and other
Customer Deposits, Current
Total current liabilities
Long-term Debt, Excluding Current Maturities
Other Long-Term Liabilities and Deferred Income
Contingencies (Note 3)
Shareholders' Equity
Additional paid-in capital
Retained earnings
Accumulated other comprehensive income (loss)
Treasury stock, 39 shares at 2011 and 2010 of Nanonull Inc. and 31 shares a
Total shareholders' equity
Common Stock, Value, Issued
Liabilities and Stockholders' Equity
http://www.nanonull.com/taxonomy/role/StatementOfFinancialPositionClassifiedParenthetical
Text
Grid
Schema
WSDL
XBRL
Authentic
Browser

Ein Beziehungs-Node hat genau einen Aspect Constraint: Concept oder Explicit Dimension. Daher entspricht die Überschrift der einzelnen Layout-Zellen dem Concept Label (falls vorhanden) oder seinem qualifizierten Namen. In der Eingabehilfe "Details" sehen Sie auf dem Register "Details" die Eigenschaften des Beziehungs-Node, während Sie auf dem Register "Constraints" den/das durch die im Layout aktive Zelle definierte Aspect Constraint (Set) sehen.

17.5.2.2.3 Aspect Nodes

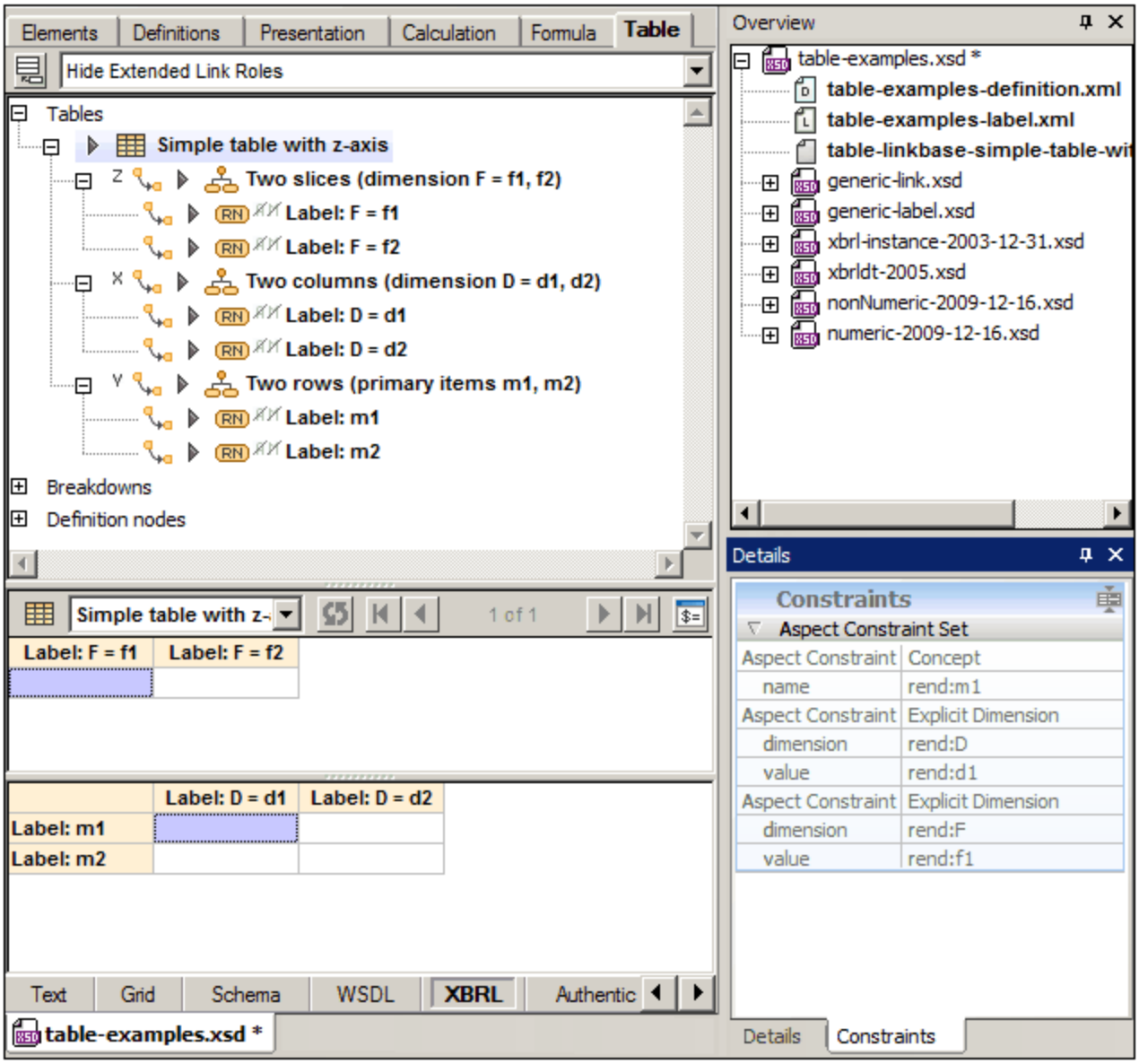
Ein Aspect Node ist ein offener Definition Node, der einen einzigen Teil-Aspect direkt definiert. Während des Layout wird ein Aspect Node für jeden einzelnen Wert seines Teil-Aspect, der in den Facts einer XBRL-Instanzdatei vorhanden ist, zu einer Zelle erweitert. Da der Aspect-Wert-Constraint nicht zur Gänze durch die Definition des Node und das DTS definiert ist, wird in der Layoutvorschau ein Platzhalter angezeigt (*siehe Abbildung unten*).



17.5.2.3 Z-Achse

Wenn eine Tabellendefinition eine Z-Achse enthält, wird diese Achse als zweidimensionale Mehrfachtabelle interpretiert. Die Z-Achse wird in der Tabellenlayoutvorschau als separate Tabelle oberhalb der XY-Tabelle angezeigt (*siehe Abbildung unten*).

© 2016-2022 Altova GmbH



Wenn eine Tabellendefinition eine Z-Achse enthält, so sind in der Tabellenvorschau immer **zwei** Datenzellen aktiv (siehe Abbildung oben). Die Koordinaten für alle drei Achsen sind folgendermaßen definiert: die X- und Y-Koordinate in der XY-Tabelle und die Z-Koordinate in der Z-Tabelle. In der Abbildung oben sehen Sie dies.

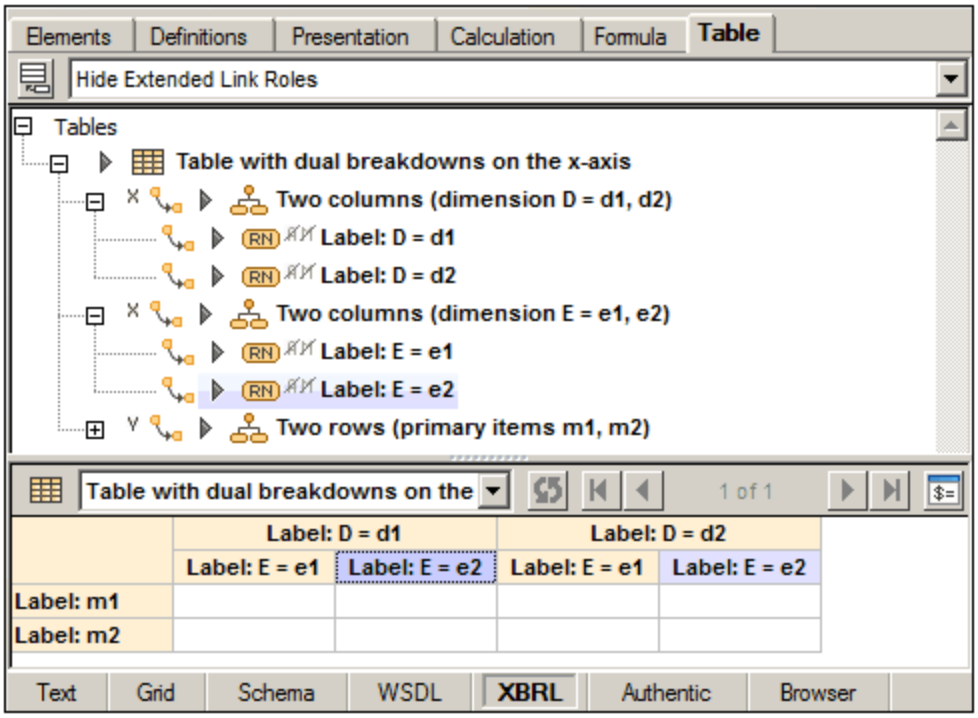
Die Eigenschaften im Zusammenhang mit Achsen und Bearbeitungsfunktionen sind dieselben wie bei der X- und Y-Achse[909]. Die Zellen-Constraints werden (mittels Tag-Selektoren, falls vorhanden) anhand der Achsen berechnet und in der Eingabehilfe "Details" auf dem Register "Constraints" angezeigt (*siehe Abbildung oben*).

Projektionen für mehrere Aufschlüsselungen

Mit einer einzigen Tabellenachse können mehrere unabhängige Aufschlüsselungen verknüpft sein. Der Mechanismus, wie mehrere Aufschlüsselungen zu einer einzigen "effektiven" Aufschlüsselung kombiniert

werden, wird als Projektion bezeichnet. Die relative Vorrangigkeit mehrerer Aufschlüsselungen für eine einzige Achse wird durch das `@order` Attribut der einzelnen Aufschlüsselungen definiert. Die Aufschlüsselungen werden in Form von Baumstrukturen dargestellt. Für jedes Blatt der ersten Aufschlüsselung wird die gesamte zweite Aufschlüsselung angehängt. Dies setzt sich rekursiv weiter fort.

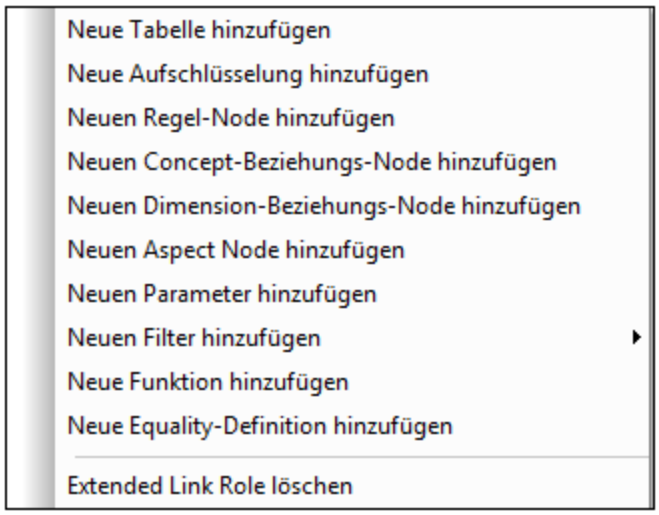
So sind z.B. in der Abbildung unten zwei Aufschlüsselungen für die X-Achse vorhanden: **`dimension D`** ist in der Vorrangigkeit höher gereiht als **`dimension E`**. Dadurch wird für jedes Blatt von **`dimension D`** (**`d1`** und **`d2`**) die gesamte Baumstruktur von **`dimension E`** angehängt. Da die X-Achse Spalten generiert, bilden diese Aufschlüsselungen eine Projektion für die Spaltenstruktur der Tabelle. In der Abbildung unten sehen Sie dazu die Tabellenlayoutvorschau.



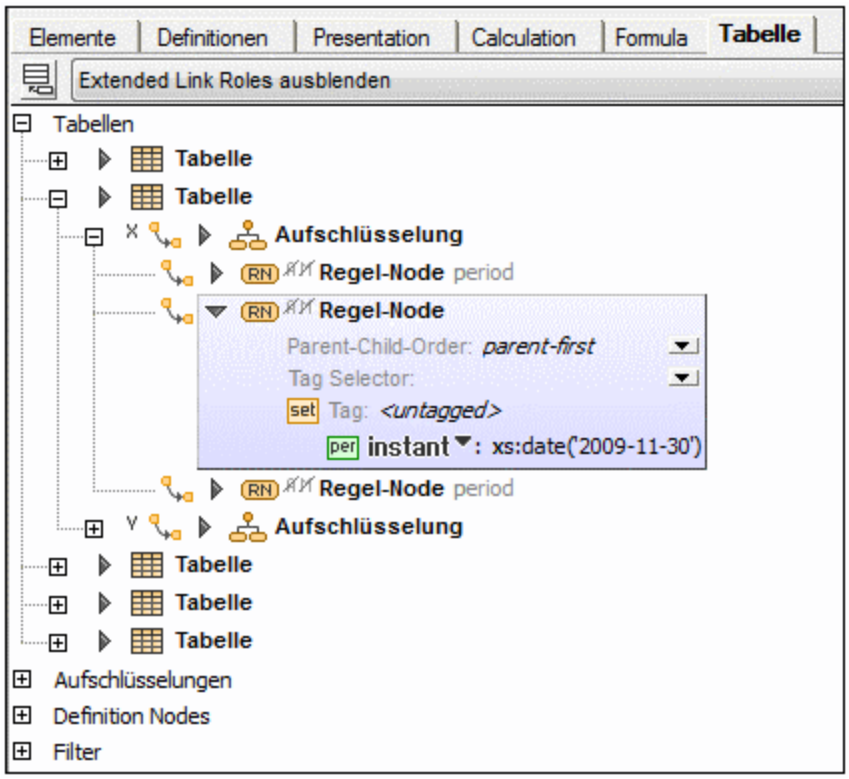
17.5.3 Tabellenkomponenten

Neue Tabellenkomponenten werden über das Kontextmenü einer Link Role (*Abbildung unten)* oder bei ausgewählter Ansichtsoption *Extended Link Roles ausblenden* über die Symbolleistenschaltfläche  *Neue Tabellenkomponente hinzufügen* erstellt.

© 2016-2022 Altova GmbH

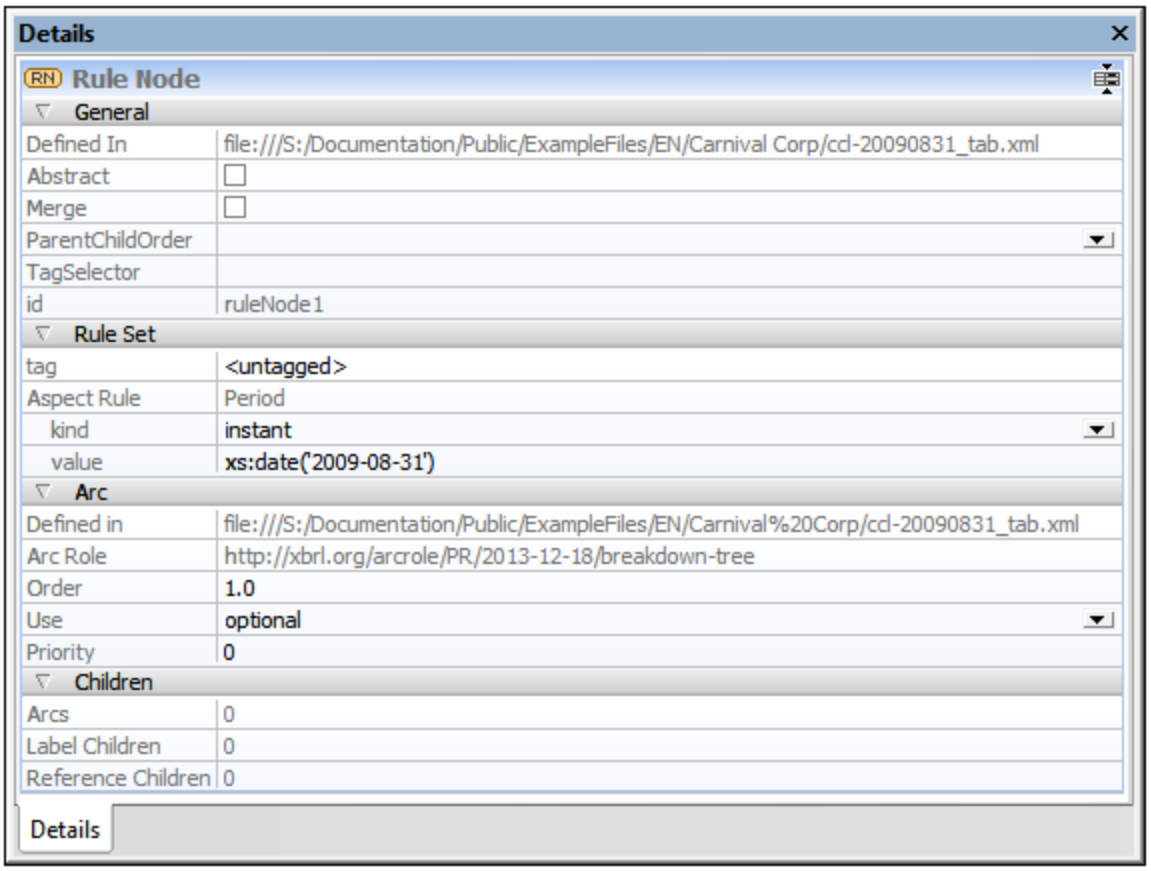


Wie verschiedene Komponenten hinzugefügt werden, wird in den Unterabschnitten dieses Abschnitts beschrieben. Nachdem eine Tabellenkomponente hinzugefügt wurde, wird Sie auf dem Register "Tabelle" im Diagramm angezeigt (*siehe Abbildung unten*).



Aus Gründen der Übersichtlichkeit sind die Tabellenkomponenten in Abschnitte gegliedert, wobei die Beziehungen zu anderen Komponenten (die Arcs) in Form einer Baumstruktur angezeigt werden (*siehe Abbildung oben*). Die Eigenschaften von Komponenten und ihren Beziehungen (Arcs) werden im Diagramm in Form von Symbolen links von der Komponente bzw. dem Arc gekennzeichnet (*siehe Abbildung oben*).

Unten sehen Sie die Eingabehilfe "Details" des in der Abbildung oben gezeigten Regel-Node. Die Eigenschaften des Node werden im Abschnitt *Allgemein* aufgelistet. Die Werte von Booleschen Eigenschaften werden durch ein Häkchen für `true` und kein Häkchen `false` angezeigt. Weitere Einzelheiten zum Node sind in zusätzlichen Abschnitten aufgelistet.



Um die Eigenschaften eines Arc in der Eingabehilfe "Details" zu sehen, wählen Sie im Diagramm die `to`-Komponente (Ziel) aus; die Eigenschaften des Arc werden daraufhin im Abschnitt *Arc* aufgelistet.

Kontextmenüs im Tabelleneditor

Je nach Art der Komponente werden unterschiedliche Kontextmenüs für Tabellenkomponenten angezeigt. Die Menübefehle sind in die folgenden Abschnitte gegliedert:

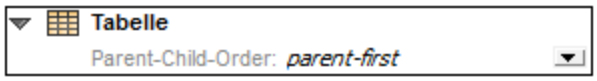
- Änderung von Beziehungen (nur für untergeordnete Einträge): *Arc überschreiben/entfernen*
- Änderung von Inhalt (Regel-Node, Relationship Nodes): z.B. *Aspect-Regel anhängen/einfügen*
- *Labels/References hinzufügen*
- Erstellung von neuen Child-Komponenten (einschließlich Beziehungen): z.B. *Neue Aufschlüsselung hinzufügen*
- Löschung einer Komponente (einschließlich Beziehungen)
- *Nächstes suchen/Weitersuchen* (von Komponente)

Hinweis: Content-Elemente, die über das Kontextmenü erstellt oder entfernt werden können, werden in der Eingabehilfe "Details" in zusätzlichen Abschnitten, wie z.B. *Regeln* angezeigt.

© 2016-2022 Altova GmbH

17.5.3.1 Tabelle

Eine Tabelle stellt die Eigenschaft `parent-child-order` (parent-first/children-first) bereit. Sie definiert die Standardposition von Roll-up Nodes aus allen Closed Definition Nodes in der Tabelle, für die diese Position nicht außer Kraft gesetzt wurde.



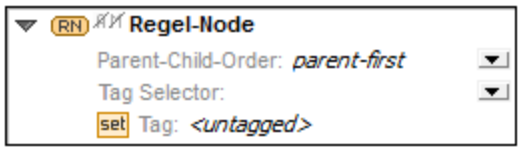
17.5.3.2 Aufschlüsselung (Breakdown)

Eine Aufschlüsselung stellt die Eigenschaft `parent-child-order` (parent-first/children-first) bereit. Sie definiert die Standardposition von Roll-up Nodes aus allen Closed Definition Nodes in der Aufschlüsselung und setzt den von der Tabelle vererbten Wert außer Kraft.



17.5.3.3 Definition Node: Regel

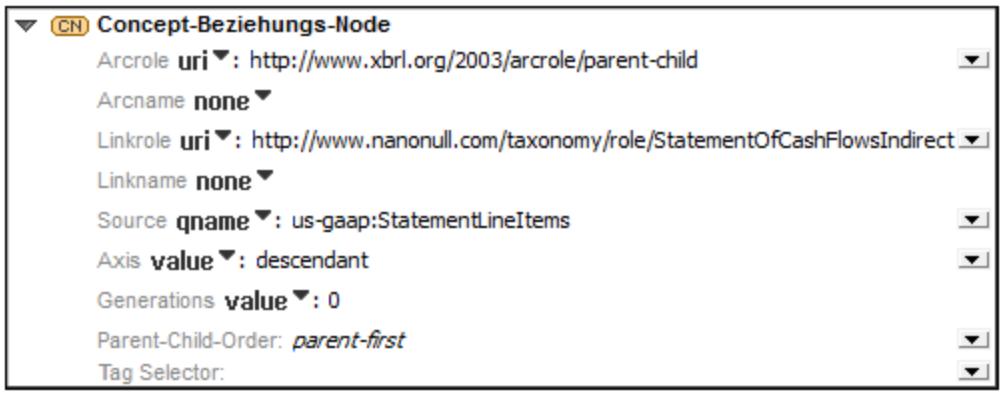
Ein Regel-Node (Rule Node) ist ein Closed Definition Node, der Constraints mittels Aspect-Regeln (siehe Formula-Komponente) definiert. Ein Regel-Node definiert ein oder mehrere Regelgruppen (Rule Sets), also Gruppen von Aspect-Regeln. Jede dieser Regelgruppen kann einen Tag definieren. In maximal einer dieser Regelgruppen kann der Tag weggelassen werden. Diese Regelgruppe ohne Tag wird immer vor allen mit Tag versehenen Regelgruppen angezeigt. Eine leere Regelgruppe ohne Tag wird nicht angezeigt, wenn mindestens eine mit Tag versehene Regelgruppe vorhanden ist. Ein Regel-Node stellt zwei Boolesche Eigenschaften in Form von Symbolen bereit: `abstract` und `merge`. In der Abbildung unten sehen Sie einen Regel-Node ohne Aspect-Regeln.



17.5.3.4 Definition Node: Concept-Beziehung

Ein Concept-Beziehungs-Node findet Concepts, indem er einen Tree Walk durchführt, d.h. die Verzweigungen eines XBRL 2.1-Netzwerks durchsucht. Dieser Vorgang wird eindeutig durch das Netzwerk und eine oder

mehrere Beziehungsquellen identifiziert. Eine Concept-Beziehung muss ein einziges Netzwerk identifizieren. In den meisten Fällen genügt die Kombination von Link Role und Arc Role um das Netzwerk eindeutig zu identifizieren, unter Umständen müssen zusätzliche Informationen angegeben werden wie z.B. der Name des Arc oder der Name des Extended Link.

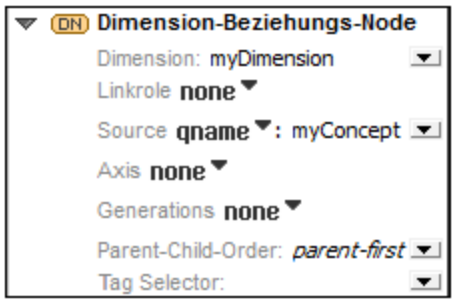



```
Arcrole:     Kind = uri | exp
Arcname:     Kind = none | qname | exp
Linkrole:    Kind = none | uri | exp
Linkname:    Kind = none | qname | exp
Source:      Kind = qname | exp
Axis:        Kind = none | value | exp
Generations: Kind = none | value | exp
```

Concept-Beziehungs-Nodes dürfen keine untergeordneten Strukturen haben.

17.5.3.5 Definition Node: Dimension-Beziehung

Mit einem Dimension-Beziehungs-Node wird eine Struktur von expliziten Dimension Members in Form eines Tree Walk eines Dimensional Relationship Set (DRS) beschrieben. Dieser Tree Walk wird durch eine oder mehrere Beziehungsquellen eindeutig identifiziert.




```
Linkrole:    Kind = none | uri | exp
Source:      Kind = qname | exp
Axis:        Kind = none | value | exp
Generations: Kind = none | value | exp
```

Dimension-Beziehungs-Nodes dürfen keine untergeordneten Strukturen haben.

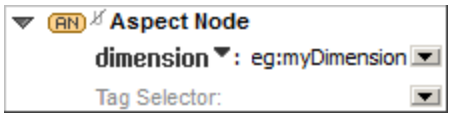
© 2016-2022 Altova GmbH

17.5.3.6 Definition Node: Aspect

Ein Aspect Node definiert genau einen Aspect.



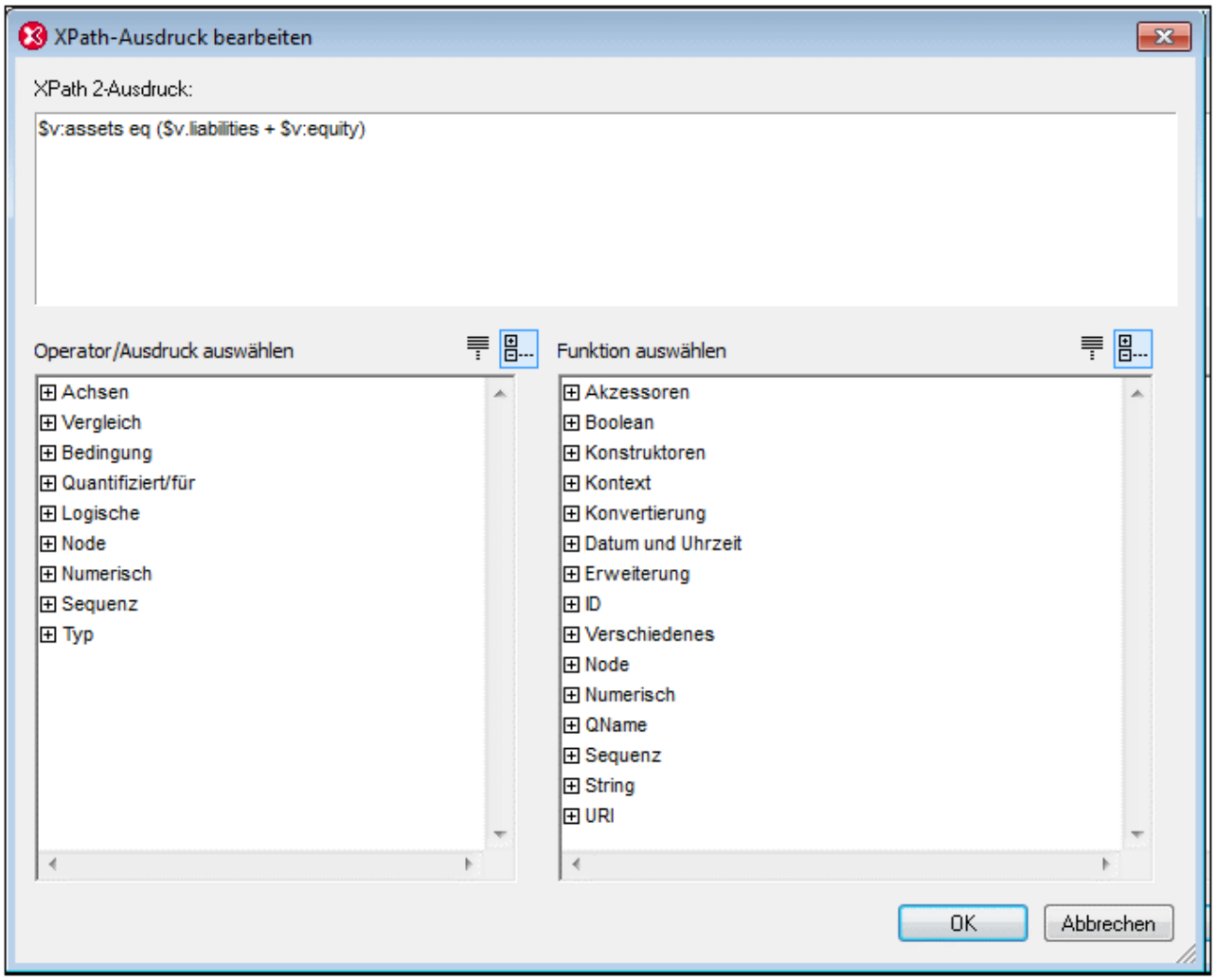
Dimensional Aspect-Spezifikationen stellen die zusätzliche Boolesche Eigenschaft `include unreported value` als Symbol bereit.



17.5.4 Bearbeiten von Komponenteneigenschaften und -inhalt

Die Eigenschaften von Formula-Komponenten können direkt im Diagramm oder in der Eingabehilfe "Details" bearbeitet werden.

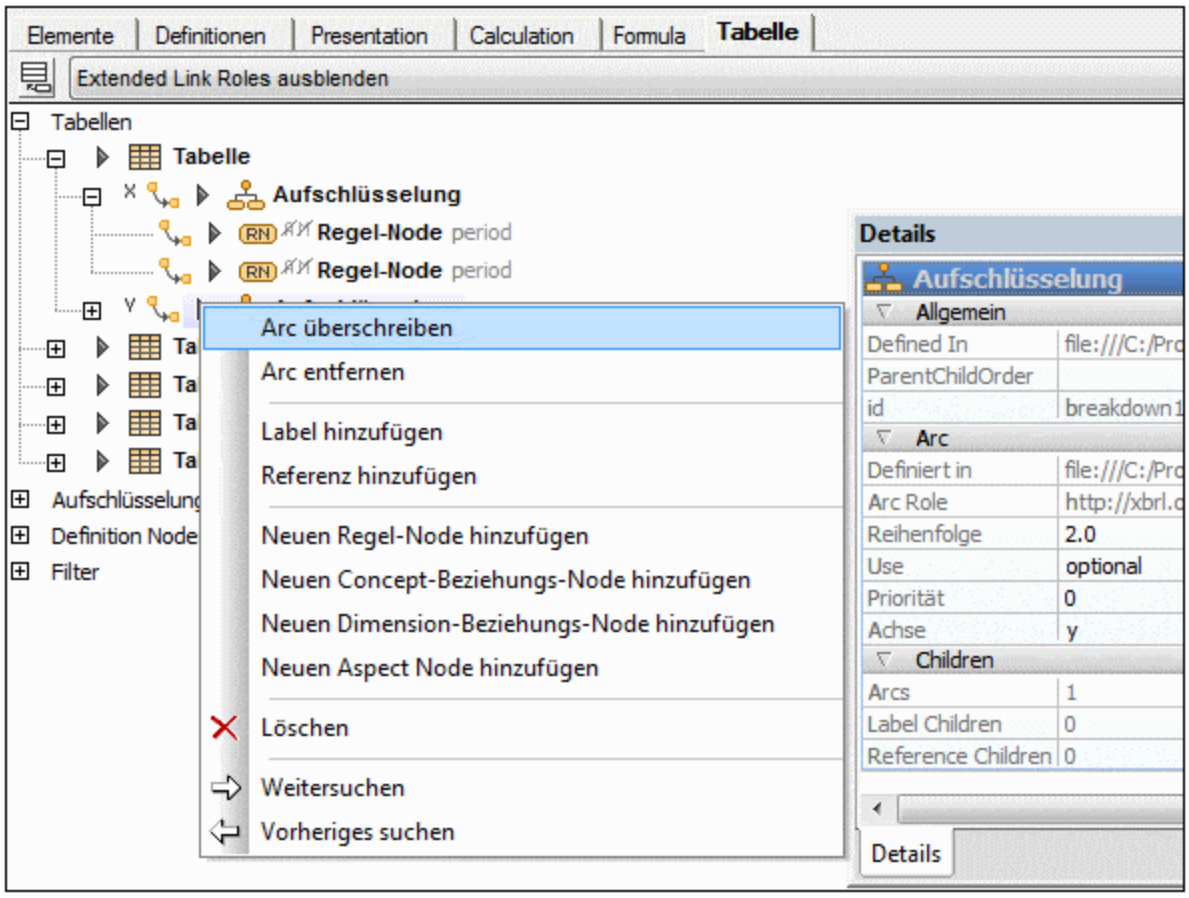
Wenn eine Komponente im Diagramm in reduzierter Ansicht angezeigt wird, wird entweder ihr Name (falls vorhanden) oder der Wert der entsprechenden Standardeigenschaft in Grau neben dem Beschreibungstext zur Komponente angezeigt. Wenn Sie auf die Komponente doppelklicken, wird die Ansicht erweitert. Wenn Sie auf eine Eigenschaft doppelklicken, wird diese im Bearbeitungsmodus angezeigt. Wenn eine Eigenschaft oder Inhalt einen XPath-Ausdruck enthält, wird das Dialogfeld "XPath-Ausdruck bearbeiten" angezeigt.



17.5.5 Tabellenkomponentenbeziehungen

Eine Beziehung zwischen Tabellenkomponenten kann durch Verknüpfung einer Tabellenkomponenten mit einer anderen mittels Drag-and-Drop erstellt werden. Die Reihenfolge der Children einer Parent-Komponente hängt von den Werten der Arc-Eigenschaft `order` ab. Diese Reihenfolge kann durch Verschieben der Children mittels Drag-and-Drop geändert werden. Eine Child-Komponente kann auch auf oder unter eine andere Parent-Komponente gezogen werden, um die Beziehung (einschließlich Ihrer Eigenschaften) zu kopieren oder zu verschieben.

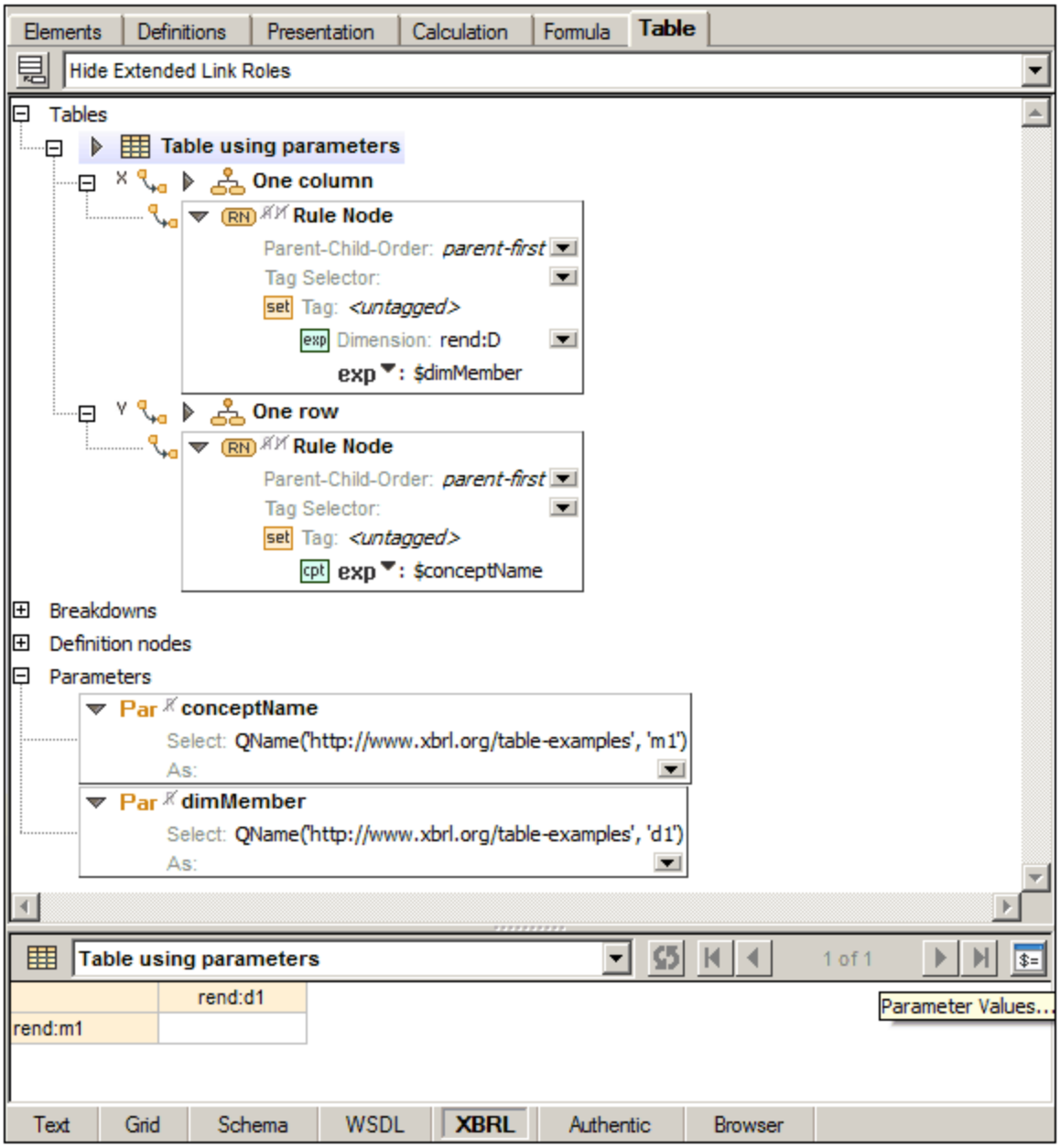
© 2016-2022 Altova GmbH



Wenn eine neue Komponentte über das Kontextmenü einer bestehenden (Parent)-Komponente erstellt wird, wird die Beziehung (d.h. der Arc) automatisch generiert. Mit den Befehlen **Arc überschreiben** und **Arc entfernen** aus dem Kontextmenü einer Child-Komponente können Sie die Beziehung zwischen der Komponente und ihrem Parent überschreiben oder entfernen. Mehrere Arcs von überschriebenen, d.h. außer Kraft gesetzten Beziehungen werden in Subzeilen angezeigt. Die Arcrole von Tabellenkomponentenbeziehungen kann nicht geändert werden.

17.5.6 Tabellenparameter

Mit Hilfe von Tabellenparametern können die Achsen einer Tabelle definiert werden. So wird z.B. in der Abbildung unten die X-Achse der ausgewählten Tabelle durch den Parameter `$dimMember`; definiert. Die Y-Achse wird durch den Parameter `$conceptName` definiert. Die Definitionen der beiden Parameter selbst werden unterhalb der Tabellendefinitionen in der Liste der (globalen) Parameter angezeigt. In der Tabellenlayoutvorschau in unteren Bereich sehen Sie die Tabelle, die generiert wird. Die Achsen werden als Zeile und Spalte der Tabelle erstellt.



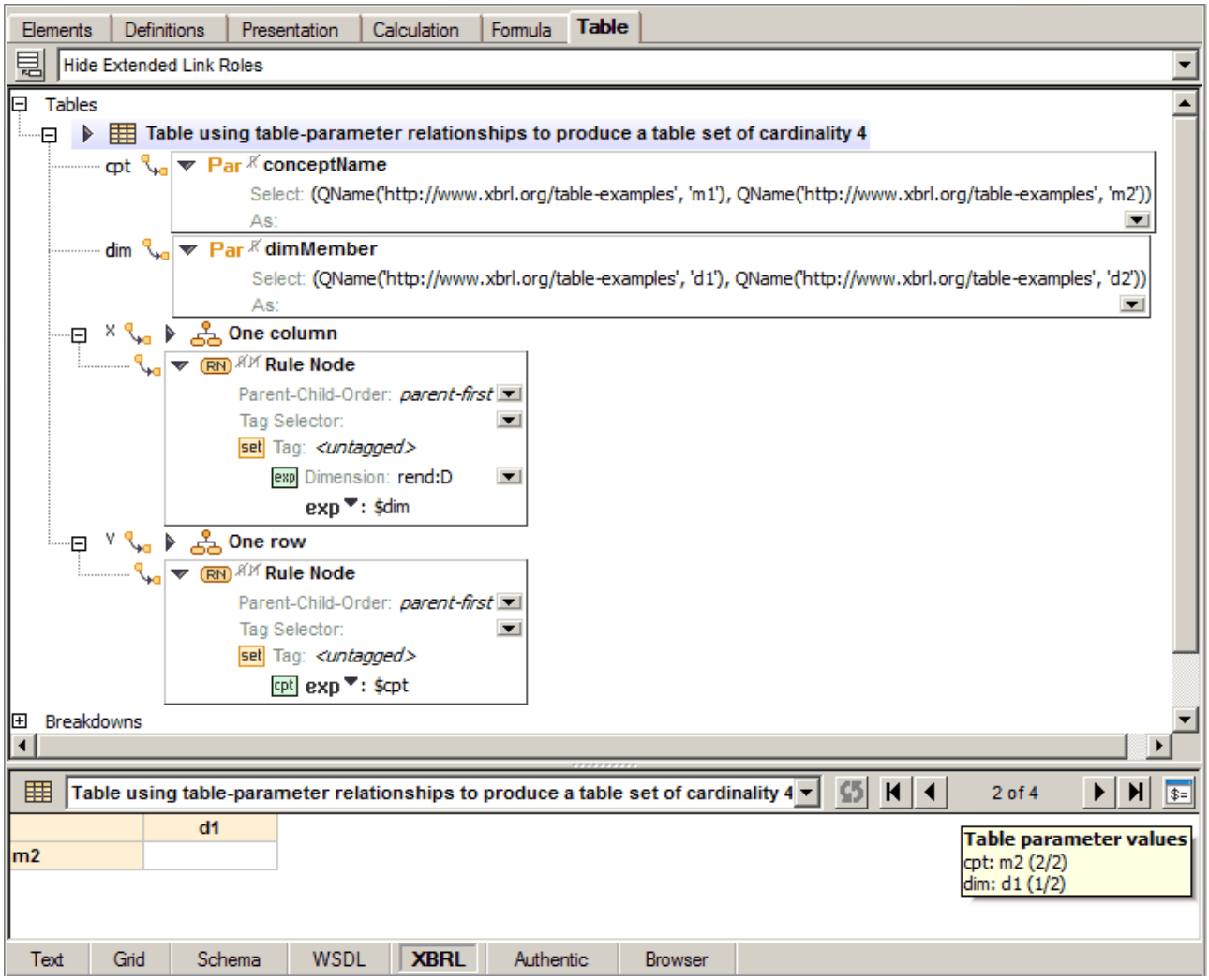
Mit Hilfe von Tabellenparametern können anhand einer einzigen Tabellendefinition mehrere miteinander in Zusammenhang stehende Tabellen erzeugt werden, die eine Tabellengruppe bilden.

- Wenn die Auswertung eines einzigen Parameters eine Wertesequenz ergibt, so enthält die Tabellengruppe für jedes Element in der Ergebnissequenz eine Tabelle.
- Wenn die Tabellendefinition mehrere Parameter hat, so entspricht die Tabellengruppe einem geordneten kartesischen Produkt der durch Auswertung der Parameter erhaltenen Sequenzen. Im Folgenden sehen Sie Beispiele für ein geordnetes kartesisches Produkt:

  ```
  A × B = {1,2} × {3,4} = {(1,3), (1,4), (2,3), (2,4)}
  B × A = {3,4} × {1,2} = {(3,1), (3,2), (4,1), (4,2)}
  ```


© 2016-2022 Altova GmbH

Tabellendefinition mit zwei Tabellenparametern (`conceptName` und `dimMember`), von denen jeder zu einer Sequenz von zwei QNames ausgewertet wird (*siehe XPath-Ausdrücke der Select-Eigenschaft*).



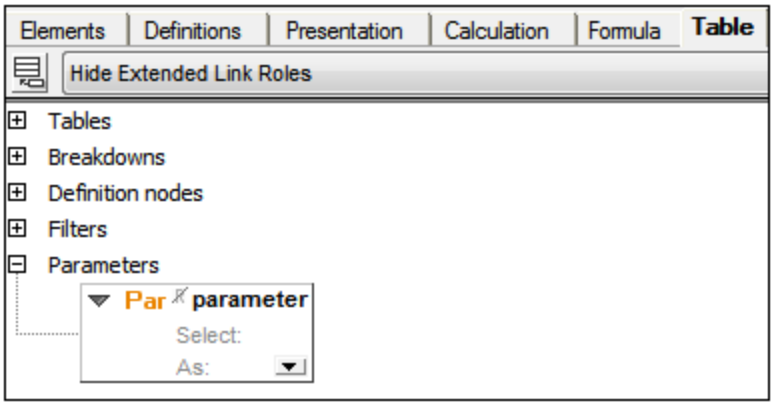
Beachten Sie die folgenden Punkte:

- Die Parameter sind lokale Parameter, die für diese spezifische Tabelle durch Rechtsklick auf die Tabellenkomponente und Auswahl von µ erstellt wurden. Es handelt sich hierbei nicht um globale Parameter wie in der ersten Abbildung oben.
- Das geordnete kartesische Produkt der beiden Sequenzen von zwei QNames erzeugt vier Tabellen: `dimMember × conceptName = {d1,d2} × {m1,m2} = {(d1,m1), (d1,m2), (d2,m1), (d2,m2)}`
- Wenn im Diagramm eine Tabellendefinition, die eine Tabellengruppe beschreibt, ausgewählt wird, werden die Navigationssymbole in der Tabellenlayoutvorschau aktiv und Sie können durch eine Vorschau der Tabellen in der Tabellengruppe navigieren. Die aktuell in der Vorschau angezeigte Tabelle wird in der Symbolleiste durch ihren Index in der geordneten Tabellengruppe angegeben. In der Abbildung oben ist die aktuelle Tabelle Nr. `2 von 4` zu sehen. Die aktuell in der Vorschau angezeigten Tabellenparameterwerte werden ebenfalls in einem Popup angezeigt (*siehe Abbildung*).

- Die Symbolleisten-Schaltfläche "Aktualisieren" der Tabellenlayoutvorschau ist aktiv, wenn die Vorschau nicht mehr mit den aktuellen Definitionen übereinstimmt, z.B. nachdem ein neues Concept hinzugefügt wurde.
- Mit der Symbolleisten-Schaltfläche "Parameterwerte" öffnen Sie das Dialogfeld "XML-Parameterwerte", in dem Sie die Werte und Datentypen aller (globalen und lokalen) Tabellenparameter bearbeiten können.

Definieren von XBRL-Parametern

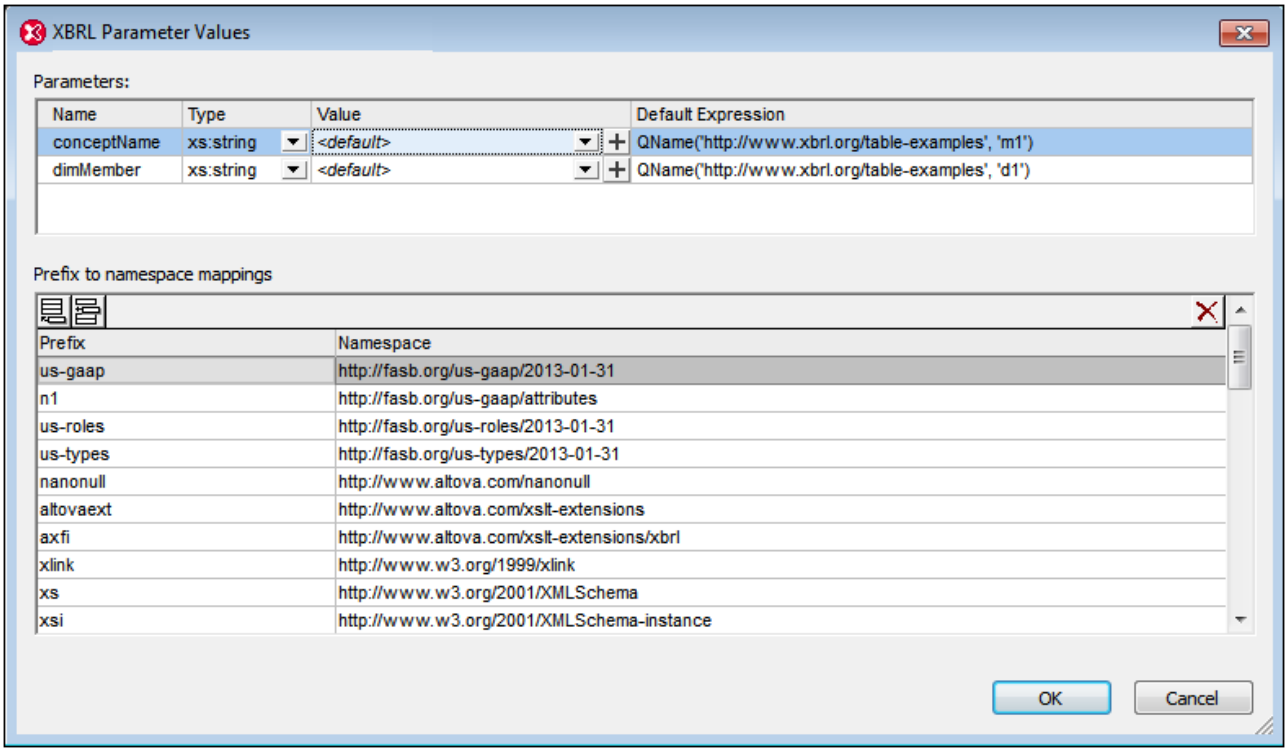
XBRL-Parameter können in XPath-Ausdrücken in Formulas und Tabellendefinitionen verwendet werden. Parameter, die als Formula-Parameter (aus der Formula Linkbase) verwendet werden sollen, werden im Abschnitt "Formula" erstellt, während Tabellenparameter (aus der Table Linkbase) auf dem Register "Tabelle" erstellt werden. Sowohl Formula- als auch Tabellenparameter können lokal oder global sein. Bei lokalen Parametern handelt es sich im Grunde um globale Parameter, die zum Zeitpunkt ihrer Erstellung mit der entsprechenden Komponente (Formula oder Tabelle) verknüpft werden. Lokale Parameter werden durch Rechtsklick auf die Komponente (Formula oder Tabelle) und Auswahl des Befehls **Neuen Parameter hinzufügen** erstellt, während globale Parameter durch Rechtsklick in einen leeren Bereich des entsprechenden Registers und Auswahl des Befehls **Neuen Parameter hinzufügen** erstellt werden. Daraufhin wird ein neuer Parameter namens `parameter` in das Diagramm eingefügt (*in der Abbildung unten sehen Sie einen globalen Parameter*). Um den Parameternamen zu ändern, doppelklicken Sie auf den Namen und bearbeiten Sie ihn.



Jeder Parameter hat ein *Required* Flag. Wenn das Flag gesetzt ist, ist der Parameter obligatorisch, d.h. sein Wert muss von der verarbeitenden Applikation bereitgestellt werden. Wenn der Parameter nicht obligatorisch ist und kein Wert von der verarbeitenden Applikation bereitgestellt wird, kann der Wert mit Hilfe des in der Eigenschaft *Select* angegebenen XPath-Ausdrucks berechnet werden. Doppelklicken Sie in das *Select*-Feld, um einen XPath-Ausdruck einzugeben. Dieser Wert wird als Standardwert für den Parameter verwendet. In der optionalen Eigenschaft *As* wird der vom Parameter benötigte Datentyp definiert. Wählen Sie einen Datentyp aus der Dropdown-Liste der Auswahlliste aus.

Bei Parametern, die als Tabellenparameter verwendet werden, können Sie den Datentyp des Parameters bearbeiten und einen Parameterwert angeben, der den Standardwert außer Kraft setzt. Klicken Sie dazu auf **XBRL | Parameterwerte**. Geben Sie anschließend im Dialogfeld, das angezeigt wird (*Abbildung unten*), einen Parameterwert ein. Dieser Wert setzt den Standardwert außer Kraft. Da Parameter, die als Tabellenparameter verwendet werden, mehrere Werte haben können, können Sie durch Klicken auf das **+** Symbol in der Spalte *Wert* zusätzliche Parameterwerte definieren.

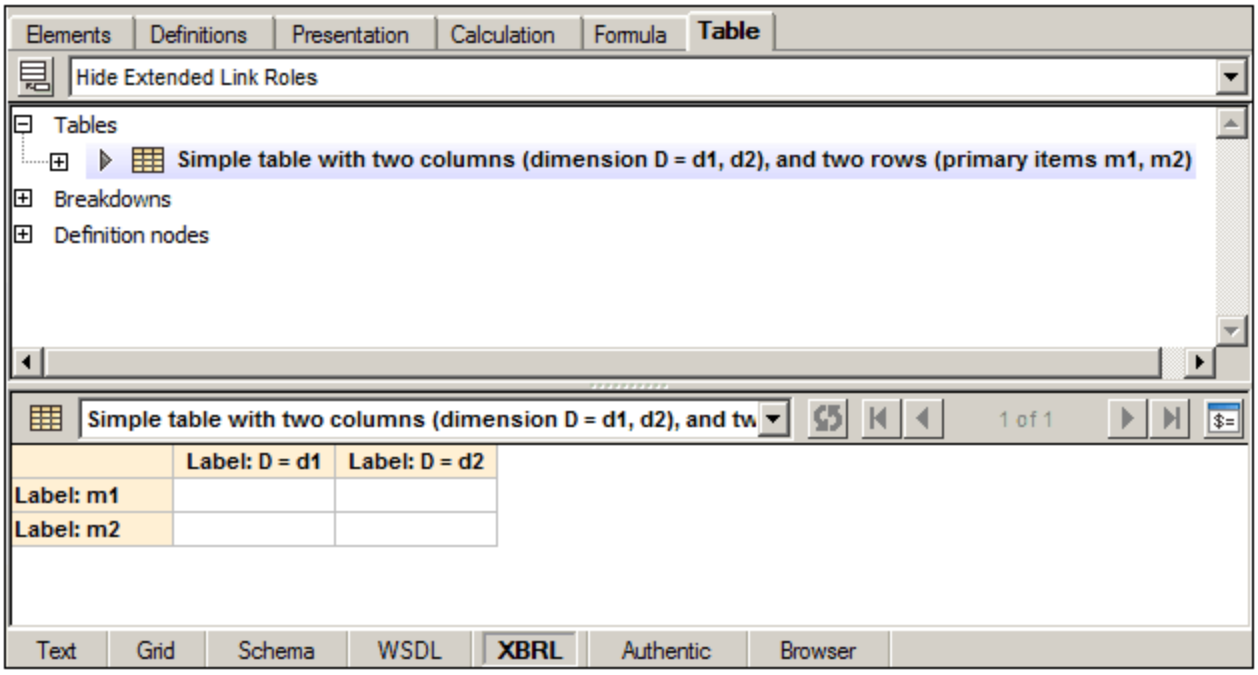
© 2016-2022 Altova GmbH



Die in diesem Dialogfeld zugewiesenen Werte von globalen Parametern werden nur für Tabellenparameter ausgewertet. Werte von in Formulas verwendeten Parametern können in diesem Dialogfeld nicht bearbeitet werden.

17.5.7 Tabellenlayoutvorschau

Das XBRL-Tabellenlayoutvorschau-Fenster befindet sich auf dem Register "Tabelle" unterhalb der Tabellendefinitionsstruktur (*siehe Abbildung unten*). In einer Auswahlliste im Bereich Tabellenlayoutvorschau werden alle Tabellen in der Tabellen-Linkbase der aktiven Taxonomie aufgelistet. Um eine Vorschau eines Tabellenlayout zu sehen, wählen Sie die gewünschte Tabelle in der Vorschau-Auswahlliste aus (*siehe Abbildung unten*). Beachten Sie, dass in der Vorschau nur das Layout angezeigt wird. Die Tabellenzellen werden nicht befüllt, da die XBRL-Taxonomie keine Daten enthält.



In der Tabellenlayoutvorschau können Sie Folgendes tun:

- Tabellenlayouts visualisieren, wobei die Vorschau automatisch aktualisiert wird, wenn Tabellendefinitionen geändert werden
- direkt zur Definition einer Komponente springen, indem Sie auf eine Tabellenzelle klicken. (Umgekehrt können Sie durch Klicken auf eine Komponente in der Tabellendefinitionsstruktur zur Tabellenzelle springen)
- das Dialogfeld "XBRL-Tabellenparameter" (über die Symbolleisten-Schaltfläche "Parameterwerte") aufrufen, um Tabellenparameter zu verwalten

Bearbeitung

Änderungen an Tabellendefinitionen und der Taxonomie werden folgendermaßen behandelt:

- *Tabellenänderungen:* Wenn (auf dem Register "Tabelle" im Diagramm oder über die Eingabehilfe "Details") die Struktur einer Tabellendefinition geändert wird, wird die Layoutvorschau der Tabelle sofort aktualisiert. Die Aktualisierung wird auch durch Änderungen von Parameterdefinitionen oder Parameterwerten ausgelöst.
- *DTS-Änderungen:* In der Tabellenlayoutvorschau können mit Hilfe des Altova XPath-Prozessors XPath-Ausdrücke in Definition Nodes ausgewertet werden. Das XPath-Modell wird erstellt, wenn ein Taxonomieschema in die XBRL-Ansicht geladen wird und wird bei der Validierung aktualisiert. Wenn das zugrunde liegende DTS geändert wird (z.B. durch Bearbeiten eines Concept oder einer Linkbase), so zeigt die Tabellenvorschau nicht mehr den Zustand des geänderten DTS an. Das Symbol *Tabelle stimmt nicht mehr mit DTS überein* in der Symbolleiste der Vorschau zeigt diesen Status an. Im Tooltipp des Symbols finden Sie den Hinweis: *Die Vorschau muss über die Schaltfläche "Aktualisieren" manuell aktualisiert werden*. Mit dem Befehl **Aktualisieren** wird das DTS erneut eingelesen, was einer vollständigen Validierung der Taxonomie entspricht.

	Tabelle stimmt nicht mehr mit DTS überein

© 2016-2022 Altova GmbH

	*Tabelle aktualisieren*

Fehlerbehandlung

Fehler, die im Zusammenhang mit der Tabellenlayoutvorschau auftreten, werden folgendermaßen behandelt:

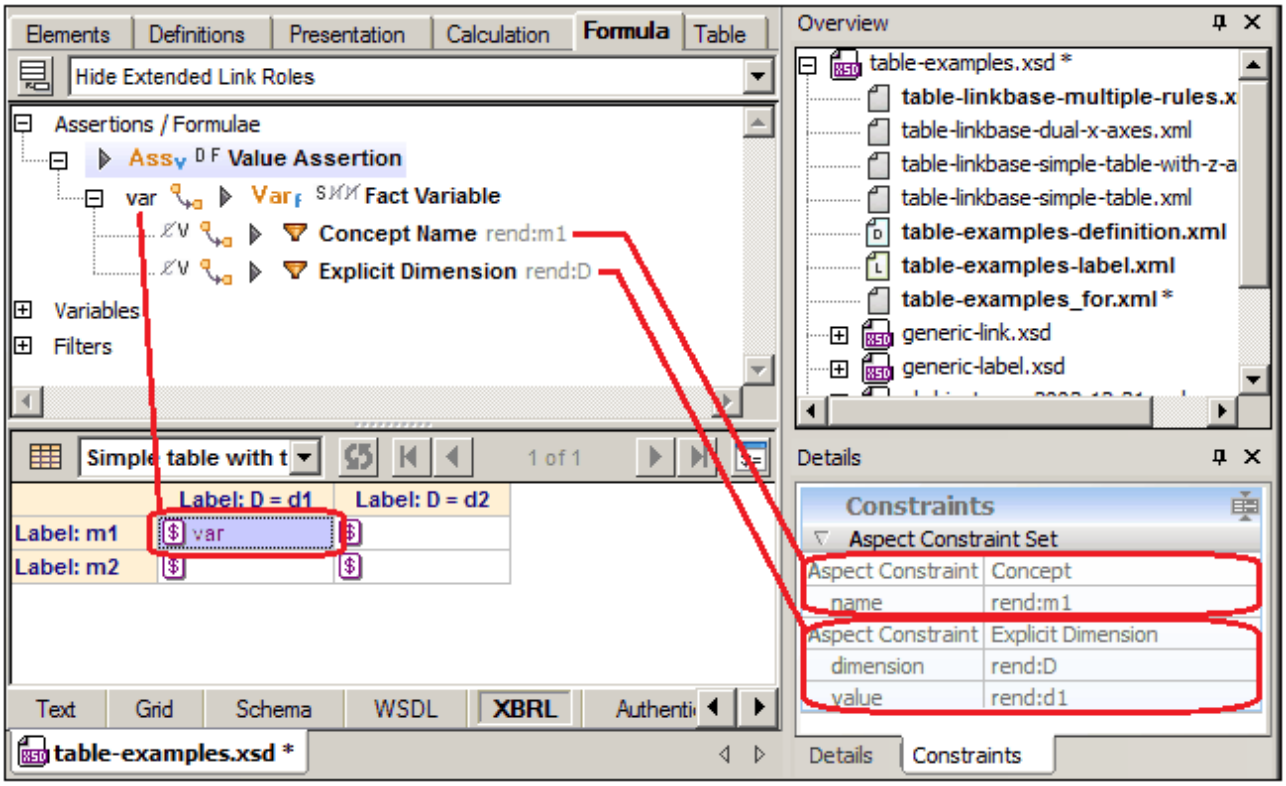
- *Ungültige Ausdrücke in Tabellendefinitions-Nodes:* Wenn ein Tabellendefinitions-Node einen XPath-Ausdruck enthält, der nicht aufgelöst werden kann, wird die Überschrift der entsprechenden Layoutzelle rot angezeigt. In diesem Fall wird der ungültige Aspect Constraint in der Eingabehilfe "Details" auf dem Register "Constraints" markiert.
- *Nicht auflösbare Beziehungs-Nodes:* Wenn ein Beziehungs-Node 912 aufgrund ungültiger Eigenschaften oder eines ungültigen DTS nicht aufgelöst werden kann, wird im Layout eine Platzhalterzelle mit dem markierten Fehlertext angezeigt.
- *Zusammengeführte Regel-Nodes ohne Child Nodes:* Wenn ein zusammengeführter Regel-Node 912 keine Child Nodes hat, wird im Layout eine Platzhalterzelle mit dem markierten Fehlertext angezeigt.
- *Ungültiges DTS:* Wenn die Taxonomie beim Laden eines Taxonomieschemas in den XBRL-Taxonomie-Editor oder nach der Validierung ungültig ist, steht das XPath-Modell nicht zur Verfügung. Für die Tabellenlayoutvorschau wird über die Symbolleisten-Schaltfläche *Tabelle stimmt nicht mehr mit DTS überein* ein Fehler angezeigt. Das Layout kann trotzdem bis zu einem gewissen Grad erstellt werden, mit der Einschränkung, dass XPath-Ausdrücke nicht ausgewertet werden können. Der Tooltipp der Symbolleisten-Schaltfläche gibt Aufschluss darüber, wie das Problem gelöst werden kann (d.h. durch Behebung des Validierungsfehlers und Neuvalidierung der Taxonomie).

17.5.7.1 Erstellen von Formulas in der Tabellenlayoutvorschau

Zur Unterstützung der Erstellung von Fact-Variablen unter Variablengruppen (d.h. unter Formulas oder Value/Existence Assertions) wird die Tabellenlayoutvorschau auch auf dem Register "Formula" angezeigt (*siehe Abbildung unten*). In diesem Fall können die Zellen innerhalb der Tabellenachsen nicht ausgewählt werden, da die dazugehörigen Tabellendefinitions-Nodes in der Formula Linkbase nicht angezeigt werden. Für Datenzellen wird jedoch ein Symbol **Fact-Variable hinzufügen** angezeigt, welches aktiv ist, sobald die Variablengruppe in der Formula-Struktur ausgewählt ist (*siehe Abbildung unten*).

 *Fact-Variable zur ausgewählten Formula oder Assertion hinzufügen*

Um eine Fact-Variable zu einer Variablengruppe hinzuzufügen, wählen Sie die Variablengruppe auf dem Register "Formula" aus. (Eine Variablengruppe ist eine Formula oder Value/Existence Assertion. Die ausgewählte Variablengruppe in der Abbildung unten ist eine Value Assertion). In den Zellen der Tabellenlayoutvorschau ist das Symbol **Fact-Variable hinzufügen** daraufhin aktiv. Klicken Sie auf das Symbol, um die Variable zur Variablengruppe hinzuzufügen. Bei der Ausführung wird unter der ausgewählten Variablengruppe (d.h. Formula oder Assertion) eine neue Fact-Variable erstellt, die einen entsprechenden Filter für die einzelnen durch die Datenzelle definierten Aspect Constraints enthält.



Wenn die ausgewählte Formula oder Assertion eine Variable hat, die Filter enthält, die mit den Aspect Constraints einer Datenzelle in der Vorschau übereinstimmen, so wird der Variablenname von der Datenzelle angezeigt. Dies sollte v.a. nach Erstellung einer neuen Fact-Variablen über die Schaltfläche **Fact-Variable hinzufügen** der Fall sein (*siehe Abbildung oben*).

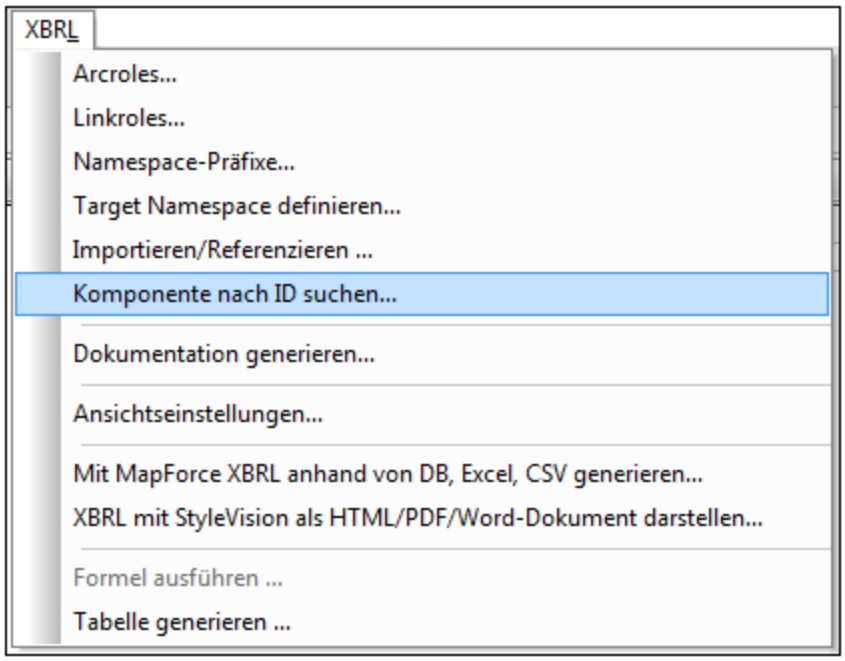
17.5.8 Suchen von Tabellenkomponenten

Tabellenkomponenten können (i) anhand Ihrer IDs und (ii) durch Navigation durch die Instanzen der Komponente im Dokument gesucht werden.

Suchen einer Tabellenkomponente anhand ihrer ID

In Taxonomien mit großen Formula- oder Tabellen-Linkbases, die mehrere Komponenten derselben Art enthalten (z.B. Assertions, Filter, Tabellen), empfiehlt sich die Suche einer Komponente anhand ihrer ID. Mit dem Menübefehl **XBRL | Komponente nach ID suchen** können Sie eine Suche nach der ID durchführen.

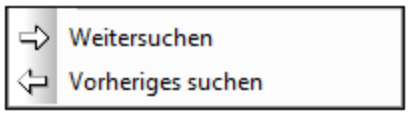
© 2016-2022 Altova GmbH



Bei Aufruf des Befehls wird ein Dialogfeld angezeigt, in das Sie die gewünschte ID eingeben müssen.

Suchen von Komponenteninstanzen

Die meisten Tabellenkomponenten werden mehrmals im Tabellen-Linkbase-Diagramm angezeigt: (i) die Definition, die sich direkt unterhalb dem entsprechenden Abschnitts-Node befindet und (ii) alle Referenzen auf die Komponente (über Beziehungen). Mit den Befehlen **Weitersuchen** und **Vorheriges suchen** aus dem Kontextmenü der Komponente (*Abbildung unten*) können Sie zu allen Stellen navigieren, an denen diese Tabellenkomponente referenziert wird.



Diese Befehle können auch über die Symbolleistenschaltflächen (*Abbildung unten*) aufgerufen werden.



Wenn die Definition der Komponente erreicht wurde, wird eine entsprechende Meldung angezeigt.

17.6 XULE

XULE (aus XBRL Rule) ist eine Sprache für die Abfrage von XBRL-Berichten und Taxonomien. Hauptsächlich dient diese Sprache dazu, Berichte vor ihrer Einreichung bei den zuständigen Stellen zum Zweck der Qualitätssicherung abzufragen und zu überprüfen. Mit Hilfe von XULE können Berichte auf zwei Arten überprüft werden:

- *Um anhand der Berichtsdaten Ausgabedaten zu liefern:* Durch Abfrage der Daten im Bericht und Berechnung von Ergebnissen anhand der Berichtsdaten. Die Qualität der Ausgabe kann beurteilt werden.
- *Erstellung von Assertions:* Die Daten im Bericht können anhand dieser Assertions überprüft werden. Auf Basis der Ergebnisse können in der Folge die entsprechenden Aktionen durchgeführt werden.

Internet Links

- XULE Homepage (enthält eine kurze Übersicht über XULE)
- Dokumentation der XULE-Sprachsyntax

XMLSpy-Funktionalitäten

XMLSpy bietet die folgenden XULE-Funktionalitäten:

- einen **integrierten XULE-Prozessor**, der XULE-Dokumente und -Ausdrücke anhand eines XBRL-Instanzdokuments verarbeitet.
- Erstellung von XULE-konformen Dokumenten
- Validierung von XULE-Dokumenten zur Überprüfung der Syntax anhand der XULE-Spezifikation
- Syntaxfärbung in XULE-Dokumenten
- Autokomplettierung von XULE-Sprachkonstrukten bei der Bearbeitung von XULE-Dokumenten
- Ein spezielles **XULE-Fenster** zur interaktiven Abfrage von XBRL-Instanzdokumenten
- Verarbeitung einer XBRL-Instanz anhand eines einzigen XULE-Dokuments oder einer Gruppe von in einer ZIP-Datei gespeicherten XULE-Dokumenten; die Verarbeitung kann mit dem XMLSpy-Prozessor oder dem RaptorXML(+XBRL) Server[1088]-Prozessor durchgeführt werden
- Integration der Ausführung von XULE-Dokumenten und Dokumentenmappen in XMLSpy-Projekte[1080]

Diese Funktionalitäten werden in den Unterabschnitten dieses Abschnitts näher beschrieben.

RaptorXML+XBRL Server von Altova

Das Altova-Produkt RaptorXML+XBRL Server ermöglicht eine schnelle benutzerdefinierte XULE-Verarbeitung, wodurch Sie XULE-Dokumente mittels Skripts und einer Reihe von Server und Prozessor-APIs, darunter einer leistungsstarken Python API, über die Befehlszeile verarbeiten können. Nähere Informationen dazu finden Sie unter:

- Die RaptorXML+XBRL-Seite auf der Altova Website
- DQC-Zertifizierung von Altova RaptorXML+XBRL
- Produktdokumentation zu RaptorXML+XBRL
- Dokumentation zur RaptorXML Python API

© 2016-2022 Altova GmbH

17.6.1 XULE-Dokumente

XMLSpy bietet eine Reihe von Funktionalitäten für die Erstellung, Validierung und Ausführung von XULE-Dokumenten. Diese Funktionen sind in diesem Abschnitt beschrieben.

XULE-konforme Dateien (.xule-Dateien)

Der Dateityp **.xule** ist in XMLSpy als XULE-konform vordefiniert, d.h. wenn eine **.xule**-Datei in XMLSpy geöffnet wird, stehen XULE-Bearbeitungshilfen in Form von Syntaxfärbung und Autokomplettierungsvorschlägen zur Verfügung. Im Abschnitt "Dateiarten" des Dialogfelds "Optionen" 1606 können Sie zusätzlich dazu weitere Dateierweiterungen als XULE-konform definieren.

XULE-Dokumentenmappen

Mehrere XULE-Dokumente können in ein ZIP-Archiv (normalerweise `.zip`) verpackt werden. Bei diesem ZIP-Archiv handelt es sich um eine XULE-Dokumentenmappe (oder XULE Ruleset). Sie können die gesamte XULE-Dokumentenmappe anschließend an einer XBRL-Instanz ausführen, indem Sie die ZIP-Datei als die auszuführende XULE-Datei 945 definieren.

Ein ZIP-Archiv kann eine beliebige Struktur haben. Bei der XULE-Ausführung werden alle XULE-Dateien auf jeder Ebene des Archivs verwendet. Nicht-XULE-Dateien werden herausgefiltert.

Syntaxfärbung

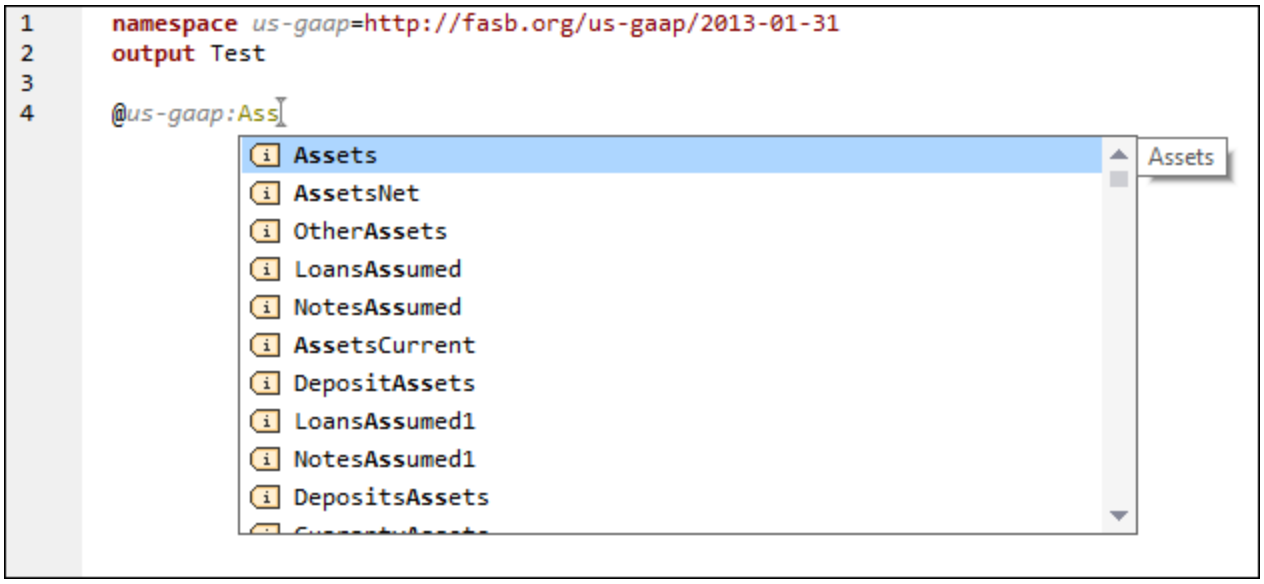
XULE-Dokumente können in der Textansicht von XMLSpy bearbeitet werden. In der Abbildung unten sehen Sie die Standardsyntaxfärbung eines XULE-Beispieldokuments. Sie können die Syntaxfärbung im Abschnitt "Schriftarten und Farben | Textansicht" 1623 des Dialogfelds "Optionen" anpassen.

```
1   namespace us-gaap=http://fasb.org/us-gaap/2013-01-31
2
3   // Balance sheet test
4   assert balance-test unsatisfied
5
6   @us-gaap:Assets#assets == @us-gaap:LiabilitiesAndStockholdersEquity#liabilities
7
8   message
9
10  "Balance sheet is unbalanced in period {$assets.period}, {$assets} != {$liabilities}"
```

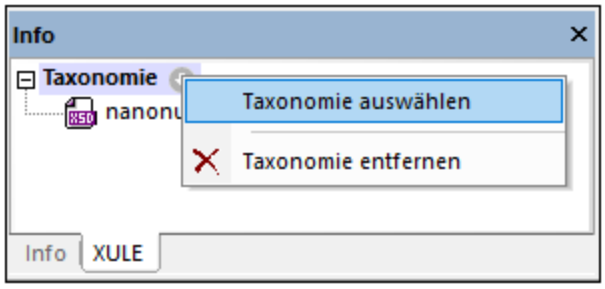
Autokomplettierung

Bei der Eingabe von Regeln in das XULE-Dokument stehen Ihnen zwei Arten von Autokomplettierungshilfen zur Verfügung:

- Hilfe im Zusammenhang mit der XULE-Sprachsyntax
- Hilfe im Zusammenhang mit der Struktur der ausgewählten XBRL-Taxonomie



Sie können die gewünschte Taxonomie im Info-Fenster auswählen (*siehe Abbildung unten*). Klicken Sie auf dem Register "XULE" auf das Symbol rechts vom Eintrag *Taxonomie* und klicken Sie im Menü, das daraufhin angezeigt wird, auf **Taxonomie auswählen**. Navigieren Sie anschließend zur gewünschten Taxonomie und wählen Sie sie aus. Es kann immer nur eine Taxonomie hinzugefügt werden. Wenn Sie eine neue Taxonomie hinzufügen, ersetzt diese die vorherige Taxonomie. Alternativ dazu können Sie eine Taxonomie auch entfernen, bevor Sie eine neue hinzufügen (*siehe Abbildung unten*).



Anmerkung: Sie müssen die Namespace-Deklarationen aller benötigten Taxonomiekomponenten zum XULE-Dokument hinzufügen (*siehe Abbildung XULE-Dokument oben*). Die Namespace-Präfixe müssen nicht mit dem der Taxonomie übereinstimmen, doch empfiehlt es sich, dasselbe Namespace-Präfix zu verwenden, um Verwirrung zu vermeiden. Wenn Sie keine Namespace-Deklarationen hinzufügen, funktioniert die Autokomplettierungsfunktion der Taxonomiekomponenten nicht.

Validierung von XULE

Mit Hilfe des Befehls **XML | Validieren (F8)** kann ein XULE-Dokument anhand der XULE-Sprachspezifikation auf korrekte Syntax überprüft werden.

Integration in XMLSpy-Projekte

Sie können XULE-Dokumente auf folgende Art in ein XMLSpy-Projekt integrieren:

1. Fügen Sie das XULE-Dokument zu einem geeigneten Ordner in einem XMLSpy-Projekt[1080] hinzu.

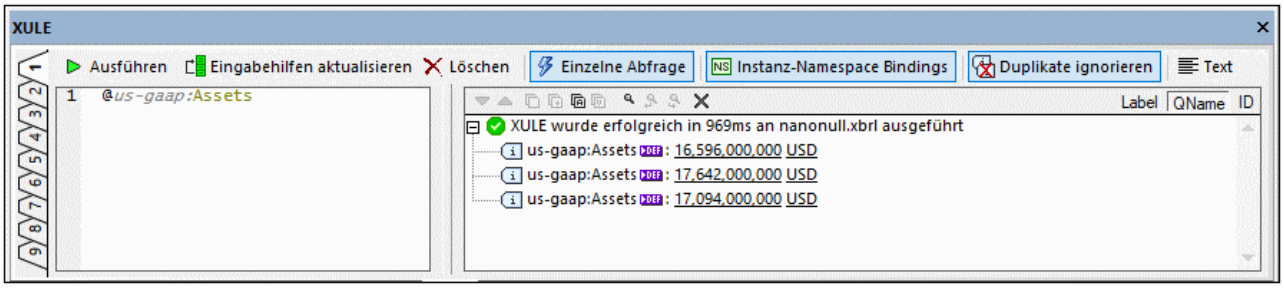
© 2016-2022 Altova GmbH

2. Geben Sie in den Projekteigenschaften dieses Ordners[1344] die XBRL-Instanz ein, an der die XULE-Datei ausgeführt werden soll.
3. Klicken Sie mit der rechten Maustaste auf das XULE-Dokument im Projekt und wählen Sie die Befehl **XBRL | XULE ausführen**. Daraufhin wird das XULE-Dokument an der für den Ordner definierten XBRL-Instanz ausgeführt und die Ergebnisse werden im Fenster "Meldungen"[122] oder einem neuen Dokument angezeigt *(siehe XULE-Ausführungsoptionen[1644])*.

17.6.2 Das Fenster "XULE"

Das Fenster "XULE" *(Abbildung unten)* ist ein Ausgabefenster[132]. Sie können das aktive XBRL-Instanzdokument über dieses Fenster interaktiv abfragen und das Ergebnis Ihrer Abfrage sehen.

Des Fenster "XULE" befindet sich standardmäßig unterhalb des Hauptfensters im unteren Bereich der XMLSpy-Benutzeroberfläche[115]. Es verfügt über neun Register, von denen jedes in zwei Bereiche gegliedert ist: (i) einen Bereich für XULE-Ausdrücke, in den Sie den XULE-Ausdruck (oder die XULE-Regel), den Sie am aktiven Dokument ausführen möchten, eingeben und einen (ii) Ergebnisbereich, in dem das Ergebnis der Ausführung angezeigt wird.



Um einen XULE-Ausdruck am aktiven XBRL-Instanzdokument interaktiv auszuführen, gehen Sie folgendermaßen vor:

1. Machen Sie das gewünschte XBRL-Instanzdokument im Hauptfenster zum aktiven Dokument.
2. Geben Sie den XULE-Ausdruck in den XULE-Ausdrucksbereich ein (*linke Seite*). Zu den Bearbeitungsfunktionen des Ausdrucksfensters gehören Syntaxfärbung und Autokomplettierung.
3. Klicken Sie in der Symbolleiste des Fensters auf **Ausführen**, um den Ausdruck auszuführen.
4. Die Ergebnisse des Ausdrucks werden im Ergebnisbereich (*rechte Seite*) angezeigt. Wenn Sie auf einen Link in den Ergebnissen klicken, gelangen Sie zum entsprechenden Node im XBRL-Instanzdokument.

Anmerkung: Die Syntaxfärbung für XULE kann im Dialogfeld Optionen[1602] (im Abschnitt Schriftarten und Farben | Textansicht[1623]) angepasst werden. Informationen zur Autokomplettierung finden Sie unter der Beschreibung des Symbolleisten-Befehls **Eingabehilfen aktualisieren** weiter unten.

Symbolleiste: Befehle und Optionen

Die Symbolleiste des Fensters "XULE" enthält Befehle und nützliche Optionen zum Erstellen und Ausführen von XULE-Ausdrücken. Eine Beschreibung dazu finden Sie weiter unten.

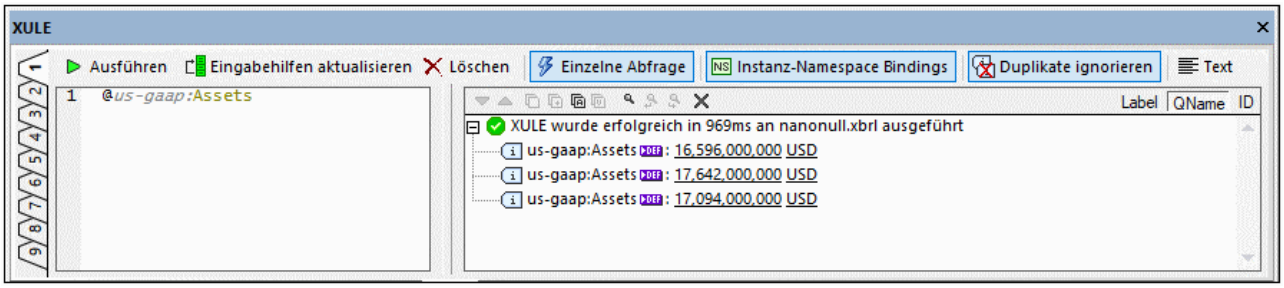
Ausführen
Klicken Sie auf **Ausführen**, um den XULE-Ausdruck auszuführen.

Eingabehilfen aktualisieren
Im XULE-Ausdrucksbereich stehen Ihnen bei der Eingabe zwei Arten von Autokomplettierungshilfen zur Verfügung: (i) die Autokomplettierung im Zusammenhang mit der XULE-Syntax und (ii) die Autokomplettierung im Zusammenhang mit der Struktur des aktiven XBRL-Instanzdokuments. Um jedoch die Struktur der XBRL-Instanz zu kennen, muss im XULE-Fenster die von der XBRL-Instanz referenzierte XBRL-Taxonomie ausgelesen werden. Klicken Sie auf **Eingabehilfen aktualisieren,** um die mit der XBRL-Instanz verknüpfte Taxonomie im Hintergrund zu laden. Nachdem die Taxonomie geladen wurde, steht die Autokomplettierungshilfe für die Dokumentstruktur zur Verfügung und diese Symbolleisten-Schaltfläche (die für dieses XBRL-Instanzdokument nicht mehr benötigt wird) wird deaktiviert. Beachten Sie auch, dass die Taxonomie auch jedes Mal, wenn Sie auf **Ausführen** klicken, ausgelesen wird.

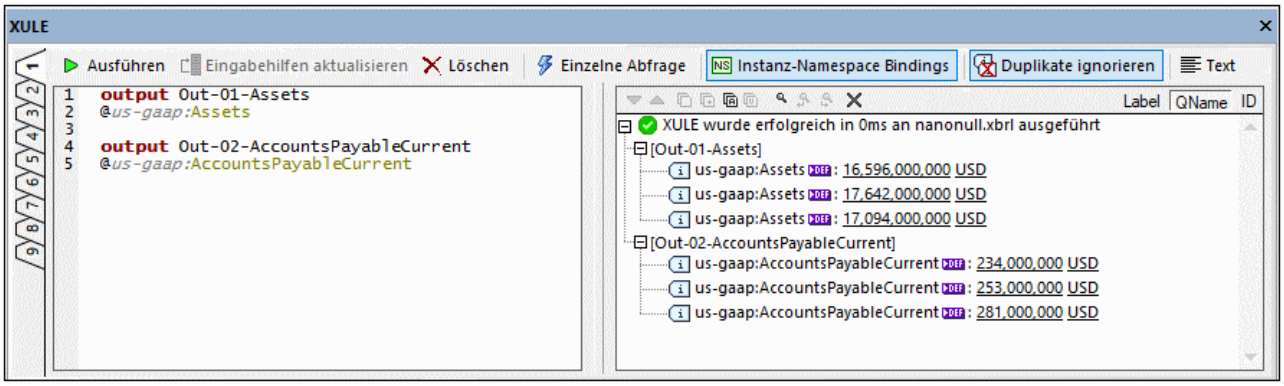
Löschen
Klicken Sie auf **Löschen**, um den Ausdruck zu löschen.

Einzelne Abfrage
Der Einzelabfragemodus ist eine XULE-Fenster-spezifische Funktion in XMLSpy. Sie können damit: (i) einen Ausdruck ohne das Schlüsselwort `output` und als eine einzige Abfrage eingeben und (ii) das Ergebnis in Form einer einzigen Ausgabe generieren. Damit die Abfrage gemäß der XULE-Syntax gültig ist, wird das Schlüsselwort `output` benötigt. Wenn Sie das XBRL-Dokument jedoch interaktiv und schnell abfragen möchten, ist es von Vorteil, eine einfache Abfrage ohne das Schlüsselwort `output` eingeben zu können.

In den beiden Abbildungen unten sehen Sie, wie Sie einzelne und mehrere Abfragen verwenden.



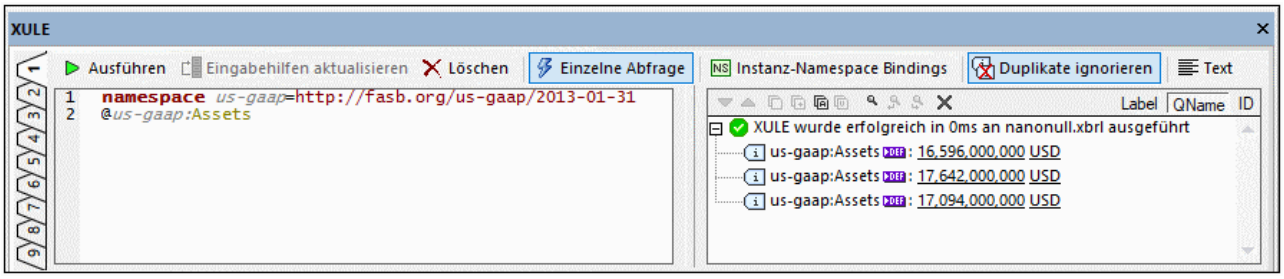
Einzelabfragemodus aktiviert: kein 'output'-Schlüsselwort erforderlich



Einzelabfragemodus deaktiviert: mehrere Abfragen mit dem Schlüsselwort 'output', erzeugt mehrere Ausgaben.

© 2016-2022 Altova GmbH

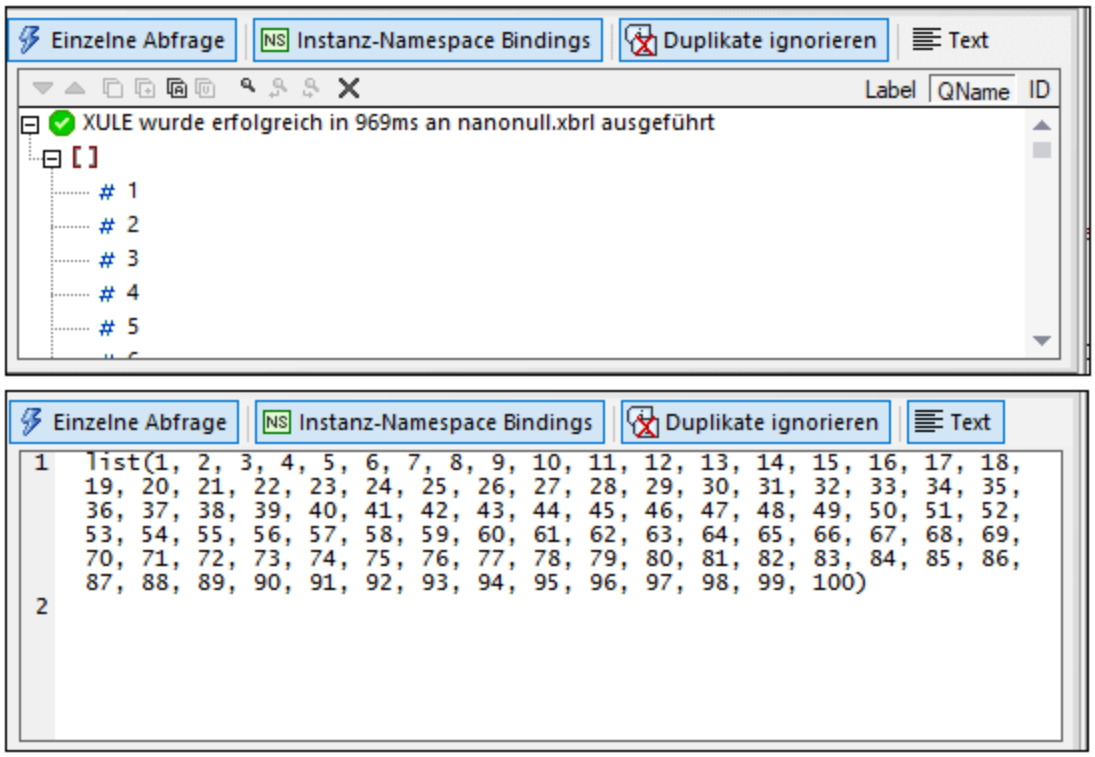
Instanz-Namespace Bindings
Wenn die Option **Instanz-Namespace Bindings** aktiviert ist, müssen keine Namespaces in der XULE-Abfrage deklariert werden; Die Namespace-Präfixe werden an die in der XBRL-Instanz deklarierten Namespace URIs gebunden. So ist etwa das Namespace-Präfix in der Abbildung oben an den im Instanzdokument dafür definierten Namespace gebunden. Wenn die Option **Instanz-Namespace Bindings** hingegen deaktiviert ist, müssen Sie den Namespace (mit dem Schlüsselwort `namespaces`, wie in der Abbildung unten gezeigt) in der XULE-Abfrage deklarieren.



Diese Option ist nützlich, da Sie sich dadurch das Ausfüllen der Namespace-Deklarationen in der XULE-Abfrage sparen.

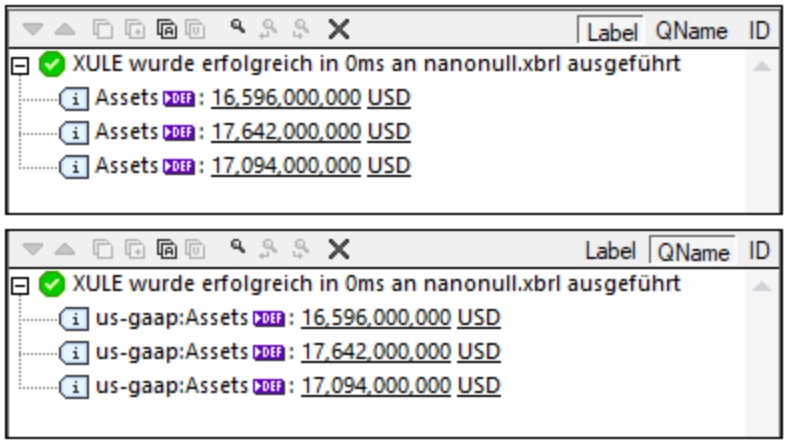
Duplikate ignorieren
Ein doppelt vorhandenes Fact kommt - meist in Inline XBRL-Dateien - dann vor, wenn dasselbe Fact mehr als einmal im HTML-Code gefunden wird. Mit der Option **Duplikate ignorieren** wird festgelegt, dass das duplizierte Fact nur einmal ausgegeben wird.

Text
Mit der Option **Text** wechseln Sie im Ergebnisbereich zwischen der Ausgabe als Text und Struktur. So sehen Sie etwa in den Abbildungen unten die Ausgabe in Struktur- (*links*) und Textform (*rechts*). Die Abfrage lautet: `list(for $i in range(100) $i).`



Der Ergebnisbereich

Ergebnisse können mit ihren Labels, QNames oder IDs angezeigt werden. Wählen Sie die gewünschte Option in der Symbolleiste des Ergebnisbereichs entweder vor oder nach Ausführung der Abfrage aus. In der Abbildung unten sehen Sie die Ergebnisse mit Labels (*links*) und QNames (*rechts*).



Die Symbolleiste des Ergebnisbereichs enthält Schaltflächen für die Navigation, Suche sowie Kopierfunktionen. In der nachstehenden Tabelle finden Sie eine Beschreibung dieser Schaltflächen von links nach rechts. Die entsprechenden Befehle stehen auch über das Kontextmenü von Ergebnislisteneinträgen zur Verfügung.

© 2016-2022 Altova GmbH

Schaltfläche	Funktion
Weiter, Zurück	Wählt jeweils den nächsten bzw. vorherigen Eintrag in der Ergebnisliste aus.
Ausgewählte Nachricht kopieren	Kopiert den ausgewählten Ergebniseintrag in die Zwischenablage.
Ausgewählte Nachricht und alle Sub-Einträge davon kopieren	Kopiert den ausgewählten Ergebniseintrag sowie alle Subeinträge davon in die Zwischenablage. Jeder Eintrag wird als separate Zeile kopiert.
Alle Nachrichten kopieren	Kopiert alle Ergebniseinträge in die Zwischenablage.
Wert der ausgewählten Zeile in die Zwischenablage kopieren	Kopiert nur den Wert des ausgewählten Ergebniseintrags in die Zwischenablage.
Suchen	Öffnet das Dialogfeld *Suchen*, um nach einem String in den Ergebnissen zu suchen.
Vorheriges suchen	Sucht die vorhergehende Instanz des zuletzt im Dialogfeld "*Suchen*" eingegebenen Begriffs.
Weitersuchen	Sucht die nächste Instanz des zuletzt im Dialogfeld "*Suchen*" eingegebenen Begriffs.
Zurücksetzen	Löscht die Ergebnisliste.

17.6.3 XULE-Ausführung

Um ein XULE-Dokument[939] an einem XBRL-Instanzdokument auszuführen, wählen Sie den Menübefehl **XBRL | XULE ausführen**[1552] aus. Der Befehl kann in den folgenden Fällen verwendet werden:

- Wenn ein XULE-Dokument das aktive Dokument ist, werden Sie bei Auswahl des Befehls aufgefordert, die XBRL-Instanz, an der das XULE-Dokument ausgeführt werden soll, auszuwählen.
- Wenn ein XBRL-Instanzdokument das aktive Dokument ist, werden Sie bei Auswahl des Befehls aufgefordert, das XULE-Dokument oder die XULE-Dokumentenmappe dafür auszuwählen.
- Wenn (i) sowohl das XULE-Dokument (`.xule`-Datei) - oder die XULE-Dokumentenmappe[939] (ZIP-Archiv) - als auch das XBRL-Instanzdokument Teil eines XMLSpy-Projekts[1080] sind, und (ii) die XBRL-Instanzdatei in den Eigenschaften des XMLSpy-Projekts[1344] als XBRL-Zieldatei definiert wurde, dann klicken Sie mit der rechten Maustaste im XMLSpy-Projektfenster auf die XULE-Datei des Projekts und wählen Sie den Befehl **XULE ausführen**. Daraufhin wird/werden das/die XULE-Dokument(e) an der XBRL-Datei, die im Projekt als Ziel für die XULE-Ausführung definiert ist, ausgeführt.

Außerdem können Sie XMLSpy so konfigurieren, dass von der Benutzeroberfläche von XMLSpy aus RaptorXML(+XBRL) Server[1088]-Befehle, darunter auch XULE-Verarbeitungsbefehle, ausgeführt werden.

XULE-Ausführungsoptionen

Es stehen die folgenden XULE-Ausführungsoptionen zur Verfügung:

- Die Ausgabe kann entweder (i) an das Fenster "Meldungen" oder (ii) ein neues in einem neuen XMLSpy-Fenster angezeigtes Dokument gesendet und temporär im Arbeitsspeicher gespeichert werden. Dieses Dokument kann mit dem Befehl **Datei | Speichern unter**[1286] in einer Datei gespeichert werden.
- Doppelt vorhandene Facts beziehen sich auf mehrere Referenzen im selben Fact. (Ein doppelt vorhandenes Fast kommt meist in Inline XBRL-Dokumenten vor, wenn dasselbe Fact aufgrund des HMTL-Codes öfter als einmal aufscheint.) Sie haben die Möglichkeit auszuwählen, dass doppelt vorhandene Facts nur einmal ausgegeben werden.

Sie können die gewünschten Einstellungen auf dem Register XBRL XULE des Dialogfelds "Optionen"[1644] (**Extras | Optionen | XBRL | XULE**) vornehmen.

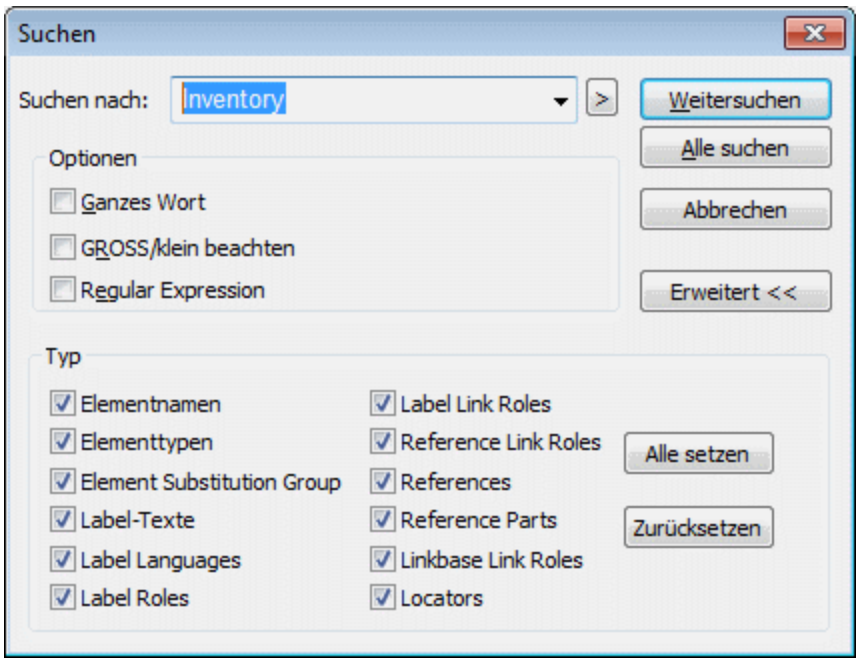
© 2016-2022 Altova GmbH

17.7 In XBRL suchen

XBRL-Taxonomien können in der XBRL-Ansicht mit dem XMLSpy-Befehl "In XBRL suchen" gesucht werden, der aktiv ist, wenn eine XBRL-Taxonomie in der XBRL-Ansicht aktiv ist. Dieser Befehl kann auf die folgenden Arten aufgerufen werden:

- Über den Befehl **Bearbeiten | Suchen,** wenn eine XBRL-Taxonomie in der XBRL-Ansicht aktiv ist.
- Über die Schaltfläche **Suchen** im Fenster "In XBRL suchen".
- Durch Drücken von **Strg+F**.

Bei Auswahl einer dieser Methoden erscheint das Dialogfeld "Suchen" (*Abbildung unten*).



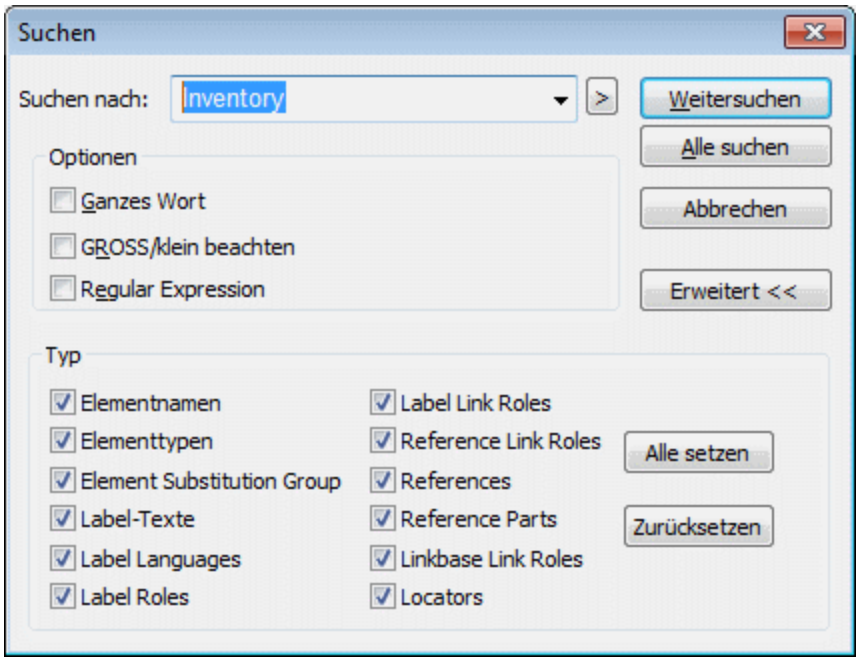
Dieses wird auf folgende Weise verwendet:

- Geben Sie im Dialogfeld "Suchen" im Textfeld "Suchen nach" den Suchbegriff ein[484] (*Abbildung oben*) und aktivieren Sie die gewünschten Optionen.
- Geben Sie im Bereich "Typen" an, nach welchen XBRL-Komponententypen gesucht werden soll[485].
- Führen Sie den Befehl aus[492], indem Sie auf **Weitersuchen** oder **Alle suchen** klicken
- Sehen Sie sich die Suchergebnisse im Fenster "In Schemas suchen" an[494] und navigieren Sie zur Komponente

17.7.1 Suchbegriff

Sie können bei einem Suchbegriff festlegen, dass die Groß- und Kleinschreibung beachtet werden soll, oder dass nur nach ganzen Wörtern gesucht werden soll. Aktivieren Sie dazu die entsprechenden Optionen im Bereich "Optionen" (*siehe Abbildung unten*). Wenn Sie mit Hilfe einer Regular Expression suchen möchten, aktivieren Sie im Bereich "Optionen" die Option *Regular Expression,* bevor Sie auf die Schaltfläche

Weitersuchen oder **Alle suchen** klicken. Nähere Informationen zur Verwendung von Regular Expressions finden Sie weiter unten.

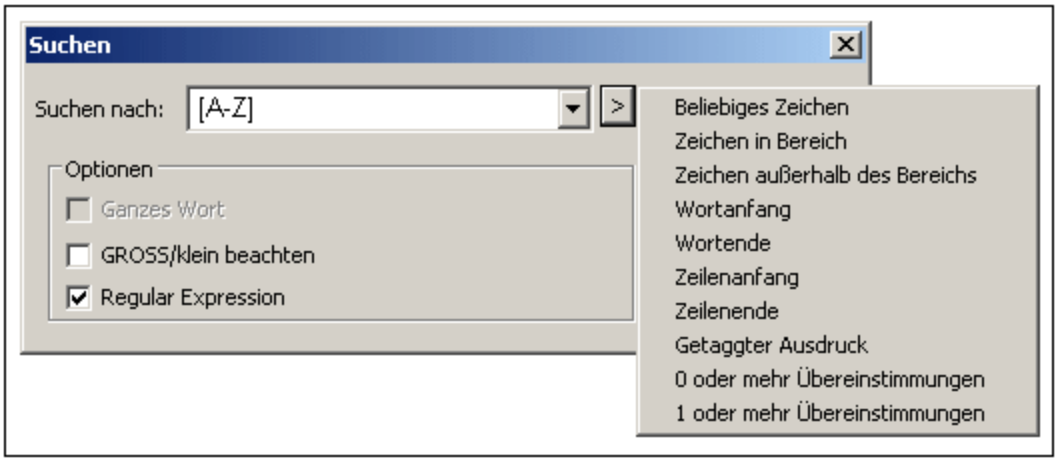


Anmerkung: Als ganzes Wort gilt jedes Wort, das zwischen Zeichen steht, die weder alphanumerisch sind oder ein Unterstrich. Bei Eingabe des Suchbegriffs `asset` wird der Text `xbrl:asset` gefunden, da der Doppelpunkt (:) als Word-Trennzeichen betrachtet wird.

Definieren Sie im Bereich "Typ" die zu durchsuchenden Komponenten.

Regular Expressions

Sie können zur weiteren Eingrenzung Ihrer Suchkriterien Regular Expressions verwenden. Als Hilfe bei der Erstellung steht eine Popup-Liste zur Verfügung. Um die Liste aufzurufen, klicken Sie auf die Schaltfläche **>** rechts vom Eingabefeld für den Suchbegriff.



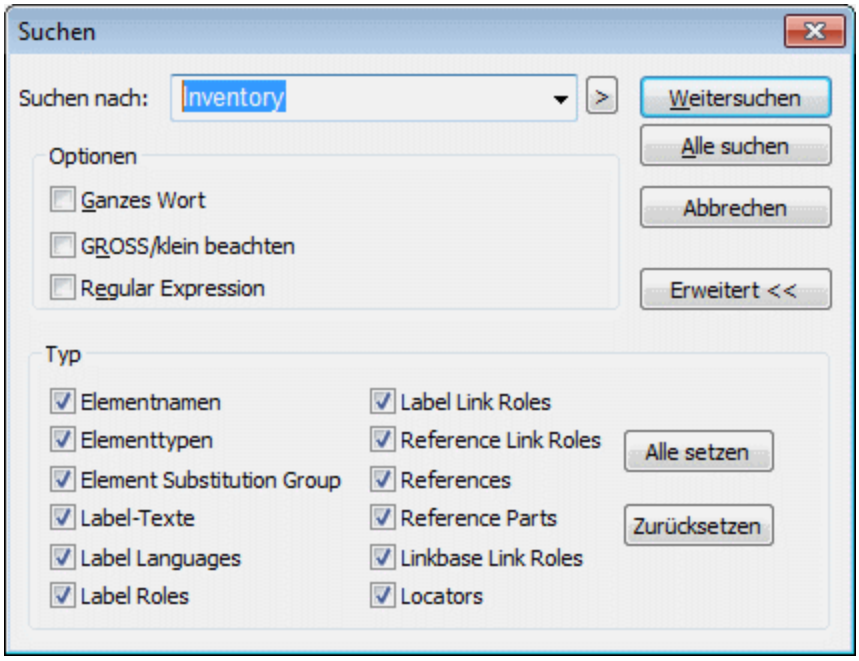
© 2016-2022 Altova GmbH

Wenn Sie auf die Beschreibung des gewünschten Suchausdrucks klicken, wird die entsprechende Expression-Syntax im Eingabefeld übernommen. Im Folgenden sehen Sie eine Liste von Syntax-Zeichen, die für Regular Expressions verwendet werden.

`.`	Steht für jedes beliebige Zeichen. Dies ist ein Platzhalter für ein einzelnes Zeichen.
`\ (`	Markiert den Beginn einer Region, die gesucht werden soll.
`\ )`	Markiert das Ende einer zu suchenden Region.
`\<`	Steht für den Beginn eines Worts.
`\>`	Steht für das Ende eines Worts.
`\x`	Damit können Sie ein Zeichen x verwenden, das sonst eine spezielle Bedeutung hätte. So würde z.B. \[als [und nicht als Beginn einer Zeichenmenge interpretiert werden.
`[...]`	Kennzeichnet eine Zeichenmenge, [abc] z.B. steht für jedes der Zeichen a, b oder c. Sie können auch Bereiche angeben, z.B. [a-z] für alle klein geschriebenen Zeichen.
`[^...]`	Die invertierte Zeichenmenge. [^A-Za-z] z.B. steht für jedes Zeichen mit Ausnahme alphabetischer Zeichen.
`^`	Steht für den Zeilenanfang (es sei denn dieses Zeichen wird innerhalb einer Menge verwendet, siehe oben).
`$`	Steht für das Zeilenende. Beispiel: A+$ findet ein oder mehrere A's am Ende der Zeile.
`*`	Steht für 0 oder öfter. Mit Sa*m werden z.B. Sm, Sam, Saam, Saaam usw. gefunden.
`+`	Steht für 1 oder öfter. Mit Sa+m werden z.B. Sam, Saam, Saaam usw. gefunden.

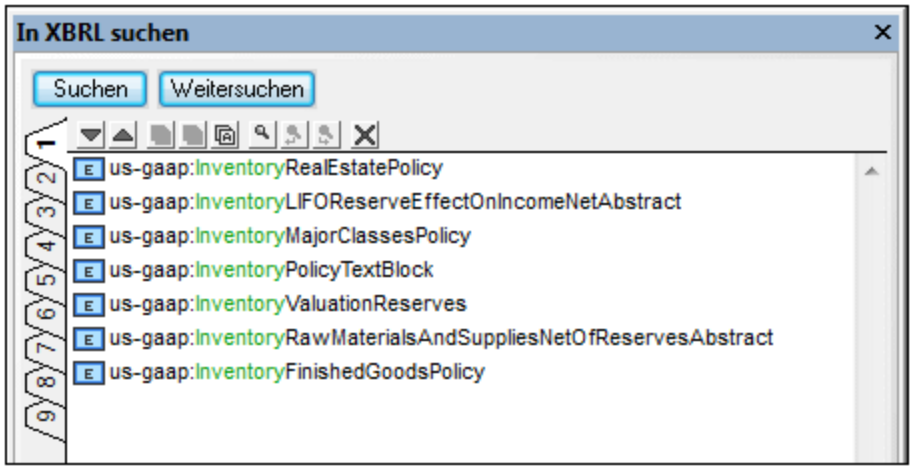
17.7.2 Befehlsausführung

Nachdem der Suchbegriff eingegeben wurde und die Suchoptionen und Filter für den Komponententyp festgelegt wurden, stehen zwei Optionen für die Befehlsausführung zur Verfügung: **Weitersuchen** und **Alle suchen**. Diese Befehle stehen in Form von Schaltflächen im Suchdialogfeld zur Verfügung (*siehe Abbildung unten*).



Weitersuchen

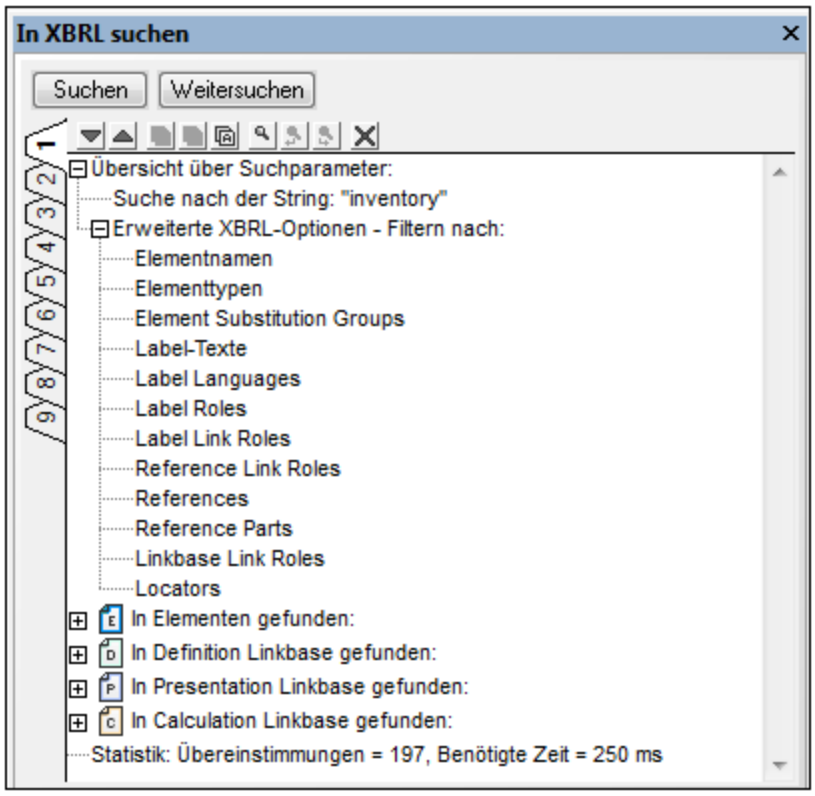
Bei Auswahl des Befehls **Weitersuchen** wird im Fenster "In XBRL suchen" (*Abbildung unten*) die nächste Instanz des Suchbegriffs angezeigt. Die Suche nach der nächsten Instanz beginnt bei der nächsten Zelle nach der aktuellen Cursorposition im aktiven Dokument. Sie können mit **Weitersuchen** fortfahren, bis alle Instanzen im Dokument angezeigt werden.



Alle suchen

Bei Auswahl des Befehls **Alle suchen** werden alle Instanzen des Suchbegriffs zusammen mit einer Zusammenfassung der Suche im Fenster "In XBRL suchen" (*Abbildung unten*) angezeigt.

© 2016-2022 Altova GmbH

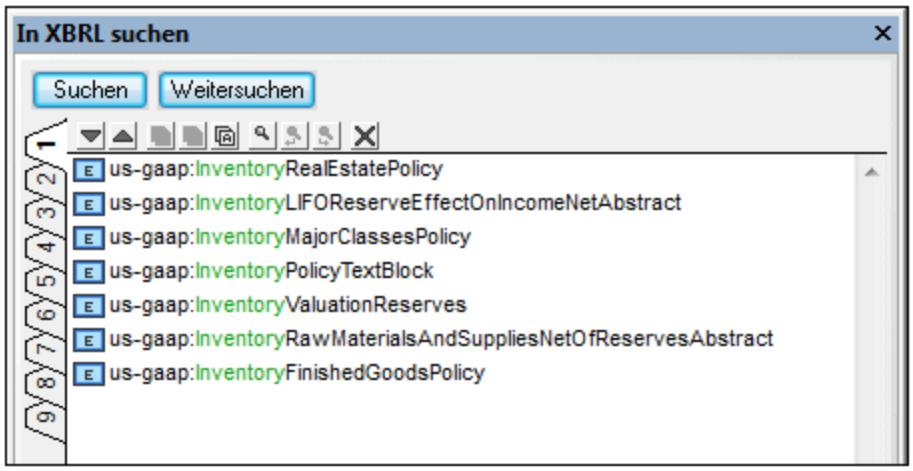


Das Ergebnis enthält: (i) eine Zusammenfassung der XBRL-Komponententypen, die durchsucht wurden; (ii) eine Liste der gefundenen Instanzen des Suchbegriffs, geordnet nach Linkbase; und (iii) eine Statistik zur Suche mit der Anzahl der gefundenen Treffer und der Zeit, die für die Suche benötigt wurde. Jede Linkbase-Gruppe kann erweitert oder reduziert werden, um die Treffer in dieser Gruppe anzuzeigen. Wenn Sie auf einen Treffer klicken, wird das entsprechende Element im XBRL-Dokument im Hauptfenster markiert.

Informationen zu den Funktionen des Fensters "In XBRL suchen" finden Sie im Abschnitt Ergebnisse und Informationen 951.

17.7.3 Ergebnisse und Informationen

Jedes Mal, wenn der Befehl **Suchen** oder **Weitersuchen** ausgeführt wird, werden die Ergebnisse im Fenster "In XBRL suchen" (*Abbildung unten*) angezeigt. Der gesuchte Begriff wird grün angezeigt; (in der Abbildung unten sehen Sie, dass `Inventory` der Suchbegriff war).



Funktionen des Fensters "In XBRL suchen"

Die Ergebnisse werden auf neun separaten Registern (mit den Nummern 1 bis 9) angezeigt. Sie können auf diese Weise die Ergebnisse einer Suche auf einem Register behalten, eine neue Suche durchführen und die Ergebnisse vergleichen. Wenn Sie im Fenster "In XBRL suchen" auf ein Ergebnis klicken, erscheint die entsprechende Komponente im Hauptfenster der XBRL-Ansicht und wird markiert angezeigt. Auf diese Art können Sie mit Hilfe des Fensters "In XBRL suchen" die gewünschte Komponente schnell finden und zu ihr navigieren.

In der Symbolleiste "In XBRL suchen" stehen die folgenden Befehle zur Verfügung:

- Mit den Schaltflächen **Nächstes** und **Vorheriges** wird das nächste bzw. vorherige Suchergebnis nach bzw. vor dem gerade ausgewählten Ergebnis ausgewählt.
- Mit den Befehlen zum **Kopieren von Meldungen** kann bzw. können die ausgewählte Meldung, bzw. die ausgewählte Meldung und ihre untergeordneten Einträge bzw. alle Meldungen in die Zwischenablage kopiert werden.
- Mit dem Befehl **Suchen** werden Textstrings im Fenster "In XBRL suchen" gesucht.
- Mit dem Befehl **Löschen** werden alle Meldungen auf dem gerade aktiven Register gelöscht.

© 2016-2022 Altova GmbH

17.8 OIM

Die Open Information Model (OIM) 1.0-Spezifikation enthält ein syntaxunabhängiges Modell für XBRL-Daten, wodurch eine zuverlässige Transformation von XBRL-Daten in andere Darstellungen (OIM-XML, OIM-JSON und OIM-CSV) ermöglicht wird. XMLSpy bietet im Zusammenhang mit OIM die folgenden Funktionalitäten:

- Konvertierung[1470] von XBRL-Datendokumenten ins OIM xBRL-JSON- und OIM xBRL-CSV-Format (über das Menü Konvertieren[1470]).
- Konvertierung[1470] aus einem beliebigen der folgenden drei Formate in ein anderes der beiden Formate: OIM xBRL-XML, OIM xBRL-JSON und OIM xBRL-CSV (über das Menü Konvertieren[1470]).
- Validierung[954] eines als XBRL-Dokument erkannten Dokuments in einem der OIM-Formate und entsprechende Validierung.

17.9 Validieren von XBRL-Instanzen und -Taxonomien

Um eine XBRL-Instanz oder -Taxonomie zu validieren, machen Sie das XBRL-Dokument zum aktiven Dokument und wählen Sie eine der unten aufgelisteten Validierungsmethoden aus:

- **XML | XML validieren (F8)**[1353]. Die Validierung erfolgt mit dem integrierten Prozessor von XMLSpy.
- **XML | XML am Server validieren (Strg+F8)**[1357]. Die Validierung wird auf einem entfernten RaptorXML+XBRL Server durchgeführt (Sie können diesen über den Befehl **Extras | Raptor-Server verwalten**[1579] konfigurieren**).**

Die entsprechenden Validierungsoptionen können auf dem Register "XBRL: Validierung" des Dialogfelds "Optionen"[1639] (**Extras | Optionen**) eingestellt werden.

© 2016-2022 Altova GmbH

18 Office Open XML, ZIP, EPUB

Office Open XML (OOXML), ZIP- und EPUB-Dateien ähneln einander insofern, als es sich bei allen diesen Dateien um Pakete handelt, die weitere Dateien enthalten. Die Archivansicht-Ansicht von XMLSpy enthält eine Benutzeroberfläche, über die Sie die interne Struktur dieser Pakete anzeigen, diese Strukturen ändern und die Dateien im Paket zur Bearbeitung in XMLSpy aufrufen können. Bei EPUB-Dateien kann das EPUB-Buch über die Archivansicht-Ansicht auch direkt in der Browser-Ansicht von XMLSpy angezeigt werden.

Office Open XML (OOXML)

OOXML ist ein Dateiformat zur Beschreibung von Dokumenten, Tabellenkalkulationen und Präsentationen. Ursprünglich wurde es von Microsoft für seine Office Suite entwickelt, mittlerweile ist es jedoch eine offene ECMA-Spezifikation.

Struktur einer OOXML-Datei

Jedes OOXML-Dokument ist ein Paket bestehend aus mehreren Dateien, das der Open Packaging Convention entspricht. Ein Paket besteht aus XML- und anderen Datendateien (wie z.B. Bilddateien) sowie einer Beziehungsdatei, in der die Beziehungen zwischen den verschiedenen Dateien im Paket definiert sind.

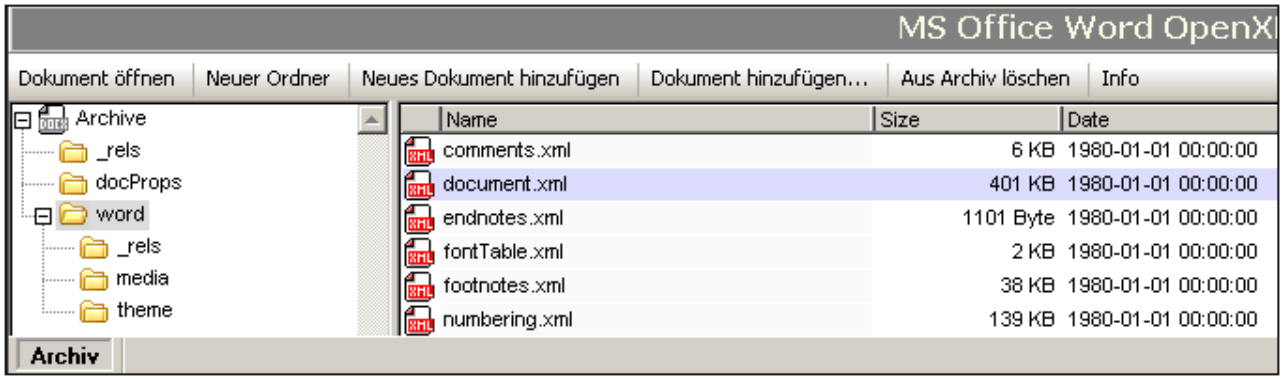
Die interne Struktur und die internen Ordner- und Dateinamen einer OOXML-Datei variieren je nach Dateiart. Es gibt jedoch eine gemeinsam Struktur: eine XML-Datei namens `[Content_Types].xml` an der Root der Verzeichnisstruktur und drei Verzeichnisse: `_rels`, `docProps` und ein Verzeichnis für die Dokumentart (im Fall von `.docx`-Dokumenten hieße dieser Ordner z.B. `word`; in `.xlsx`-Dokumenten `xl` und in `.pptx`-Dokumenten `ppt`).

```
OOXML File
|-- File:            [Content_Types].xml
|-- Folder:   _rels
|-- Folder:   docProps
|-- Folder:   word/xl/ppt
```

- Der Ordner `_rels` enthält eine `rels.xml` Datei, in der die Beziehungen zwischen den verschiedenen Dateien im Paket definiert sind.
- Der Ordner `docProps` enthält `app.xml` und `core.xml`, in denen die Schlüsseleigenschaften der Dokumente beschrieben sind.
- Die Ordner `word`, `xl` und `ppt` enthalten XML-Dateien, die den Inhalt des Dokuments enthalten. So enthält z.B. die Datei `document.xml` im Ordner `word` den eigentlichen Inhalt des Dokuments.

OOXML in der Archivansicht von XMLSpy

In der Archivansicht von XMLSpy (*Abbildung unten*) können Sie den Inhalt einer OOXML-Datei anzeigen und bearbeiten.



In der Ordneransicht auf der linken Seite sehen Sie die Ordner in diesem Paket, während Sie im Hauptfenster die Dateien sehen, die sich im Ordner befinden, der in der Ordneransicht ausgewählt ist. In der Archivansicht können Dateien und Ordner zum Archiv hinzugefügt und daraus gelöscht werden. Wenn Sie in der Archivansicht auf eine Datei doppelklicken, wird diese sofort zur Bearbeitung in XMLSpy geöffnet.

Intelligente Bearbeitung interner OOXML-Dateien

Die XML-Dokumente in OOXML-Paketen basieren auf Standardschemas. XMLSpy bietet in Form von Eingabehilfen, Autokomplettierung und Validierung intelligente Bearbeitungsfunktionen für OOXML-Dokumente.

ZIP-Dateien

ZIP-Dateien enthalten mehrere Dateien in komprimierter Form, ohne dass dabei Daten verloren gehen. Dabei kann es sich um verschiedene Dateiarten handeln. In der Archivansicht-Ansicht von XMLSpy können ZIP-Dateien erstellt, die innere Struktur geändert und Dateien im Archiv bearbeitet werden. Diese Vorgänge werden im Unterabschnitt ZIP-Dateien[961] dieses Abschnitts beschrieben.

EPUB-Dateien

Eine EPUB-Datei ist eine gezippte Gruppe von Dateien, die für die Verteilung digitaler Publikationen (EPUB-Bücher) verwendet werden. Sie können EPUB-Dateien in der Archivansicht[284]-Ansicht öffnen, diese erstellen und bearbeiten, eine Vorschau des digitalen EPUB-Buchs anzeigen, die Komponentendateien des EPUB-Archivs direkt in XMLSpy bearbeiten, die EPUB-Datei validieren und die Komponentendateien wieder im EPUB-Archiv speichern. Nähere Informationen dazu finden Sie im Abschnitt EPUB-Dateien[963].

© 2016-2022 Altova GmbH

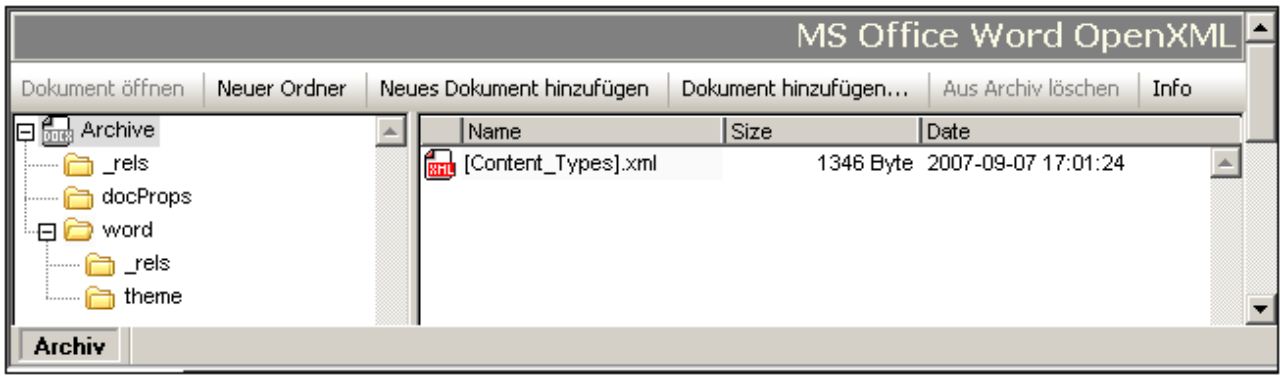
18.1 Arbeiten mit OOXML-Dateien

In diesem Abschnitt wird beschrieben, wie Sie in der Archivansicht-Ansicht mit OOXML-Dokumenten arbeiten. Es werden die folgenden Operationen beschrieben.

- Erstellen, Öffnen und Speichern von OOXML-Dateien 957
- Bearbeiten der Struktur einer OOXML-Datei 957
- Öffnen, Bearbeiten und Speichern von internen OOXML-Dokumenten 958
- Intelligente Bearbeitung von internen OOXML-Dokumenten 958
- Ansprechen von Dokumenten in OOXML-Dateien 958
- Vergleichen von OOXML-Archiven 957

Erstellen, Öffnen und Speichern von OOXML-Dateien

OOXML-Dateien werden über das Dialogfeld "Neues Dokument anlegen" (Befehl **Datei | Neu**) erstellt, in dem Sie den gewünschten Dateityp auswählen (`.docx`, `.pptx`, oder `.xlsx`). Sie werden aufgefordert, einen Dateinamen und -pfad für die Datei anzugeben. Die neue Datei wird im angegebenen Ordner angelegt und anschließend in der Archivansicht-Ansicht (*Abbildung unten*) geöffnet. Beachten Sie, dass die grundlegende interne Struktur des OOXML-Dokuments erstellt wurde.



Eine vorhandene OOXML-Datei wird in der Archivansicht über das XMLSpy-Dialogfeld "Öffnen" (**Datei | Öffnen**) geöffnet. OOXML-Dateien werden mit dem Befehl **Datei | Speichern (Strg+S)** gespeichert. Damit werden die Struktur und die Beziehungen der OOXML-Datei gespeichert.

Bearbeiten der Struktur einer OOXML-Datei

Der Inhalt einer OOXML-Datei kann durch Hinzufügen und Löschen von Ordnern und Dokumenten über die Funktionalitäten der Archivansicht-Ansicht 284 geändert werden. Anschließend muss die OOXML-Datei gespeichert werden (**Datei | Speichern**) damit die Änderungen wirksam werden. Beachten Sie bitte die folgenden Punkte:

- Wenn ein neuer Ordner oder ein neues Dokument über die Befehlsschaltflächen in der Archivansicht-Ansicht 284 hinzugefügt wird, muss er/es sofort bei der Erstellung benannt werden. Sie können einen Ordner oder ein Dokument in der Archivansicht-Ansicht nicht umbenennen.
- Nachdem ein neues Dokument zu einem Archivordner hinzugefügt wurde, wird es durch Speichern in seinem eigenen Fenster oder durch Speichern der OOXML-Datei gespeichert.

Öffnen, Bearbeiten, Speichern interner OOXML-Dokumente

Ein internes OOXML-Dokument, d.h. ein Dokument in einem OOXML-Dateipaket, wird durch Doppelklick in der Archivansicht-Ansicht oder durch Auswahl des Dokuments im Hauptfenster und Klicken auf die Befehlsschaltfläche Dokument öffnen[284] geöffnet. Das Dokument wird in einem separaten XMLSpy-Fenster geöffnet. Nachdem Sie es bearbeitet haben, speichern Sie es einfach wieder im OOXML-Archiv; es ist nicht notwendig, die OOXML-Datei selbst zu speichern.

Intelligente Bearbeitung von internen OOXML-Dokumenten

XMLSpy bietet intelligente Bearbeitungsfunktionen für interne Office Open XML-Dokumenten, d.h. für Dokumente in einem OOXML-Dateipaket. Dazu gehören Eingabehilfen, Autokomplettierung und Validierung.

Ansprechen von Dokumenten in OOXML-Dateien

Dokumente in OOXML-Dateien können mittels normaler Dateipfade plus dem Pipe-Symbol angesprochen werden. So wird z.B. mit dem Dateipfad:

```
C:\Dokumente und Einstellungen\<Benutzer>\Eigene
Dokumente\Altova\XMLSpy2023\Examples\Office2007\ExcelDemo.xlsx|zip\xl\tables\table1.xml
```

die Datei `table1.xml` gefunden, die sich im Ordner `xl\tables` der OOXML-Datei `ExcelDemo.xlsx` im Ordner `Examples\Office2007` des XMLSpy-Beispielordners befindet.

Vergleichen von OOXML-Archiven

Wenn eine OOXML-Datei in der Archivansicht-Ansicht geöffnet ist, können Sie sie mit dem Befehl **Extras | Verzeichnisse vergleichen**[1572] mit einem anderen Archiv vergleichen.

© 2016-2022 Altova GmbH

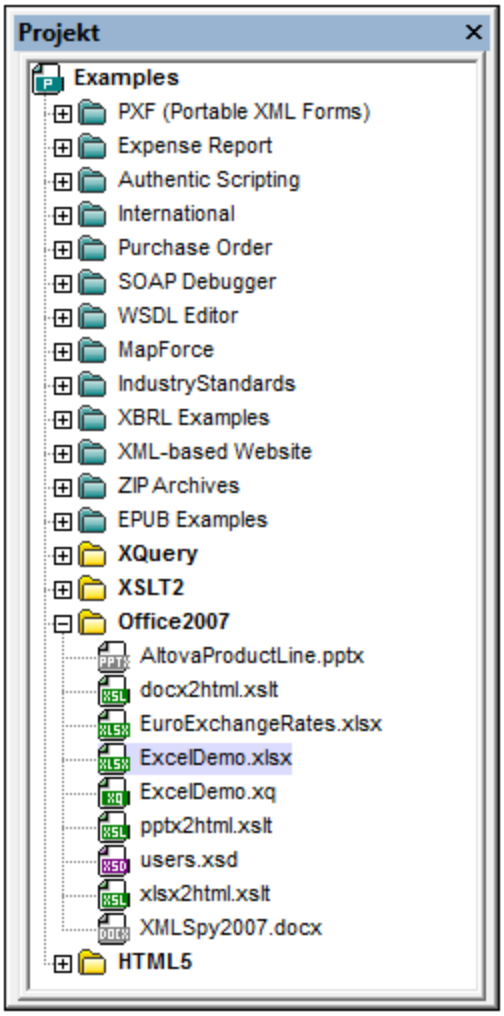
18.2 OOXML-Beispieldateien

Im Ordner `Examples\Office2007` Ihres XMLSpy Applikationsordners befinden sich die folgenden Beispieldateien:

- OOXML-Dateien: (i) eine Word Open XML-Datei (`.docx`), und (ii) eine Excel Open XML-Datei (`.xlsx`) sowie (iii) eine PowerPoint Open XML-Datei (`.pptx`)
- XSLT-Dateien: (i) `docx2html.xslt` (zum Konvertieren der `.docx`-Beispieldatei in HTML), und (ii) `xslx2html.xslt` (zum Konvertieren der `.xslx` Beispieldatei in HTML) sowie (iii) `pptx2html.xslt` (zum Konvertieren der `pptx`-Beispieldatei in HTML)
- Eine XQuery-Datei: `ExcelDemo.xq` (zum Abrufen der Daten aus der `.xslx` Datei)

Die XSLT- und die XQuery-Datei sind dazu gedacht, um zu zeigen, wie man mit Hilfe von XSLT und XQ Daten in ZIP-Archiven aufrufen und transformieren kann. Zum Ausführen dieser Dokumente können Sie eine der folgenden Optionen verwenden:

- Öffnen Sie die OOXML-Datei in der Archivansicht-Ansicht. Wählen Sie in der Ordneransicht `Archive` aus und klicken Sie auf den Menübefehl **XSL/XQuery | XSL Transformation** (um eine XSLT-Transformation durchzuführen) oder **XSL/XQuery | XQuery-Ausführung** (um eine XQuery auszuführen). Suchen Sie die XSLT- oder die XQuery-Datei und klicken Sie auf **OK**.
- Klicken Sie im Projektfenster von XMLSpy im Ordner `Office2007` des Projekts `Examples` mit der rechten Maustaste auf die Datei `.xlsx`, `.pptx` oder `.docx` (*Abbildung unten*) und wählen Sie den Transformationsbefehl. Suchen Sie die Transformationsdatei und klicken Sie auf **OK**.



- Öffnen Sie die XSLT oder XQuery-Datei in XMLSpy und klicken Sie auf den Menübefehl **XSL/XQuery | XSL-Transformation** bzw. **XSL/XQuery | XQuery-Ausführung**. Wenn Sie aufgefordert werden, die gewünschte XML-Datei anzugeben, suchen Sie nach der `.docx-, .xlsx-` oder `.pptx`-Datei (je nachdem, ob das XSLT/XQ-Dokument für MS Word, MS Excel oder MS PowerPoint gedacht ist).

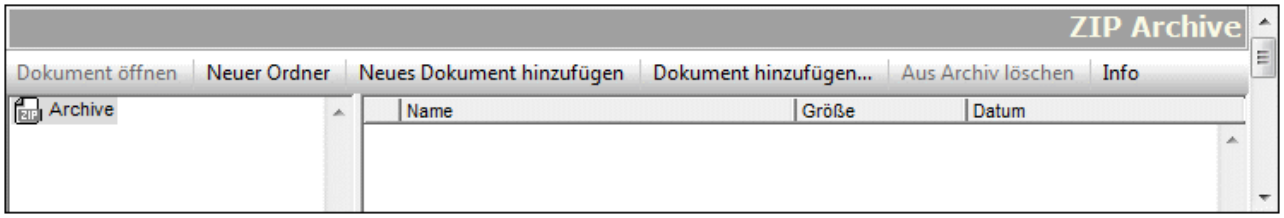
© 2016-2022 Altova GmbH

18.3 ZIP-Dateien

Sie können in der Archivansicht-Ansicht[284] WinZip-Dateien erstellen, die interne Struktur von ZIP-Dateien (WinZip, WinRAR, usw.) ändern und Dateien in ZIP-Paketen direkt in XMLSpy bearbeiten und wieder im ZIP-Archiv speichern.

Erstellen und Speichern von WinZip-Dateien

Eine WinZip-Datei wird über das Dialogfeld "Neues Dokument anlegen" (Befehl **Datei | Neu**) erstellt, in dem Sie den gewünschten Dateityp `.zip` auswählen. Ein leeres WinZip-Archiv wird in einem neuen Fenster in XMLSpy erstellt (*Abbildung unten*). Sie müssen die Datei nun mit dem Befehl **Datei | Speichern (Strg+S)** im gewünschten Ordner speichern. Fügen Sie, wie unten beschrieben Ordner und Dateien hinzu und speichern Sie die ZIP-Datei anschließend, um Ihre Änderungen zu speichern.



Eine vorhandene ZIP-Datei wird in der Archivansicht über das XMLSpy-Dialogfeld "Öffnen" (**Datei | Öffnen**) geöffnet.

Anmerkung: Das Erstellen einer neuen ZIP-Datei unterscheidet sich insofern vom Erstellen einer neuen OOXML-Datei, als Sie nicht aufgefordert werden, einen Pfad zum Speichern der Datei anzugeben, bevor das Archiv in der Archivansicht-Ansicht geöffnet wird. Um die ZIP-Datei vom leeren Archiv aus, das in der Archivansicht-Ansicht geöffnet ist, zu speichern, müssen Sie explizit den Befehl **Datei | Speichern (Strg+S)** verwenden.

Hinzufügen von Ordnern und Dateien und Ändern der Archivstruktur

Sie können Ordner (klicken Sie auf die Schaltfläche **Neuer Ordner**), bestehende Dateien (**Dokument hinzufügen**) und neue Dateien (**Neues Dokument hinzufügen**) zum ausgewählten Archivordner hinzufügen. Beachten Sie: Wenn Sie einen neuen Ordner oder ein neues Dokument hinzufügen, müssen Sie sofort einen Namen für die Ordner bzw. die Datei eingeben; Ordner oder Dokumente können in der Archivansicht-Ansicht zu einem späteren Zeitpunkt nicht mehr umbenannt werden.

Ansprechen von Dokumenten in ZIP-Dateien

Dokumente in ZIP-Dateien können mittels normaler Dateipfade plus dem Pipe-Symbol angesprochen werden. So wird z.B. mit dem Dateipfad:

```
C:\Dokumente und Einstellungen\<Benutzer>\Eigene
Dokumente\Altova\XMLSpy2023\Examples\Test.zip|zip\TestFolder\MyFile.xml
```

die Datei `MyFile.xml` gefunden, die sich im Ordner `TestFolder` der ZIP-Datei `Test.zip` im Ordner `Examples` des XMLSpy-Beispielordners befindet.

Vergleichen von ZIP-Archiven

Wenn eine ZIP-Datei in der Archivansicht-Ansicht geöffnet ist, können Sie sie mit dem Befehl **Extras | Verzeichnisse vergleichen** 1572 mit einem anderen Archiv vergleichen.

© 2016-2022 Altova GmbH

18.4 EPUB-Dateien

Bei einer EPUB-Datei handelt es sich um eine gezippte Gruppe von Dateien, die dem EPUB-Standard des International Digital Publishing Forum (IDPF) entsprechen. Dieser Standard regelt die Verbreitung und den Austausch digitaler Web-Publikationen.Sie können EPUB-Dateien in der Archivansicht[284]-Ansicht öffnen, das digitale EPUB-Buch auf einem Vorschauregister anzeigen, Komponentendateien des EPUB-Archivs direkt in XMLSpy bearbeiten, die EPUB-Datei validieren und die Komponentendateien wieder im EPUB-Archiv speichern.

Anmerkung: (i) XMLSpy unterstützt EPUB 2.0.1. (ii) Im Projekt `Examples` und im Ordner `(My) Documents/Altova/XMLSpy2023/Examples` finden Sie eine EPUB-Beispieldatei.

Terminologie

In der nachstehenden Beschreibung werden die folgenden Begriffe verwendet:

- ***EPUB-Datei:*** Damit wird die EPUB-Datei mit der Dateierweiterung `.epub` bezeichnet. Dies ist die ZIP-Datei, die das gesamte Archiv enthält. Dies ist die Datei, die in der Archivansicht-Ansicht geöffnet wird.
- Alle im EPUB-Archiv enthaltenen Dateien sind ***Archivdateien.***
- Mit ***EPUB-Buch*** wird die digitale Publikation bezeichnet, die aus der gezippten EPUB-Datei generiert wird.

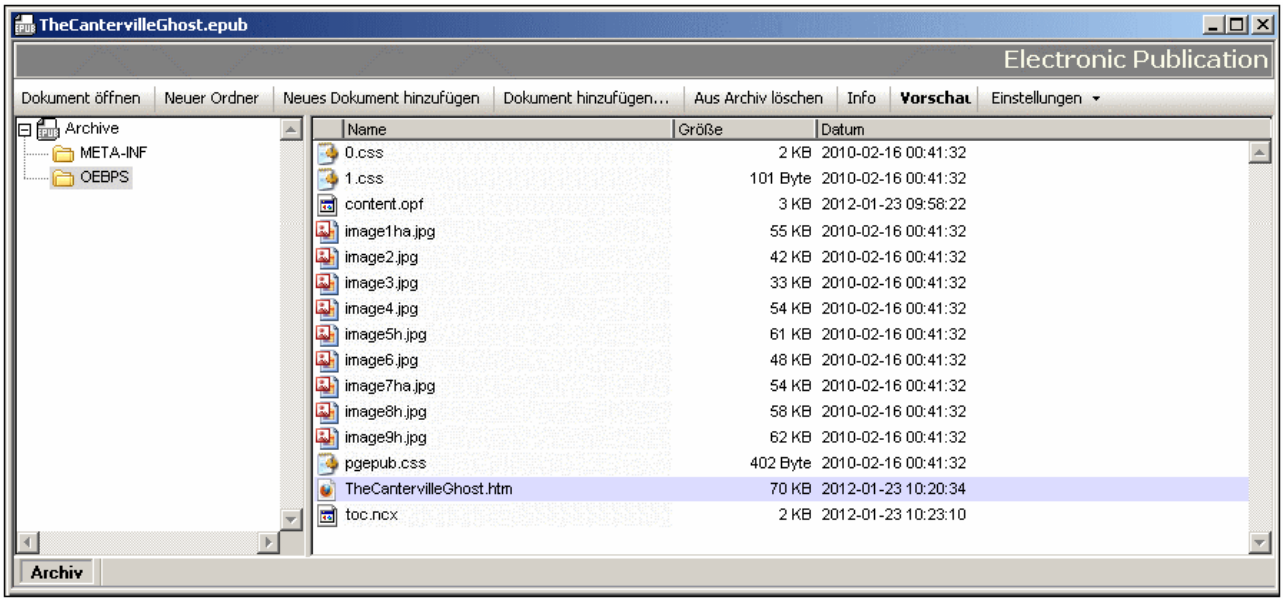
Dieser Abschnitt

Die Beschreibung der EPUB-Funktionen in XMLSpy ist in die folgenden Unterabschnitte gegliedert:

- Öffnen von EPUB-Dateien in der Archivansicht-Ansicht[963]
- Erstellen einer neuen EPUB-Datei[964]
- Anzeigen einer Vorschau eines EPUB-Buchs[964]
- Ändern des Inhalts und der Struktur eines EPUB-Archivs[965]
- Info und Einstellungen[965]
- Bearbeiten von Archivdateien direkt in XMLSpy[966]
- Eingabehilfen für Archivdateien[966]
- Validieren der EPUB-Datei[966]

Öffnen von EPUB-Dateien in der Archivansicht-Ansicht

Wählen Sie den Menübefehl **Datei | Öffnen**, navigieren Sie zur EPUB-Datei und klicken Sie auf **Öffnen**. Daraufhin wird die EPUB-Datei in der Archivansicht-Ansicht geöffnet (*Abbildung unten*). Alternativ dazu können Sie im Windows Explorer mit der rechten Maustaste auf die EPUB-Datei klicken und im Kontextmenü den Befehl zum Öffnen der Datei mit XMLSpy auswählen. Wenn Sie XMLSpy als Standard-Editor für EPUB-Dateien definiert haben[1606], wird die EPUB-Datei bei Doppelklick darauf automatisch in der <%ARCHIVE-VIEW %>-Ansicht geöffnet.



In der Ordneransicht auf der linken Seite werden die Ordner im Archiv angezeigt, während im Hauptfenster die Dateien des in der Ordneransicht ausgewählten Ordners zu sehen sind. Ein EPUB-Archiv hat die folgende Struktur und die folgenden Schlüsselkomponenten.

```
Archiv
|-- Mimetype-Dateien
|
|-- META-INF-Ordner
|     |-- container.xml
|
|-- DOKUMENT-Ordner (In der Abbildung oben ist OEBPS der Dokumentordner.)
      |-- Enthält HTML-, CSS-, Bilddateien sowie OPF- und NCX-Dateien
```

Erstellen einer neuen EPUB-Datei

Um eine neue EPUB-Datei zu erstellen, wählen Sie den Menübefehl **Datei | Neu**. Wählen Sie nun im Dialogfeld "Neues Dokument erstellen" den Dateityp `.epub` aus. Wählen Sie im Dialogfeld "Speichern unter", das daraufhin angezeigt wird, einen Namen für Ihr EPUB-Dokument aus und klicken Sie auf **Speichern.** Daraufhin wird ein EPUB-Skelettarchiv in einem neuen Fenster der Archivansicht-Ansicht erstellt. Dieses Archiv enthält alle Ordner und Dateien eines gültigen EPUB-Archivs (*siehe Archivstruktur oben*). Fügen Sie die gewünschten Ordner und Dateien wie oben beschrieben zum Archiv hinzu und speichern Sie die EPUB-Datei anschließend. Um eine Archivdatei direkt in XMLSpy zu bearbeiten, doppelklicken Sie in der Archivansicht-Ansicht auf die Datei. Daraufhin wird die Datei in einem neuen XMLSpy -Fenster geöffnet. Bearbeiten Sie die Datei und speichern Sie sie mit dem Befehl **Datei | Speichern (Strg+S)**.

Anzeigen einer Vorschau eines EPUB-Buchs

Um eine Vorschau eines EPUB-Buchs anzuzeigen, machen Sie die EPUB-Datei in der Archivansicht-Ansicht zur aktiven und klicken Sie dann in der Symbolleiste der Archivansicht-Ansicht auf die Schaltfläche **Vorschau**. Daraufhin wird das EPUB-Buch in einem (Internet Explorer) Browser-Fenster in XMLSpy geöffnet. Wenn eine der für die Vorschau verwendeten Dateien - eine Inhaltsdatei oder eine für die Struktur erforderliche Datei - geändert aber nicht gespeichert wurde, werden Sie aufgefordert die Datei zu speichern. Wenn Sie die

 © 2016-2022 Altova GmbH

Änderungen nicht speichern, werden die früheren Daten für die Vorschau verwendet, d.h. die Vorschau ist möglicherweise nicht aktuell. Durch Aktivieren der entsprechenden Einstellung (über die Schaltfläche **Einstellungen** in der Symbolleiste der Archivansicht-Ansicht) können Sie festlegen, dass alle geänderten Dateien vor der Vorschau automatisch gespeichert werden.

Beachten Sie Folgendes:

- Wenn Sie in der Archivansicht-Ansicht auf die Schaltfläche **Vorschau** klicken, während ein Vorschaufenster des EPUB-Buchs noch geöffnet ist, so wird das EPUB-Buch im offenen Vorschaufenster neu geladen.
- Wenn sie das Vorschaufenster selbst in Internet Explorer mit **Aktualisieren (F5)** aktualisieren, wird das Vorschaufenster nicht aktualisiert. Sie müssen zuerst auf die Schaltfläche **Vorschau** der entsprechenden EPUB-Datei klicken, um das Vorschaufenster zu aktualisieren (*siehe vorheriger Punkt*).
- Um die Vorschau zu schließen, schließen Sie das Vorschaufenster.

Anmerkung: Internet Explorer unterstützt nicht den gesamten EPUB Markup-Code, daher wird die Vorschau nicht immer korrekt angezeigt. Wenn außerdem das digitale Dokument für die Publikation in XML und nicht in HTML verfasst ist, kann es vorkommen, dass die Vorschau nicht funktioniert. Neuere Versionen von Internet Explorer bieten eine bessere Unterstützung des EPUB-Markup-Codes. Wenn Sie daher Probleme damit haben, installieren Sie nach Möglichkeit die neueste Version von Internet Explorer.

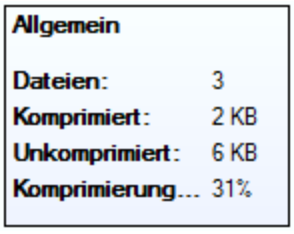
Ändern des Inhalts und der Struktur eines EPUB-Archivs

Sie können Ordner (über die Schaltfläche **Neuer Ordner**), neue Dateien (**Neues Dokument hinzufügen**) und vorhandene Dateien (**Dokument hinzufügen**) zum ausgewählten Ordner hinzufügen. Beachten Sie Folgendes: Wenn Sie einen neuen Ordner oder ein neues Dokument hinzufügen, müssen Sie sofort einen Namen für den Ordner bzw. die Datei eingeben; Ordner können in der Archivansicht-Ansicht nicht mehr umbenannt werden. Durch Auswahl einer Datei bzw. eines Ordners und Klicken auf die Schaltfläche **Aus Archiv löschen** können Sie die Datei bzw. den Ordner löschen.

Nachdem Sie das Archiv bearbeitet haben, müssen Sie die EPUB-Datei speichern (**Datei | Speichern**), damit die Änderungen gespeichert werden.

Info und Einstellungen

Wenn Sie auf die Schaltfläche **Info** klicken, wird am unteren Rand der Archivansicht-Ansicht eine Zusammenfassung der wichtigsten Archivinformationen angezeigt (*Abbildung unten*). Wenn Sie nochmals auf die Schaltfläche **Info** klicken, wird die Zusammenfassung wieder ausgeblendet. Die Zusammenfassung enthält die Anzahl der Dateien im Archiv (einschließlich Mimetype-Datei und `container.xml`), die Größe der komprimierten EPUB-Datei und die Gesamtgröße der entpackten Dateien.



Mit der Schaltfläche **Einstellungen** wird eine Auswahlliste mit zwei Optionen zur automatischen Dateispeicherung, die aktiviert und deaktiviert werden können, angezeigt: die EPUB-Datei (i) vor der Validierung und (ii) vor Anzeige der Vorschau in XMLSpy (über die Schaltfläche **Vorschau**) automatisch zu speichern.

Bearbeiten von Archivdateien direkt in XMLSpy

Um eine Archivdatei direkt in XMLSpy zu bearbeiten, doppelklicken Sie in der Archivansicht-Ansicht auf die Datei. Alternativ dazu können Sie die Datei in der Archivansicht-Ansicht auswählen und in der Symbolleiste der Archivansicht-Ansicht auf die Schaltfläche "Dokument öffnen" klicken. Daraufhin wird die Datei in einem neuen XMLSpy-Fenster geöffnet. Bearbeiten Sie die Datei und speichern Sie sie mit dem Befehl **Datei | Speichern (Strg+S)**.

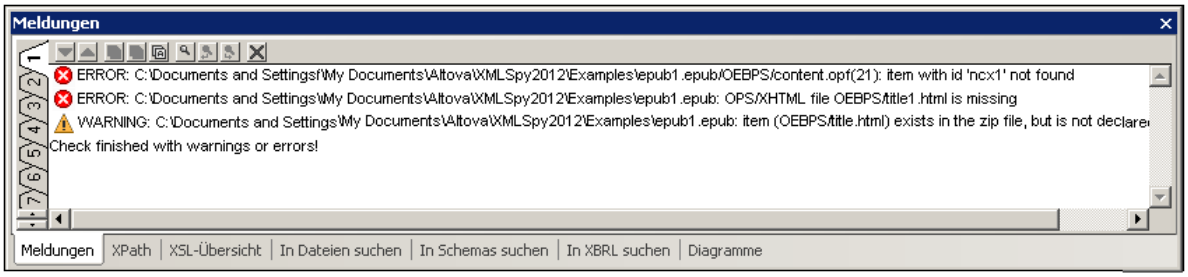
Eingabehilfe für Archivdateien

Die Eingabehilfen für standardbasierte Archivdateien stehen zur Verfügung, wenn diese Archivdateien in XMLSpy geöffnet werden. Diese Archivdateien sind:

- Die OPF-Datei: Normalerweise hat sie den Namen `content.opf` und enthält die Metadaten des EPUB-Buchs. Sie basiert auf der Open Packaging Format (OPF) Spezifikation.
- Die NCX-Datei (Navigationsdatei für XML): Normalerweise hat sie den Namen `toc.ncx` und enthält das Inhaltsverzeichnis der Publikation. Sie basiert auf dem NCX-Abschnitt der OPF-Spezifikation.
- Der Ordner mit dem Namen `META-INF` muss die Datei `container.xml` enthalten. Diese Datei verweist auf die Datei, in der der Inhalt des Buchs definiert ist (die OPF-Datei). In der Datei `container.xml` wird definiert, wie die Archivdateien gemäß den Regeln der Open Container Format (OCF) Spezifikation organisiert werden sollen.

Validieren einer EPUB-Datei

Um eine EPUB-Datei zu validieren, wählen Sie den Befehl **XML | XML validieren (F8)**. Die Ergebnisse der Validierung werden im Fenster "Meldungen" (*Abbildung unten*) angezeigt. Wenn eine der Archivdateien - ob Inhaltsdatei oder Strukturdatei - geändert, aber noch nicht gespeichert wurde, werden Sie aufgefordert die Datei zu speichern. Sie müssen die geänderten Dateien speichern, um die EPUB-Datei validieren zu können. Durch Aktivieren der entsprechenden Einstellung (über die Schaltfläche **Einstellungen** in der Symbolleiste der Archivansicht-Ansicht) können Sie festlegen, dass alle geänderten Dateien vor der Validierung automatisch gespeichert werden.



Anzeige in den Fehlermeldungen: (i) die Datei, in der der Fehler gefunden wurde und gegebenenfalls die Nummer der Zeile, in der der Fehler auftrat; (ii) eine Beschreibung des Fehlers. In der Abbildung oben tritt der markierte Fehler in Zeile 21 der Datei `content.opf` auf. Wenn Sie im Fenster "Meldungen" auf die Fehlerzeile klicken, wird die entsprechende Datei geöffnet und der Fehler darin markiert.

Anmerkung: Der EPUB-Validierungsprozessor ist ein Java-Tool. Daher muss Java auf Ihrem Rechner installiert sein, damit der Validierungsprozessor ausgeführt werden kann.

© 2016-2022 Altova GmbH

19 Datenbanken

Sie können in XMLSpy zu den verschiedensten Datenbanken (DB) eine Verbindung herstellen und anschließend Operationen wie z.B. Abfragen der Datenbank, Importieren der Datenbankstruktur als XML-Schema, Generieren eine XML-Datendatei anhand der Datenbank und Exportieren von Daten in eine Datenbank durchführen. Jede Funktion im Zusammenhang mit Datenbanken steht in XMLSpy als Menübefehl zur Verfügung und ist im Abschnitt Menübefehle[1275] dieser Dokumentation unter dem entsprechenden Befehl verfügbar. Im Folgenden finden Sie eine vollständige Liste dieser Befehle mit Links zu den entsprechenden Beschreibungen.

In diesem Abschnitt wird Folgendes beschrieben:

- Herstellung einer Verbindung zu einer Datenbank[969]: Dies ist eine Voraussetzung, um einen der XMLSpy Befehle für die Arbeit mit Datenbanken ausführen zu können.
- Liste aller Datenbanken[1059], die getestet wurden und erfolgreich mit XMLSpy verwendet werden können.

Anmerkung: Wenn Sie die 64-Bit-Version von XMLSpy verwenden, stellen Sie sicher, dass Sie Zugriff auf die 64-Bit-Datenbanktreiber haben, die für die entsprechenden Datenbank benötigt werden.

XMLSpy Funktionen für die Arbeit mit Datenbanken

Die XMLSpy Funktionen im Zusammenhang mit Datenbanken werden durch Befehle in den Menüs DB[1440] und Konvertieren[1470] ausgeführt.

- *Datenbank abfragen*[1440]*:* Im Menü **DB**. Lädt die Struktur der Datenbank in einem separaten Datenbankabfragefenster und ermöglicht Abfragen an der Datenbank. Die Ergebnisse werden im Fenster "Datenbankabfrage" angezeigt.
- *IBM DB2*[1456]*:* Im Menü **DB**. IBM DB2 ist eine XML-Datenbank. In XMLSpy können Sie die XML-Schemas der XML-Datenbank verwalten und die XML-Datenbank bearbeiten und validieren.
- *SQL Server*[1462]*:* Im Menü **DB**. Sie können die XML-Schemas der Datenbank in XMLSpy verwalten, bearbeiten und validieren.
- *Oracle XML DB*[1465]: Im Menü **DB**. Bietet eine Reihe von Funktionen für Oracle XML-Datenbanken, u.a. die Verwaltung von XML-Schemas, Datenbankabfragen und die Generierung von XML-Dateien anhand von Datenbankschemas.
- *Datenbankdaten importieren*[1473]: Im Menü **Konvertieren**. Importiert Datenbankdaten in eine XML-Datei.
- *XML-Schema von DB-Struktur erzeugen*[1478]: Im Menü **Konvertieren**. Generiert anhand der Struktur einer Datenbank ein XML-Schema.
- *DB-Import auf Basis von XML-Schema*[1483]: Im Menü **Konvertieren**. Während in XMLSpy ein XML-Schemadokument aktiv ist, wird eine Datenbankverbindung hergestellt und die Daten einer ausgewählten DB-Tabelle können importiert werden. Die Struktur des erzeugten XML-Dokuments basiert auf dem XML-Schema, das bei Herstellung der Datenbankverbindung aktiv war.
- *DB-Struktur von XML-Schema erzeugen*[1484]: Im Menü **Konvertieren**. Datenbanktabellen ohne Daten werden auf Basis der Struktur des bestehenden XML-Schemas erzeugt.
- *Export in Datenbank*[1490]: Im Menü **Konvertieren**. Daten aus einem XML-Dokument können in eine Datenbank exportiert werden. Bestehende Datenbanktabellen können mit den XML-Daten aktualisiert werden oder neue Tabellen können erstellt werden, die die XML-Daten enthalten.

Datentypkonvertierungen

Beim Konvertieren von Daten zwischen XML-Dokumenten und Datenbanken müssen Datentypen in Typen konvertiert werden, die mit dem entsprechenden Format kompatibel sind. In den Anhängen Datentypen in DB auf XML-Schema 2256 und Datentypen in DB, die anhand von XML-Schemas generiert wurden 2262 wird beschrieben, wie Datentypen in XMLSpy konvertiert werden.

Altova DatabaseSpy

Altova DatabaseSpy ist ein Multidatenbank-Abfrage und DB-Design-Tool, das zusätzlich zu den DB-Funktionen, die in XMLSpy zur Verfügung stehen, weitere DB-Funktionen enthält. Nähere Informationen zu Altova DatabaseSpy finden Sie auf der Altova Website.

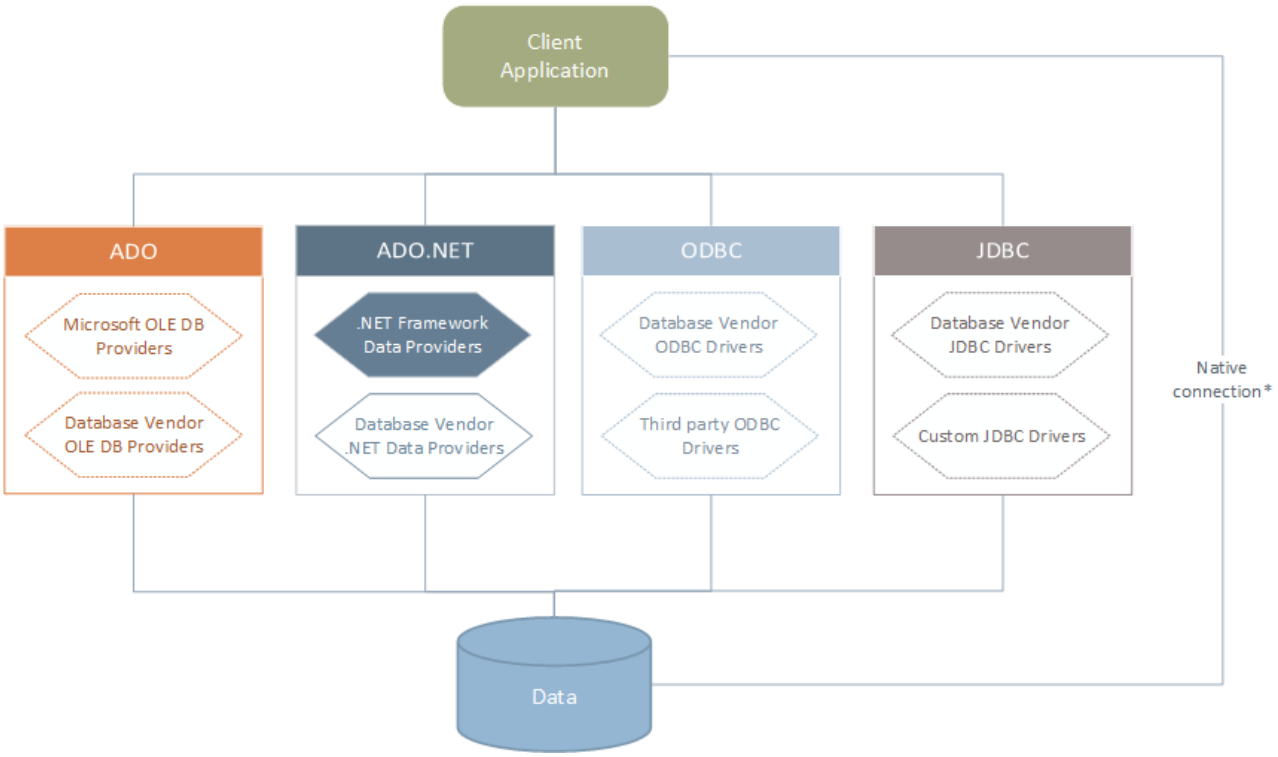
© 2016-2022 Altova GmbH

19.1 Herstellen einer Verbindung zu einer Datenquelle

Im einfachsten Fall kann es sich bei einer Datenbank um eine lokale Datei wie z.B. eine Microsoft Access- oder SQLite-Datenbankdatei handeln. In komplexeren Szenarien befindet sich die Datenbank manchmal auf einem entfernten Server oder eine Datenbank-Netzwerk-Server, auf dem nicht notwendigerweise dasselbe Betriebssystem wie das der damit verbundenen Applikation verwendet wird. Während z.B. XMLSpy auf einem Windows-System läuft, könnte die Datenbank, über die Sie die Daten aufrufen möchten (z.B. MySQL), auf einem Linux-Rechner installiert sein.

Für die Verbindung mit verschiedenen Datenbanktypen - sowohl entfernten und lokalen, werden in XMLSpy Datenverbindungsschnittstellen und Datenbanktreiber verwendet, die auf Ihrem Betriebssystem bereits vorhanden sind oder von denen regelmäßig aktualisierte Versionen von Anbietern gebräuchlicher Datenbanken bereitgestellt werden. Aufgrund sich ständig weiterentwickelnder Datenbanktechnologien bietet diese Methode bessere plattformübergreifende Flexibilität und Interoperabilität.

Im folgenden Diagramm werden Datenbankverbindungsoptionen zwischen XMLSpy (als allgemeine Client-Applikation dargestellt) und einem Datenspeicher (einem Datenbank-Server oder einer Datenbankdatei) dargestellt.



** Für SQLite-, PostgreSQL-, CouchDB- und MongoDB-Datenbanken werden direkte native Verbindungen unterstützt. Für die Verbindung mit solchen Datenbanken müssen keine zusätzlichen Treiber auf Ihrem System installiert werden.*

Wie im Diagramm oben gezeigt, kann XMLSpy zu jeder der gebräuchlichen Datenbankarten über die folgenden Technologien eine Verbindung herstellen:

- ADO (Microsoft® ActiveX® Data Objects), wofür wiederum ein zugrunde liegender OLE DB (Object Linking and Embedding, Database) Provider verwendet wird
- ADO.NET (eine Gruppe von im Microsoft .NET Framework verfügbaren Bibliotheken, die die Interaktion mit Daten ermöglichen)
- JDBC (Java Database Connectivity)
- ODBC (Open Database Connectivity)

Anmerkung: Einige ADO.NET-Anbieter werden nicht oder nur eingeschränkt unterstützt. Siehe Anmerkungen zur Unterstützung von ADO.NET[988].

Datenzugriffstechnologien

Welche Datenverbindungsschnittstelle Sie verwenden sollten, hängt größtenteils von der vorhandenen Software-Infrastruktur ab. Normalerweise werden Sie die Datenzugriffstechnolgie und den Datenbanktreiber verwenden, die enger mit dem gewünschten Datenbanksystem integriert sind. Um z.B. eine Verbindung zu einer Microsoft Access 2013-Datenbank herzustellen, würden Sie einen ADO Connection String erstellen, der einen nativen Provider wie z.B. den **Microsoft Office Access Database Engine OLE DB Provider** verwendet. Um eine Verbindung zu Oracle herzustellen, sollten Sie eventuell die neueste JDBC-, ODBC- oder ADO.NET-Schnittstelle von der Oracle Webseite herunterladen.

Während die Treiber für Windows-Produkte (wie z.B. Microsoft Access oder SQL Server) wahrscheinlich bereits auf Ihrem Windows Betriebssystem vorhanden sind, ist dies bei anderen Datenbanktypen möglicherweise nicht der Fall. Die wichtigsten Datenbankanbieter bringen regelmäßig öffentlich verfügbare Datenbank Client-Software und Treiber heraus, die durch beliebige Kombinationen von OLE DB, ODBC oder JDBC plattformübergreifenden Zugriff auf die jeweilige Datenbank ermöglichen. Zusätzlich dazu steht für jede der oben angeführten Technologien eine Reihe von Treibern von Drittanbietern zur Verfügung. In den meisten Fällen gibt es mehrere Möglichkeiten, um von Ihrem Betriebssystem und somit von XMLSpy aus, eine Verbindung zur gewünschten Datenbank herzustellen. Welche Funktionalitäten und Performance-Parameter zur Verfügung stehen und welche bekannten Einschränkungen es gibt, hängt normalerweise von der Datenzugriffstechnologie oder den Treibern, die Sie verwenden, ab.

19.1.1 Starten des Verbindungsassistenten

XMLSpy bietet einen Datenbankverbindungsassistenten, der Sie Schritt für Schritt durch das Herstellen einer Verbindung zu einer Datenquelle führt. Bevor Sie den Assistenten aufrufen, denken Sie daran, dass bei einige Datenbanktypen vorher einige Dinge wie z.B. ein Datenbanktreiber oder Datenbank Client-Software separat installiert und konfiguriert werden müssen. Diese erhalten Sie normalerweise vom jeweiligen Datenbankanbieter. Darin enthalten ist die Dokumentation zu Ihrer jeweiligen Windows-Version. Unter Übersicht über Datenbanktreiber[972] finden Sie eine Liste von Datenbanktreibern, gruppiert nach Datenbanktyp.

Um den Datenbankverbindungsassistenten (*siehe Abbildung unten*) zu starten, gehen Sie folgendermaßen vor:

- Klicken Sie im Menü **DB** auf **Datenbank abfragen**.

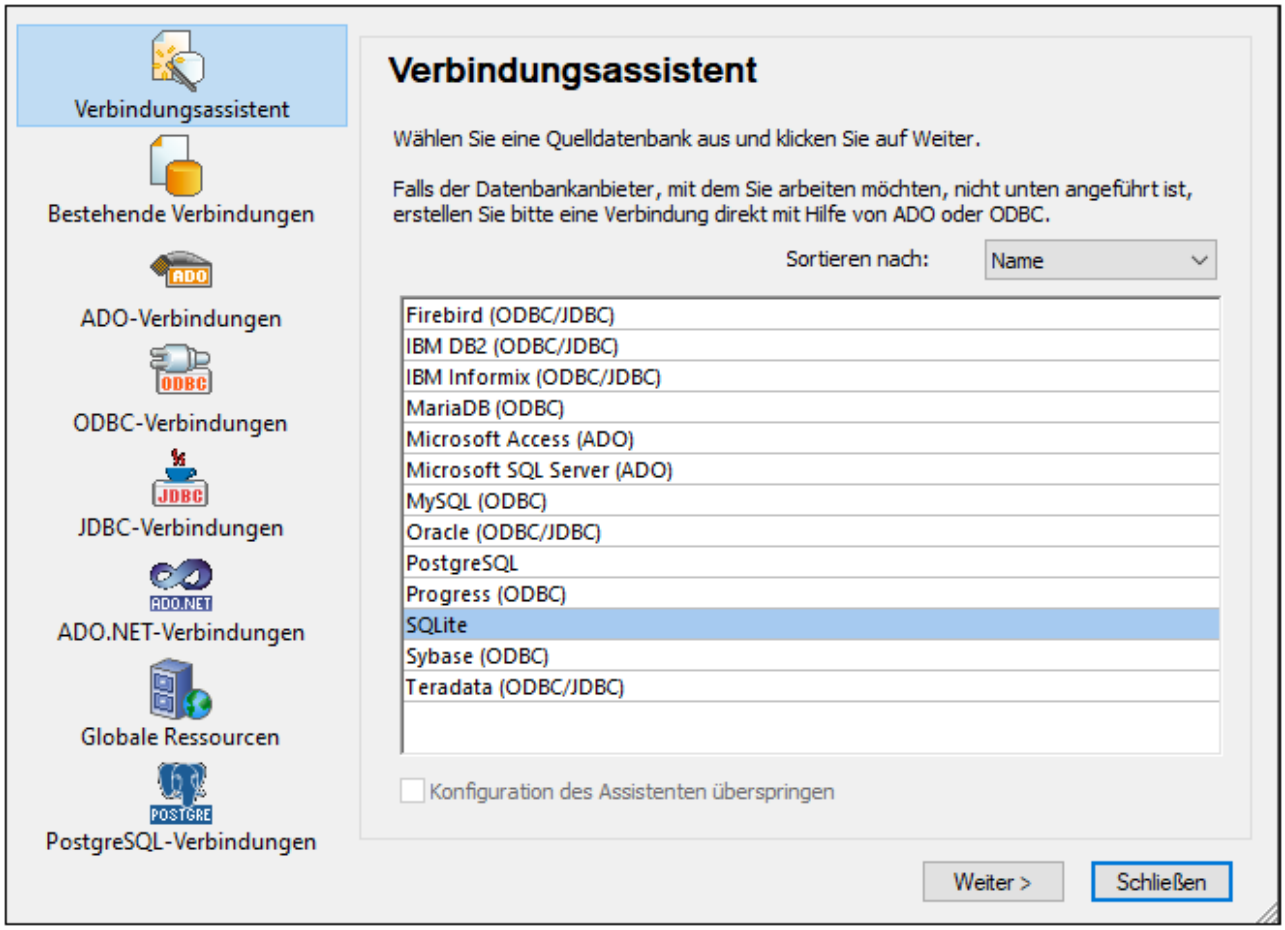
Daraufhin wird der Datenbankverbindungsassistent gestartet (*Abbildung unten*). Auf der linken Seite des Fensters können Sie aus den folgenden Verbindungsarten die am besten geeignete auswählen:

- Verbindungsassistent: Hier werden Sie aufgefordert, Ihren Datenbanktyp auszuwählen. Anschließend werden Sie Schritt für Schritt durch den Vorgang zur Herstellung einer Verbindung mit einer Datenbank dieses Typs geführt.

 © 2016-2022 Altova GmbH

- Auswahl einer bestehenden Verbindung
- Auswahl einer Datenzugriffstechnologie: ADO, ADO.NET, ODBC oder JDBC
- Auswahl einer globalen Altova-Ressource, in der die Datenbankverbindung gespeichert ist
- Native PostgreSQL-Verbindung

Die Datenbanken können im Verbindungsassistent-Fenster (*siehe Abbildung unten*) alphabetisch nach dem Namen des Datenbanktyps oder dem zuletzt verwendeten Datenbanktyp sortiert werden. Wählen Sie die gewünschte Option in der Auswahlliste *Sortieren nach* aus. Klicken Sie nach Auswahl des gewünschten Datenbanktyps auf **Weiter**.



Je nach gewähltem Datenbanktyp, nach gewählter Verbindungstechnologie (ADO, ADO.NET, ODBC, JDBC) und Treiber werden Sie vom Assistenten durch den Verbindungsvorgang geführt. Beispiele zum jeweiligen Datenbanktyp finden Sie im Abschnitt Beispiele für Datenbankverbindungen 1003.

Anstelle des Verbindungsassistenten können Sie eine der folgenden Datenbankzugriffstechnologien verwenden:

- Einrichten einer ADO-Verbindung 975
- Einrichten einer ADO.NET-Verbindung 982
- Einrichten einer ODBC-Verbindung 989
- Einrichten einer JDBC-Verbindung 992

19.1.2 Übersicht über Datenbanktreiber

Die folgende Tabelle enthält eine Liste gebräuchlicher Datenbanktreiber, über die Sie mit Hilfe einer bestimmten Datenzugriffstechnologie eine Verbindung zu einer bestimmten Datenbank herstellen können. Bitte beachten Sie, dass diese Liste keinen Anspruch auf Vollständigkeit erhebt und auch nicht zwingend befolgt werden muss; Sie können neben den unten angeführten Treibern auch andere native Treiber oder Produkte von Drittanbietern verwenden.

Standardmäßig stehen auf Windows Betriebssystemen zwar einige Datenbanktreiber bereits zur Verfügung, Sie müssen eventuell aber dennoch auch andere Treiber herunterladen und verwenden. Bei einigen Datenbanken empfiehlt es sich, anstelle des mit dem Betriebssystem mitgelieferten Treibers den neuesten Treiber des Datenbankanbieters zu verwenden.

Die meisten Datenbankanbieter bieten Treiber entweder als separat herunterladbare Pakete oder mit Datenbank-Client-Software gebündelt an. In letzterem Fall enthält die Datenbank Client-Software normalerweise alle erforderlichen Datenbanktreiber oder gibt Ihnen bei der Installation die Möglichkeit, die gewünschten Treiber und Komponenten auszuwählen. Datenbank Client-Software besteht normalerweise aus Verwaltungs- und Konfigurationstools zur einfacheren Verwaltung und Datenbankanbindung sowie der Dokumentation zum Installieren und Konfigurieren des Datenbank Client und seiner Komponenten.

Damit eine funktionierende Verbindung zur Datenbank hergestellt werden kann, muss der Datenbank Client unbedingt richtig konfiguriert werden. Bevor Sie die Datenbank Client-Software installieren, empfiehlt es sich, vorher die Installations- und Konfigurationsanleitungen zum Datenbank Client sorgfältig zu lesen, da diese normalerweise je nach Datenbank- und Windows-Version unterschiedlich ist.

Um die Möglichkeiten und Einschränkungen der einzelnen Datenzugriffstechnologien im Zusammenhang mit einzelnen Datenbanktypen zu verstehen, lesen Sie die Dokumentation zum jeweiligen Datenbankprodukt und testen Sie die Verbindung in Ihrer jeweiligen Rechnerumgebung. Beachten Sie die folgenden Hinweise und Empfehlungen, um häufige Verbindungsprobleme zu vermeiden:

- Einige ADO.NET-Anbieter werden nicht oder nur eingeschränkt unterstützt. Nähere Informationen dazu finden Sie unter Anmerkungen zur Unterstützung von ADO.NET 988.
- Bei Installation eines Datenbanktreibers wird empfohlen, dass dieser dieselbe Plattform wie die Altova-Applikation hat (32-Bit oder 64-Bit). Wenn Sie z.B. eine 32-Bit-Altova-Applikation auf einem 64-Bit-Betriebssystem verwenden, richten Sie ihre Datenbankverbindung mit der 32-Bit-Treiberversion ein, siehe auch Anzeigen der verfügbaren ODBC-Treiber 991.
- Es empfiehlt sich beim Einrichten einer ODBC-Datenquelle, im Allgemeinen den DSN (Data Source Name) als System-DSN und nicht als Benutzer-DSN zu erstellen. Nähere Informationen dazu finden Sie unter Einrichten einer ODBC-Verbindung 989.
- Stellen Sie beim Einrichten einer JDBC-Datenquelle sicher, dass JRE (Java Runtime Environment) installiert ist und dass die CLASSPATH-Umgebungsvariable des Betriebssystems konfiguriert ist. Nähere Informationen dazu finden Sie unter Einrichten einer JDBC-Verbindung 992.
- Nähere Informationen zur Installation und zu den Treibern oder der Datenbank Client-Software eines Datenbankanbieters finden Sie in der Dokumentation zum jeweiligen Installationspaket.

Datenbank	Schnittstelle	Treiber
Firebird	ADO.NET	Firebird ADO.NET-Datenanbieter (https://www.firebirdsql.org/en/additional-downloads/)

© 2016-2022 Altova GmbH

Datenbank	Schnittstelle	Treiber
	JDBC	Firebird JDBC-Treiber (https://www.firebirdsql.org/en/jdbc-driver/)
	ODBC	Firebird ODBC-Treiber (https://www.firebirdsql.org/en/odbc-driver/)
IBM DB2	ADO	IBM OLE DB-Anbieter für DB2
	ADO.NET	IBM Datenserver-Anbieter für .NET
	JDBC	IBM Datenserver-Treiber für JDBC und SQLJ
	ODBC	IBM DB2 ODBC-Treiber
IBM DB2 for i	ADO	• IBM DB2 für i5/OS IBMDA400 OLE DB-Anbieter • IBM DB2 für i5/OS IBMDARLA OLE DB-Anbieter • IBM DB2 für i5/OS IBMDASQL OLE DB-Anbieter
	ADO.NET	.NET Framework Data Provider für IBM i
	JDBC	IBM Toolbox für Java JDBC-Treiber
	ODBC	iSeries Access ODBC-Treiber
IBM Informix	ADO	IBM Informix OLE DB-Treiber
	JDBC	IBM Informix JDBC-Treiber
	ODBC	IBM Informix ODBC-Treiber
Microsoft Access	ADO	• Microsoft Jet OLE DB-Anbieter • Microsoft Access Database Engine OLE DB-Anbieter
	ADO.NET	.NET Framework-Datenanbieter für OLE DB
	ODBC	• Microsoft Access-Treiber
MariaDB	ADO.NET	Falls kein eigener .NET-Konnektor für MariaDB vorhanden ist, verwenden Sie **Connector/NET** für MySQL (https://dev.mysql.com/downloads/connector/net/).
	JDBC	MariaDB Connector/J (https://downloads.mariadb.org/)
	ODBC	MariaDB Connector/ODBC (https://downloads.mariadb.org/)
Microsoft SQL Server	ADO	• Microsoft OLE DB-Treiber für SQL Server (MSOLEDBSQL) • Microsoft OLE DB-Anbieter für SQL Server (SQLOLEDB) • SQL Server Native Client (SQLNCLI)
	ADO.NET	• .NET Framework-Datenanbieter für SQL Server • .NET Framework-Datenanbieter für OLE DB
	JDBC	• Microsoft JDBC-Treiber für SQL Server (https://docs.microsoft.com/en-us/sql/connect/jdbc/microsoft-jdbc-driver-for-sql-server)

Datenbank	Schnittstelle	Treiber
	ODBC	• ODBC-Treiber für Microsoft SQL Server (https://docs.microsoft.com/en-us/SQL/connect/odbc/download-odbc-driver-for-sql-server)
MySQL	ADO.NET	• Connector/NET (https://dev.mysql.com/downloads/connector/net/)
	JDBC	Connector/J (https://dev.mysql.com/downloads/connector/j/)
	ODBC	Connector/ODBC (https://dev.mysql.com/downloads/connector/odbc/)
Oracle	ADO	• Oracle-Anbieter für OLE DB • Microsoft OLE DB-Anbieter für Oracle
	ADO.NET	Oracle-Datenanbieter für .NET (http://www.oracle.com/technetwork/topics/dotnet/index-085163.html)
	JDBC	• JDBC Thin-Treiber • JDBC Oracle Call Interface (OCI)-Treiber Diese Treiber werden normalerweise während der Installation Ihres Oracle-Datenbank-Client installiert. Stellen Sie die Verbindung über den OCI-Treiber (und nicht den Thin-Treiber) her, wenn Sie die Oracle XML DB-Komponente verwenden.
	ODBC	• Microsoft ODBC für Oracle • Oracle ODBC-Teiber (wird normalerweise während der Installation Ihres Oracle-Datenbank-Client installiert)
PostgreSQL	JDBC	PostgreSQL JDBC-Treiber (https://jdbc.postgresql.org/download.html)
	ODBC	psqlODBC (https://odbc.postgresql.org/)
	Native Verbindung	Verfügbar. Bei Verwendung der nativen Verbindung müssen keine Treiber installiert werden.
Progress OpenEdge	JDBC	JDBC Connector (https://www.progress.com/jdbc/openedge)
	ODBC	ODBC Connector (https://www.progress.com/odbc/openedge)
SQLite	Native Verbindung	Verfügbar. Bei Verwendung der nativen Verbindung müssen keine Treiber installiert werden.
Sybase	ADO	Sybase ASE OLE DB-Anbieter
	JDBC	jConnect™ für JDBC
	ODBC	Sybase ASE ODBC-Treiber
Teradata	ADO.NET	.NET-Datenanbieter für Teradata (https://downloads.teradata.com/download/connectivity/net-data-provider-for-teradata)
	JDBC	Teradata JDBC-Treiber (https://downloads.teradata.com/download/connectivity/jdbc-driver)

© 2016-2022 Altova GmbH

Datenbank	Schnittstelle	Treiber
	ODBC	Teradata ODBC-Treiber für Windows (https://downloads.teradata.com/download/connectivity/odbc-driver/windows)

19.1.3 ADO-Verbindung

Microsoft ActiveX Data Objects (ADO) ist eine Datenzugriffstechnologie, mit der Sie über OLE DB eine Verbindung zu einer ganzen Reihe von Datenquellen herstellen können. OLE DB ist eine Alternativschnittstelle zu ODBC oder JDBC und ermöglicht einen einheitlichen Zugriff auf Daten in einer COM (Component Object Model)-Umgebung. ADO ist ein Vorläufer des neueren ADO.NET[982] und ist weiterhin eine der möglichen Methoden, um eine Verbindung zu nativen Microsoft-Datenbanken wie Microsoft Access oder SQL Server herzustellen, kann aber auch für andere Datenquellen eingesetzt werden.

Beachten Sie, dass Sie zwischen mehreren ADO-Anbietern wählen können. Einige davon müssen zuerst heruntergeladen und auf Ihrem Rechner installiert werden, bevor Sie sie verwenden können. Für die Verbindung mit SQL Server stehen z.B. die folgenden ADO-Anbieter zur Verfügung:

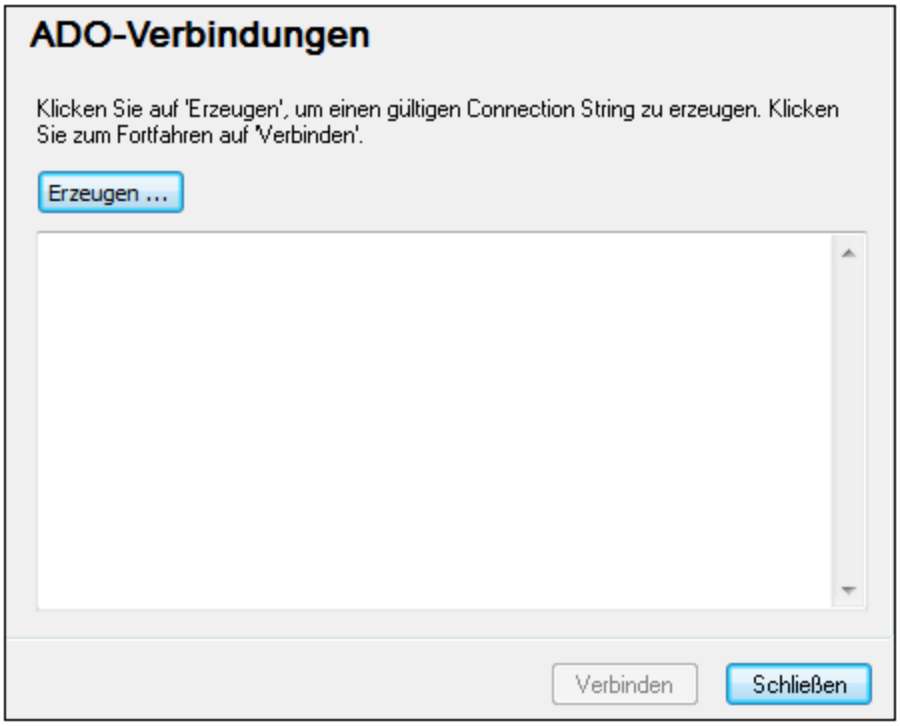
- Microsoft OLE DB-*Treiber* für SQL Server (MSOLEDBSQL)
- Microsoft OLE DB-*Anbieter* für SQL Server (SQLOLEDB)
- SQL Server Native Client (SQLNCLI)

Von den oben aufgelisteten Anbietern wird MSOLEDBSQL empfohlen; Sie können diesen von https://docs.microsoft.com/en-us/sql/connect/oledb/download-oledb-driver-for-sql-server?view=sql-server-ver15 herunterladen. Beachten Sie, dass er mit der Plattform von XMLSpy (32-Bit oder 64-Bit) übereinstimmen muss. Die Anbieter SQLOLEDB und SQLNCLI gelten als veraltet und werden daher nicht empfohlen.

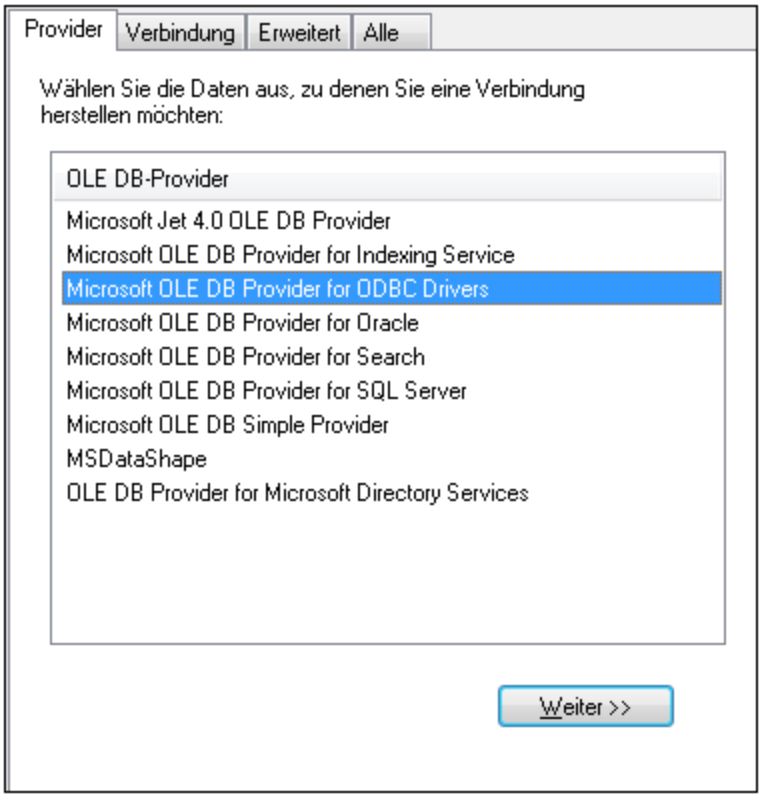
> Es ist bekannt, dass es beim **Microsoft OLE DB-Anbieter für SQL Server (SQLOLEDB)** zu Problemen mit der Parameterbindung komplexer Abfragen wie Common Table Expressions (CTE) und verschachtelten SELECT-Anweisungen kommt.

So richten Sie eine ADO-Verbindung ein:

1. Starten Sie den Datenbank-Verbindungsassistenten[970].
2. Klicken Sie auf **ADO-Verbindungen**.



3. Klicken Sie auf **Erzeugen**.



 © 2016-2022 Altova GmbH

4. Wählen Sie den Daten-Provider, über den Sie die Verbindung erstellen möchten. In der unten stehenden Tabelle sind einige häufige Szenarien aufgelistet.

Zum Verbinden mit dieser Datenbank...	Verwenden Sie diesen Anbieter...
Microsoft Access	• **Microsoft Office Access Database Engine OLE DB-Anbieter** (empfohlen) • **Microsoft Jet OLE DB-Anbieter** Wenn der **Microsoft Office Access Database Engine OLE DB-Anbieter** in der Liste nicht vorhanden ist, überprüfen Sie, ob Sie entweder Microsoft Access oder die Microsoft Access Database Engine Redistributable (https://www.microsoft.com/en-us/download/details.aspx?id=54920) auf Ihrem Rechner installiert haben.
SQL Server	• **Microsoft OLE DB-Treiber für SQL Server (MSOLEDBSQL)** - dies ist der empfohlene OLE DB-Anbieter. Damit dieser Anbieter in der Liste angezeigt wird, muss er von https://docs.microsoft.com/en-us/sql/connect/oledb/download-oledb-driver-for-sql-server?view=sql-server-ver15 heruntergeladen und installiert werden. • **Microsoft OLE DB-Anbieter für SQL Server (OLEDBSQL)** • **SQL Server Native Client (SQLNCLI)**
Andere Datenbank	Wählen Sie den für Ihre Datenbank benötigten Anbieter aus. Wenn für Ihre Datenbank kein OLE DB-Anbieter zur Verfügung steht, installieren Sie den erforderlichen Treiber des Datenbankanbieters (siehe Übersicht über Datenbanktreiber[972]). Alternativ dazu können Sie eine ADO.NET, ODBC- oder JDBC-Verbindung einrichten. Wenn das Betriebssystem über einen ODBC-Treiber für die gewünschte Datenbank verfügt, können Sie auch den **Microsoft OLE DB-Anbieter für ODBC-Treiber** verwenden oder sich vorzugsweise für eine ODBC-Verbindung[989] entscheiden.

5. Klicken Sie nach Auswahl des gewünschten Anbieters auf **Weiter** und stellen Sie den Assistenten fertig.

Die nächsten Schritte im Assistenten hängen vom verwendeten Anbieter ab. Bei SQL Server müssen Sie den Host-Namen des Datenbankservers angeben oder auswählen sowie die Authentifizierungsmethode, den Datenbanknamen und den Datenbank-Benutzernamen und das Passwort dafür. Ein Beispiel dafür finden Sie unter Verbinden mit Microsoft SQL Server (ADO)[1025]. Bei Microsoft Access müssen Sie zur Datenbankdatei navigieren bzw. den Pfad dafür angeben. Ein Beispiel dafür finden Sie unter Verbinden mit Microsoft Access (ADO)[1023].

Auf dem Register **Alle** des Verbindungsdialogfelds finden Sie die vollständige Liste der Initialisierungseigenschaften (Verbindungsparameter). Diese Eigenschaften sind je nach gewähltem Anbieter

unterschiedlich und müssen eventuell explizit konfiguriert werden, um die Verbindung herstellen zu können. In den folgenden Abschnitten finden Sie eine Anleitung, wie Sie die grundlegenden Initialisierungseigenschaften für Microsoft Access- und SQL Server-Datenbanken konfigurieren:

- Einrichten der SQL Server-Datenverknüpfungseigenschaften 979
- Einrichten der Microsoft Access-Datenverknüpfungseigenschaften 980

19.1.3.1 Herstellen einer Verbindung zu einer vorhandenen Microsoft Access-Datenbank

Gehen Sie auf diese Art und Weise vor, wenn Sie eine Verbindung zu einer nicht durch ein Passwort geschützten Microsoft Access-Datenbank herstellen möchten. Wenn die Datenbank passwortgeschützt ist, richten Sie das Datenbankpasswort, wie unter Verbinden mit Microsoft Access (ADO) 1023 beschrieben, ein.

So stellen Sie eine Verbindung zu einer vorhandenen Microsoft Access-Datenbank her:

1. Starten Sie den Datenbankverbindungsassistenten (siehe Starten des Datenbankverbindungsassistenten 970).
2. Wählen Sie **Microsoft Access (ADO)** aus und klicken Sie auf **Weiter**.



3. Wählen Sie die Option **Vorhandene MS Access Datenbank benutzen** aus.
4. Navigieren Sie zur Datenbankdatei oder geben Sie (entweder den relativen oder den absoluten) Pfad ein.
5. Klicken Sie auf **Verbinden**.

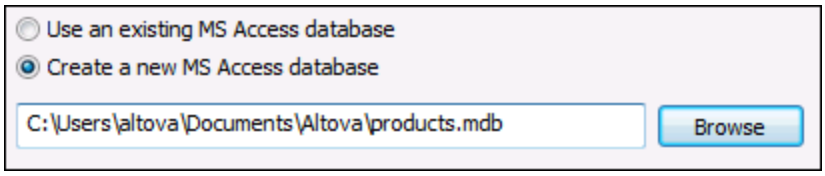
19.1.3.2 Erstellen einer neuen Microsoft Acccess-Datenbank

Als Alternative zum Herstellen einer Verbindung zu einer vorhandenen Datenbankdatei können Sie auch eine neue Microsoft Access-Datenbankdatei (.accdb, .mdb) erstellen und eine Verbindung zu dieser herstellen, auch wenn Microsoft Access auf Ihrem Rechner nicht installiert ist. Die von XMLSpy erstellte Datenbank ist leer. Um die benötigte Datenbankstruktur zu erstellen, verwenden Sie Microsoft Access oder ein Tool wie DatabaseSpy (https://www.altova.com/de/databasespy.html).

So erstellen Sie eine neue Microsoft Access-Datenbank:

1. Starten Sie den Datenbankverbindungsassistenten (siehe Starten des Datenbankverbindungsassistenten 970).
2. Wählen Sie **Microsoft Access (ADO)** aus und klicken Sie auf **Weiter**.

© 2016-2022 Altova GmbH



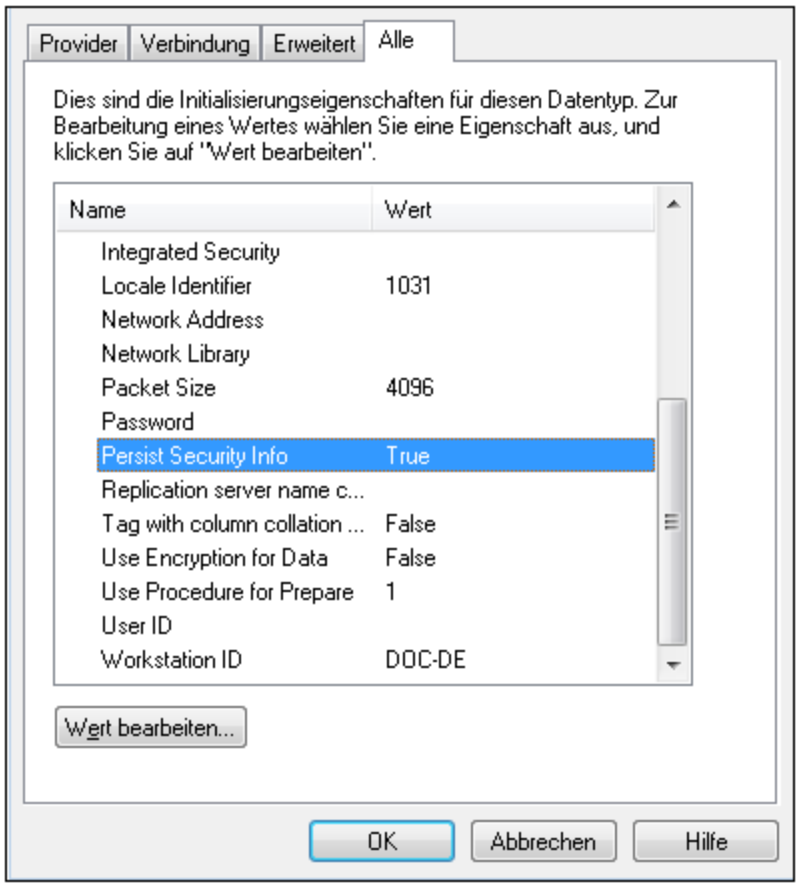
3. Wählen Sie **Neue MS Access-Datenbank erstellen** und geben Sie (entweder den relativen oder den absoluten) Pfad zur Datenbank ein (z.B. **c:\users\public\products.mdb**). Klicken Sie alternativ dazu auf **Durchsuchen**, um einen Ordner auszuwählen, geben Sie den Namen der Datenbankdatei in das Textfeld "Dateiname" ein (z.B. **products.mdb**) und klicken Sie auf **Speichern**.

> **Anmerkungen**
>
> - Sie benötigen unbedingt Schreibrechte für den Ordner, in dem Sie die Datenbankdatei erstellen möchten.
> - Der Name der Datenbankdatei muss die Erweiterung **.mdb** oder **.accdb** haben.

4. Klicken Sie auf **Verbinden**.

19.1.3.3 Einrichten der SQL Server-Datenverknüpfungseigenschaften

Wenn Sie über ADO[975] eine Verbindung zu einer Microsoft SQL Server-Datenbank herstellen, müssen Sie eventuell die folgenden Datenverknüpfungseigenschaften auf dem Register **Alle** des Dialogfelds "Datenverknüpfungseigenschaften" konfigurieren.



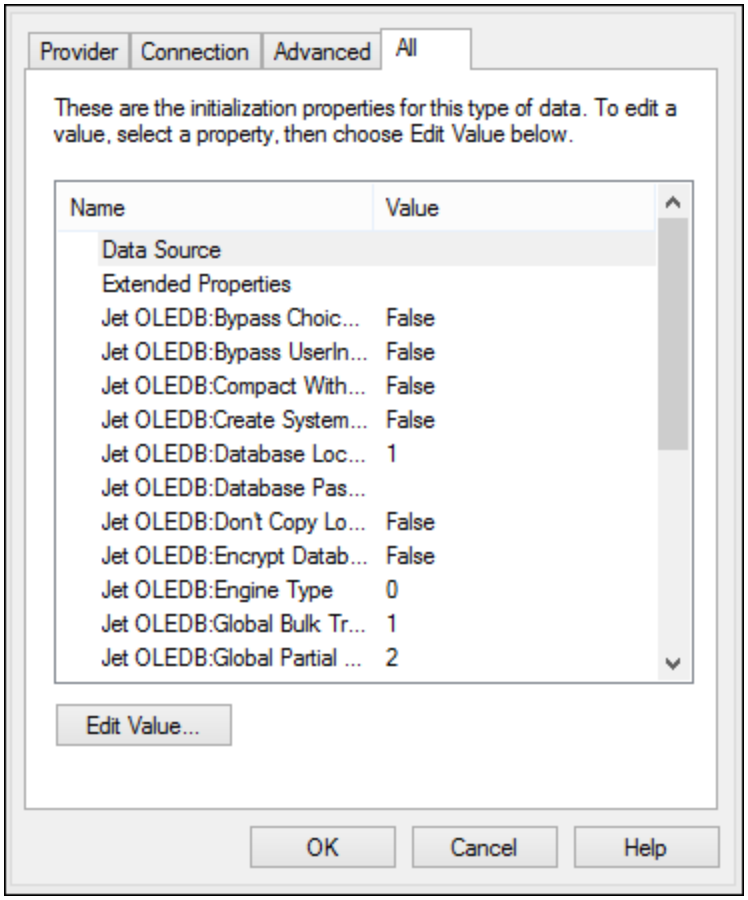
Dialogfeld "Datenverknüpfungseigenschaften"

Eigenschaft	Anmerkungen
Integrated Security	Wenn Sie den Daten-Provider **SQL Server Native Client** auf dem Register **Provider** auswählen, definieren Sie für diese Eigenschaft ein Leerzeichen.
Persist Security Info	Setzen Sie diese Eigenschaft auf **True**.

19.1.3.4 Einrichten der Microsoft Access-Datenverknüpfungseigenschaften

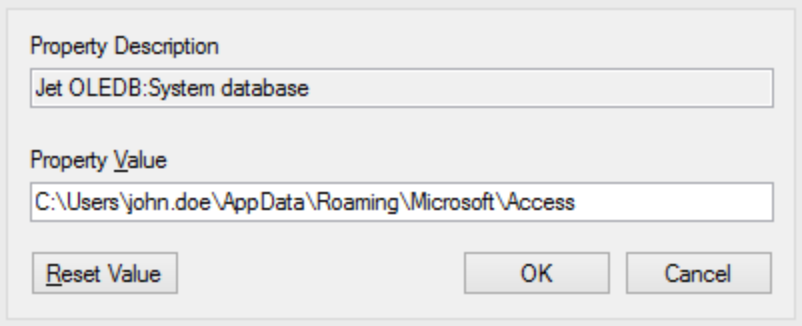
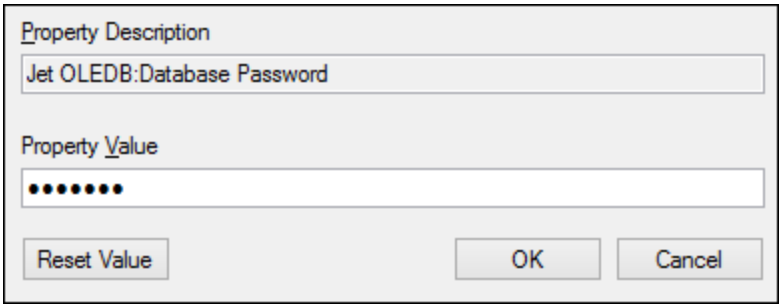
Wenn Sie über ADO[975] eine Verbindung zu einer Microsoft Access-Datenbank herstellen, müssen Sie eventuell die folgenden Datenverknüpfungseigenschaften auf dem Register **Alle** des Dialogfelds "Datenverknüpfungseigenschaften" konfigurieren.

© 2016-2022 Altova GmbH



Dialogfeld "Datenverknüpfungseigenschaften"

Eigenschaft	**Anmerkungen**
Datenquelle	In dieser Eigenschaft ist der Pfad zur Microsoft Access-Datenbankdatei gespeichert. Um Verbindungsprobleme zu vermeiden, wird empfohlen, das UNC (Universal Naming Convention)-Pfadformat zu verwenden, z.B.: `\\anyserver\share$\filepath`
Jet OLEDB:Systemdatenbank	In dieser Eigenschaft ist der Pfad zur Arbeitsgruppen-Informationsdatei gespeichert. Eventuell muss der Wert dieser Eigenschaft explizit definiert werden, bevor Sie eine Verbindung zu einer Microsoft Access-Datenbank herstellen können. Wenn die Verbindung aufgrund eines "Arbeitsgruppen-Informationsdatei"-Fehlers fehlschlägt, suchen Sie die Arbeitsgruppen-Informationsdatei (**System.MDW**) für Ihr Benutzerprofil und setzen Sie den Eigenschaftswert auf den Pfad der Datei **System.MDW**.

	
Jet OLEDB:Datenbankkennwort	Wenn die Datenbank durch ein Passwort geschützt ist, definieren Sie als Wert dieser Eigenschaft das Datenbank-Passwort. 

19.1.4 ADO.NET-Verbindung

ADO.NET ist eine Gruppe von Microsoft .NET Framework-Bibliotheken für die Interaktion mit Daten, darunter auch mit Daten aus Datenbanken. Für die Verbindung zu einer Datenbank von XMLSpy aus über ADO.NET wird Microsoft .NET Framework 4 oder höher benötigt. Wie unten gezeigt, erfolgt die Verbindung zu einer Datenbank über ADO.NET durch Auswahl eines .NET-Anbieters und Bereitstellung eines Connection String.

Ein .NET-Datenanbieter ist eine Sammlung von Klassen, mit Hilfe derer Sie eine Verbindung zu einem bestimmten Datenquellentyp (z.B. einem SQL Server oder einer Oracle-Datenbank) herstellen können, Befehle daran ausführen und Daten aus dieser Quelle abrufen können, d.h. mit Hilfe von ADO.NET kann eine Applikation wie XMLSpy über einen Datenanbieter mit einer Datenbank kommunizieren. Jeder Datenanbieter ist für den spezifischen Datenquellentyp, für den er entwickelt wurde, optimiert. Es gibt zwei Arten von .NET-Anbietern:

1. Solche, die standardmäßig mit Microsoft .NET Framework bereitgestellt werden.
2. Solche, die von Anbietern gebräuchlicher Datenbanken als Erweiterung zum .NET Framework bereitgestellt werden. ADO.NET-Anbieter dieser Art müssen separat installiert werden und können normalerweise von der Website des entsprechenden Datenbankanbieters heruntergeladen werden.

Anmerkung: Einige ADO.NET-Anbieter werden nicht oder nur eingeschränkt unterstützt. Siehe Anmerkungen zur Unterstützung von ADO.NET 988.

 © 2016-2022 Altova GmbH

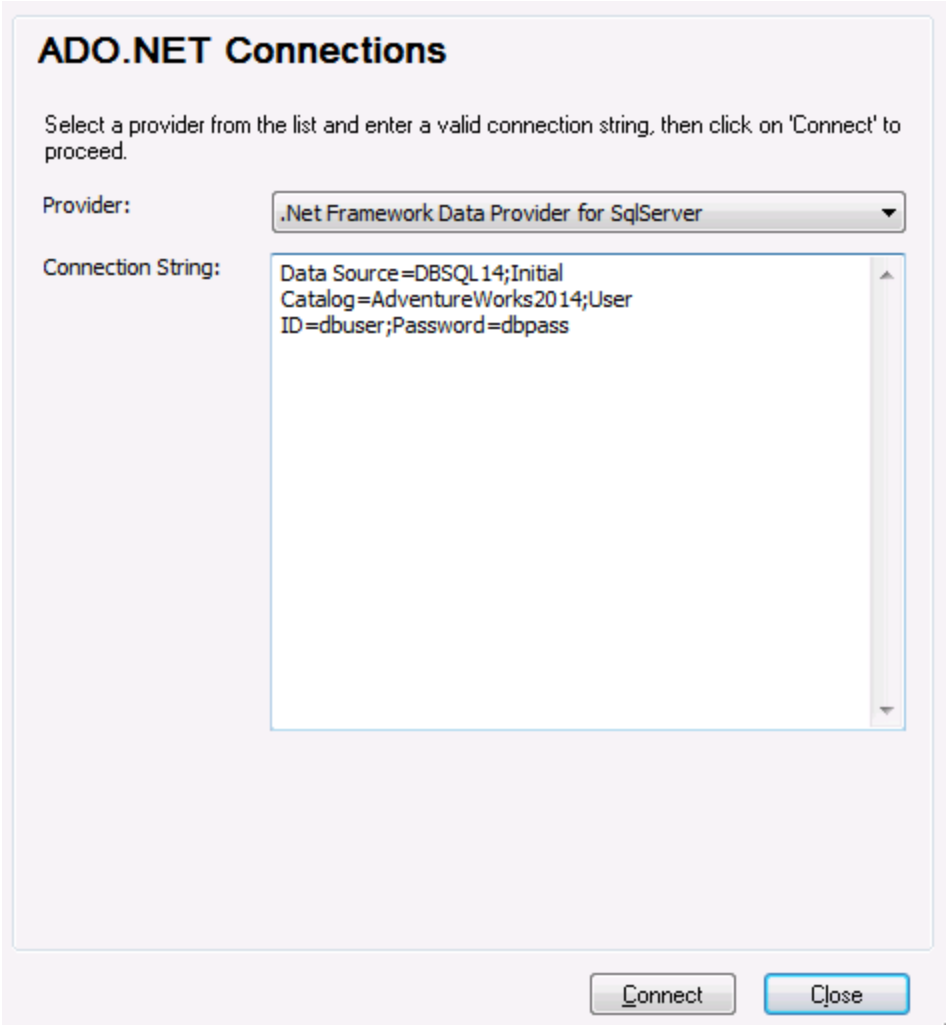
So richten Sie eine ADO.NET-Verbindung ein:

1. Starten Sie den Datenbankverbindungsassistenten[970].
2. Klicken Sie auf **ADO.NET-Verbindungen**.
3. Wählen Sie einen .NET-Datenanbieter aus der Liste aus.

   Die Liste der Anbieter, die standardmäßig mit dem .NET Framework zur Verfügung stehen, wird in der Liste der Anbieter angezeigt. Anbieterspezifische .NET-Datenanbieter stehen in der Liste nur zur Verfügung, wenn sie bereits auf Ihrem System installiert sind. Damit anbieterspezifische .NET-Anbieter zur Verfügung stehen, müssen diese durch Ausführung der vom Datenbankanbieter bereitgestellten .msi- oder .exe-Datei im GAC (Global Assembly Cache) installiert werden.

4. Geben Sie einen Datenbank-Connection String ein. Mit einem Connection String werden die Datenbankverbindungsinformationen in Form von durch Semikola getrennte Schlüssel/Wert-Paare von Verbindungsparametern definiert. Mit einem Connection String wie `Data Source=DBSQLSERV;Initial Catalog=ProductsDB;User ID=dbuser;Password=dbpass` wird z.B. eine Verbindung zur SQL Server-Datenbank `ProductsDB` auf dem Server `DBSQLSERV` unter dem Benutzernamen `dbuser` und mit dem Passwort `dbpass` hergestellt. Sie können einen Connection String erstellen, indem Sie die Schlüssel/Wert-Paare direkt in das Dialogfeld "Connection String" eingeben. Eine weitere Methode ist die Erstellung über Visual Studio (siehe Erstellen eines Connection String in Visual Studio[984]).

   Die Syntax des Connection String ist von dem in der Liste "Anbieter" ausgewählten Anbieter abhängig. Beispiele dazu finden Sie unter Beispiele für ADO.NET Connection Strings[987].



5. Klicken Sie auf **Verbinden**.

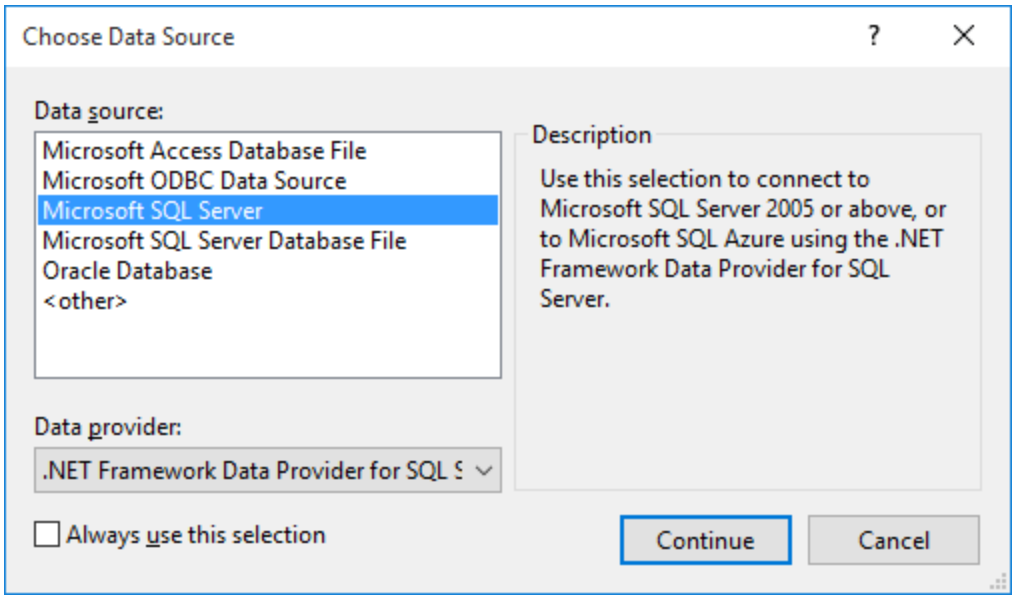
19.1.4.1 Erstellen eines Connection String in Visual Studio

Um über ADO.NET eine Verbindung zu einer Datenquelle herstellen zu können, wird ein gültiger Datenbank Connection String benötigt. Im Folgenden wird beschrieben, wie Sie über Visual Studio einen Connection String erstellen.

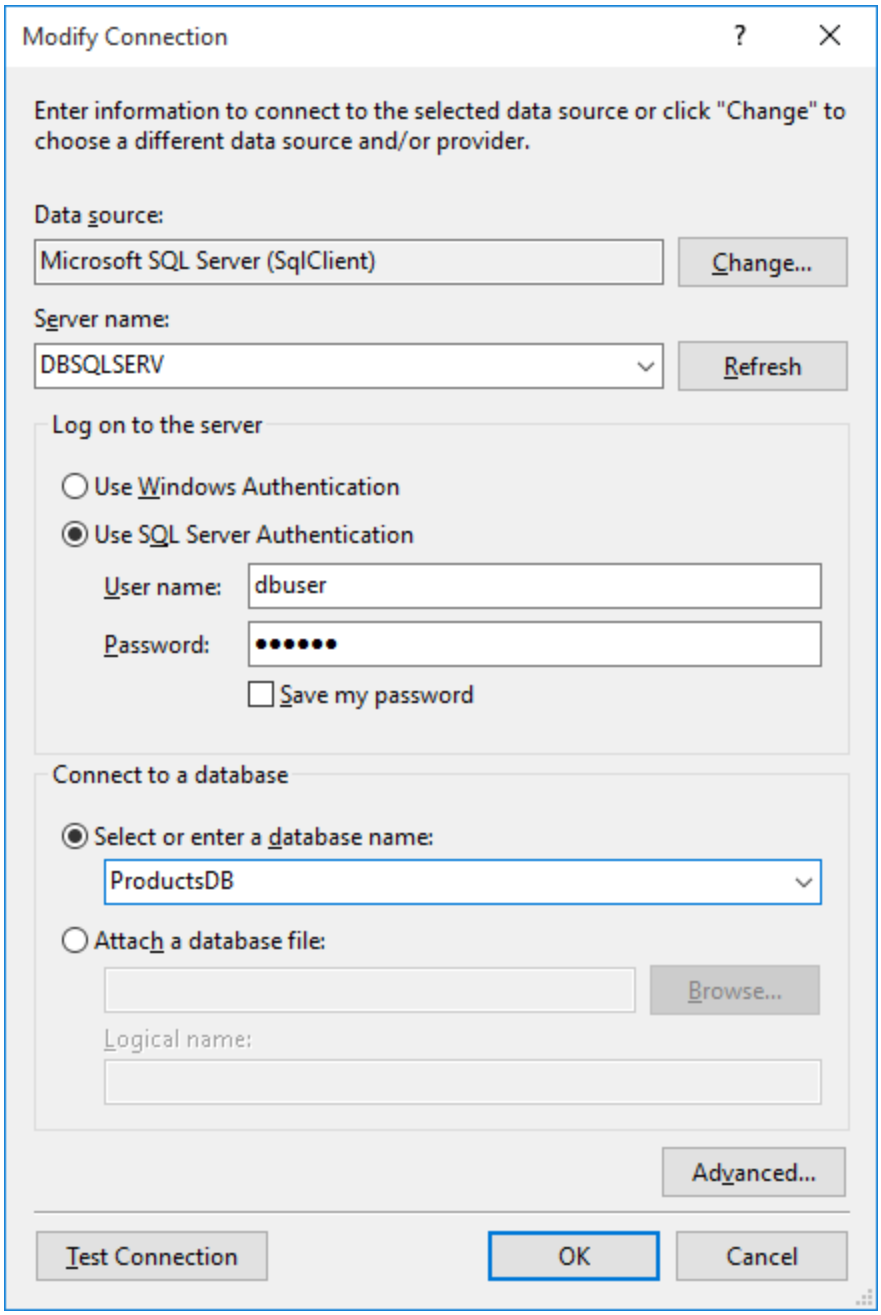
So erstellen Sie einen Connection String in Visual Studio:

1. Klicken Sie im Menü **Extras** auf **Mit Datenbank verbinden**.
2. Wählen Sie eine Datenquelle aus der Liste aus (in diesem Beispiel Microsoft SQL Server). Der Datenanbieter wird auf Basis Ihrer Auswahl automatisch ausgefüllt.

© 2016-2022 Altova GmbH



3. Klicken Sie auf **Weiter**.



4. Geben Sie den Server Host-Namen und den Benutzernamen und das Passwort für die Datenbank ein. In diesem Beispiel stellen wir unter Verwendung der SQL Server-Authentifizierung eine Verbindung zur Datenbank `ProductsDB` auf dem Server `DBSQLSERV` her.
5. Klicken Sie auf **OK**.

Wenn die Datenbankverbindung erfolgreich hergestellt wurde, wird sie im Server Explorer-Fenster angezeigt. Sie können das Server-Explorer-Fenster mit dem Menübefehl **Ansicht | Server-Explorer** aufrufen. Um den Datenbank Connection String anzuzeigen, klicken Sie im Server-Explorer-Fenster mit der rechten Maustaste auf die Verbindung und wählen Sie **Eigenschaften**. Der Connection String wird nun im Fenster "Eigenschaften"

© 2016-2022 Altova GmbH

von Visual Studio angezeigt. Beachten Sie, dass Sie das Sternchen (*) vor dem Einfügen des String in das Feld "Connection String" von XMLSpy durch das tatsächliche Passwort ersetzen müssen.

19.1.4.2 Beispiele für ADO.NET Connection Strings

Um eine ADO.NET-Verbindung einzurichten, müssen Sie einen ADO.NET-Anbieter aus dem Datenbankverbindungsdialogfeld auswählen und einen Connection String eingeben (siehe auch Einrichten einer ADO.NET-Verbindung 982). Beispiele für ADO.NET Connection Strings für verschiedene Datenbanken sind unter dem jeweiligen .NET-Anbieter aufgelistet.

.NET Data Provider für Teradata

Dieser Anbieter kann von der Teradata-Website (https://downloads.teradata.com/download/connectivity/net-data-provider-for-teradata) heruntergeladen werden. Ein Beispiel-Connection String sieht folgendermaßen aus:

```
Data Source=ServerAddress;User Id=user;Password=password;
```

.NET Framework Data Provider für IBM i

Dieser Anbieter wird im Rahmen von *IBM i Access Client Solutions - Windows Application Package*. installiert. Ein Connection String sieht folgendermaßen aus:

```
DataSource=Serveradresse;UserID=Benutzer;Password=Passwort;DataCompression=True;
```

Nähere Informationen dazu finden Sie in der im obigen Installationspaket inkludierten Hilfedatei ".NET Provider Technical Reference".

.NET Framework Data Provider für MySQL

Dieser Anbieter kann von der MySQL Website (https://dev.mysql.com/downloads/connector/net/) heruntergeladen werden. Ein Beispiel-Connection String sieht folgendermaßen aus:

```
Server=127.0.0.1;Uid=root;Pwd=12345;Database=test;
```

Siehe auch: https://dev.mysql.com/doc/connector-net/en/connector-net-programming-connecting-connection-string.html

.NET Framework Data Provider für SQL Server

Ein Beispiel-Connection String sieht folgendermaßen aus:

```
Data Source=DBSQLSERV;Initial Catalog=ProductsDB;User ID=dbuser;Password=dbpass
```

Siehe auch: https://msdn.microsoft.com/en-us/library/ms254500(v=vs.110).aspx

IBM DB2 Data Provider 10.1.2 für .NET Framework 4.0

```
Database=PRODUCTS;UID=user;Password=password;Server=localhost:50000;
```

Anmerkung: Dieser Anbieter wird normalerweise mit dem IBM DB2 Data Server Client-Paket installiert. Wenn der Anbieter nach Installation des IBM DB2 Date Server Client-Pakets nicht in der Liste der ADO.NET-Anbieter aufgelistet wird, lesen Sie nach unter: https://www-01.ibm.com/support/docview.wss?uid=swg21429586.

Siehe auch:
https://www.ibm.com/support/knowledgecenter/en/SSEPGG_10.1.0/com.ibm.swg.im.dbclient.adonet.ref.doc/doc/DB2ConnectionClassConnectionStringProperty.html

Oracle Data Provider für .NET (ODP.NET)

Das Installationspaket, das den ODP.NET-Anbieter enthält, kann von der Oracle Website heruntergeladen werden (siehe http://www.oracle.com/technetwork/topics/dotnet/downloads/index.html). Ein Beispiel-Connection String sieht folgendermaßen aus:

```
Data Source=DSORCL;User Id=user;Password=password;
```

DSORCL ist hierbei der Name der Datenquelle, der auf einen in der Datei **tnsnames.ora** definierten Oracle-Dienstnamen verweist. Eine Beschreibung dazu finden Sie unter Verbinden mit Oracle (ODBC)[1038].

Um eine Verbindung herzustellen, ohne einen Dienstnamen in der Datei **tnsnames.ora** zu konfigurieren, verwenden Sie einen String wie den folgenden:

```
Data Source=(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=host)(PORT=port)))
(CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=MyOracleSID)));User
Id=user;Password=password;
```

Siehe auch: https://docs.oracle.com/cd/B28359_01/win.111/b28375/featConnecting.htm

19.1.4.3 Anmerkungen zur Unterstützung von ADO.NET

In der folgenden Tabelle sind bekannte ADO.NET-Datenbanktreiber aufgelistet, die derzeit in XMLSpy nicht oder nur eingeschränkt unterstützt werden.

Datenbank	Treiber	Anmerkungen zur Unterstützung
Alle Datenbanken	**.Net Framework Data Provider for ODBC**	Eingeschränkte Unterstützung. Bekannte Probleme bei Microsoft Access-Verbindungen. Es wird empfohlen stattdessen direkte ODBC-Verbindungen zu verwenden.

© 2016-2022 Altova GmbH

Datenbank	Treiber	Anmerkungen zur Unterstützung
	.Net Framework Data Provider for OleDb	Eingeschränkte Unterstützung. Bekannte Probleme bei Microsoft Access-Verbindungen. Es wird empfohlen stattdessen direkte ADO-Verbindungen zu verwenden.
Firebird	**Firebird ADO.NET Data Provider**	Eingeschränkte Unterstützung. Es wird empfohlen stattdessen ODBC oder JDBC zu verwenden.
Informix	**IBM Informix Data Provider for .NET Framework 4.0**	Wird nicht unterstützt. Verwenden Sie stattdessen **DB2 Data Server Provider**.
IBM DB2 for i (iSeries)	**.Net Framework Data Provider for i5/OS**	Wird nicht unterstützt. Verwenden Sie stattdessen den im Rahmen des *IBM i Access Client Solutions - Windows Application-Pakets* bereitgestellten **.Net Framework Data Provider for IBM i-**Treiber.
Oracle	**.Net Framework Data Provider for Oracle**	Eingeschränkte Unterstützung. Der Treiber wird zwar mit dem .NET Framework zur Verfügung gestellt, doch wird von Microsoft von der Verwendung abgeraten, da er veraltet ist.
PostgreSQL	**-**	Es werden keine ADO.NET-Treiber für diesen Anbieter unterstützt. Verwenden Sie stattdessen eine native Verbindung.
Sybase	**-**	Es werden keine ADO.NET-Treiber für diesen Anbieter unterstützt.

19.1.5 ODBC-Verbindung

ODBC (Open Database Connectivity) ist eine häufig verwendete Datenzugriffstechnologie, mit der Sie von XMLSpy aus eine Verbindung zu einer Datenbank herstellen können. ODBC kann entweder als primäre Verbindungsmethode oder als Alternative zu nativen Verbindungen oder Verbindungen über OLE DB oder JDBC verwendet werden.

Um über ODBC eine Datenbankverbindung herzustellen, müssen Sie zuerst einen ODBC-Datenquellennamen (DSN = Data Source Name) auf dem Betriebssystem erstellen. Wenn bereits ein DSN erstellt wurde - möglicherweise von einem anderen Benutzer auf dem Betriebssystem - entfällt dieser Schritt. Der DSN bietet eine einheitliche Methode, um die Datenbankverbindung für jede ODBC-fähige Client-Applikation auf dem Betriebssystem einschließlich XMLSpy zu beschreiben. Es gibt folgende Arten von DSN:

- System-DSN
- Benutzer-DSN
- Datei-DSN

Eine *Systemdatenquelle* kann von allen Benutzern mit Rechten auf dem Betriebssystem aufgerufen werden. Eine *Benutzerdatenquelle* steht nur dem Benutzer, der sie erstellt hat, zur Verfügung. Wenn Sie einen *Datei-DSN* erstellen, wird die Datenquelle als Datei mit der Erweiterung .dsn erstellt, die Sie gemeinsam mit anderen Benutzern verwenden können, vorausgesetzt diese haben die für die Datenquelle erforderlichen Treiber installiert.

Alle auf Ihrem Rechner bereits verfügbaren DSNs werden im Dialogfeld "Datenbankverbindung" aufgelistet, wenn Sie im Dialogfeld "ODBC-Verbindungen" auf **ODBC-Verbindungen** klicken.



Dialogfeld "ODBC-Verbindungen"

Wenn zur gewünschten Datenbank kein DSN vorhanden ist, hilft Ihnen der XMLSpy Datenbank-Verbindungsassistent dabei, einen zu erstellen; Sie können den DSN aber auch direkt in Ihrem Windows-Betriebssystem erstellen. Stellen Sie in jedem Fall, bevor Sie fortfahren, sicher, dass der für die Datenbank erforderliche ODBC-Treiber in der Liste der ODBC-Treiber zur Verfügung steht (siehe Anzeigen der verfügbaren ODBC-Treiber 991).

So stellen Sie mit Hilfe eines neuen DSN eine Verbindung her:

1. Starten Sie den Datenbank-Verbindungsassistenten 970.
2. Klicken Sie im Dialogfeld "Datenbankverbindung" auf **ODBC-Verbindungen**.
3. Wählen Sie einen Datenquellentyp aus (Benutzer-DSN, System-DSN, Datei-DSN).

> Zur Erstellung eines System-DSN benötigen Sie Administratorrechte auf Ihrem Betriebssystem und XMLSpy muss von Ihnen als Adminstrator ausgeführt werden.

© 2016-2022 Altova GmbH

4. Klicken Sie auf **Hinzufügen** .
5. Wählen Sie einen Treiber aus und klicken Sie anschließend auf **Benutzer-DSN** oder **System-DSN** (je nachdem, welche Art von DSN Sie erstellen möchten). Wenn der Treiber für Ihre Datenbank nicht aufgelistet ist, laden Sie ihn vom Datenbankanbieter herunter und installieren Sie ihn (siehe Übersicht über Datenbanktreiber 972).
6. Füllen Sie im Dialogfeld, das daraufhin angezeigt wird, alle treiberspezifischen Informationen aus, um die Verbindung fertig zu konfigurieren.

Damit eine Verbindung hergestellt werden kann, müssen Sie den Host-Namen (oder die IP-Adresse) des Datenbankservers sowie den Datenbank-Benutzernamen und das Passwort dafür angeben. Eventuell gibt es weitere optionale je nach Anbieter unterschiedliche Verbindungsparameter. Nähere Informationen zu den Parametern für die einzelnen Verbindungsmethoden finden Sie in der Dokumentation des Treiberanbieters. Sobald der DSN erstellt wurde, steht er in der Liste der Datenquellennamen zur Verfügung. Auf diese Art können Sie die Datenbankverbindungsinformationen jedes Mal, wenn Sie eine Verbindung zur Datenbank herstellen, wiederverwenden. Beachten Sie, dass Benutzer-DSNs zur Liste der Benutzer-DSNs hinzugefügt werden, während System-DSNs zur Liste der System-DSNs hinzugefügt werden.

So stellen Sie über einen vorhandenen DSN eine Verbindung her:

1. Starten Sie den Datenbank-Verbindungsassistenten 970.
2. Klicken Sie auf **ODBC-Verbindungen**.
3. Wählen Sie einen Datenquellentyp aus (Benutzer-DSN, System-DSN, Datei-DSN).
4. Klicken Sie auf den vorhandenen DSN-Eintrag und anschließend auf **Verbinden.**

So erzeugen Sie auf Basis einer vorhandenen .dsn-Datei einen Connection String:

1. Starten Sie den Datenbank-Verbindungsassistenten 970.
2. Klicken Sie auf **ODBC-Verbindungen**.
3. Wählen Sie **Connection String erstellen** und klicken Sie anschließend auf **Erzeugen**.
4. Wenn Sie den Connection String mit Hilfe eines Datei-DSN erstellen möchten, klicken Sie auf das Register **Dateidatenquelle**. Klicken Sie andernfalls auf das Register **Computerdatenquelle**. (System-DSNs und Benutzer-DSNs werden als "Computerdatenquelle" bezeichnet.)
5. Wählen sie die benötigte .dsn-Datei aus und klicken Sie auf **OK.**

So stellen Sie die Verbindung mittels eines bereits definierten Connection String her:

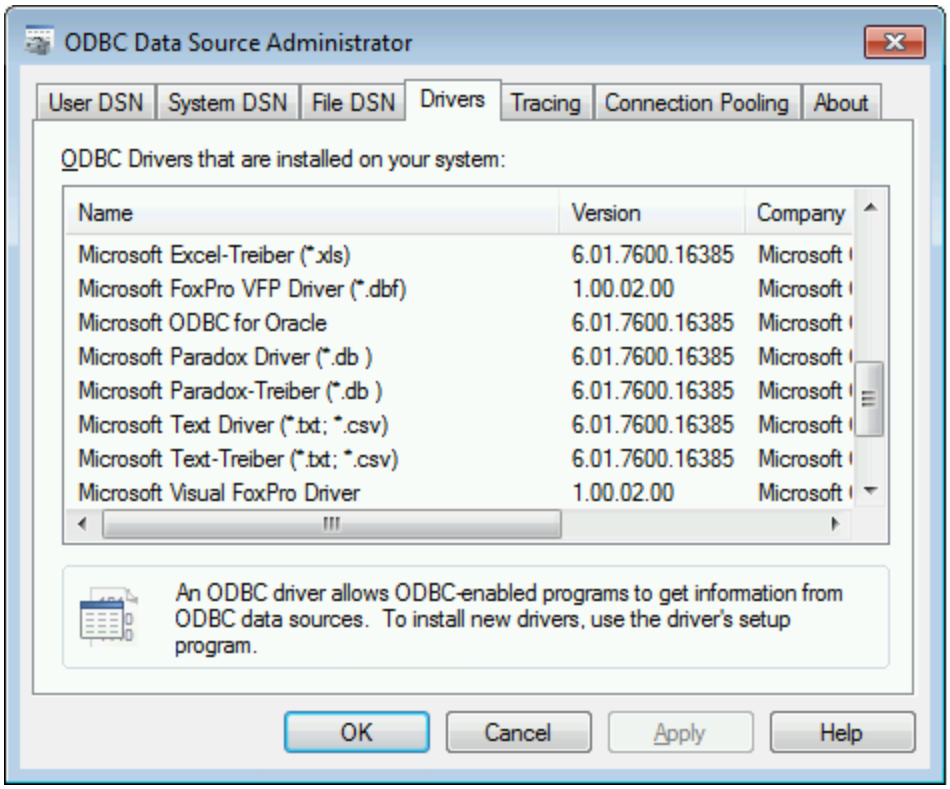
1. Starten Sie den Datenbank-Verbindungsassistenten 970.
2. Klicken Sie auf **ODBC-Verbindungen**.
3. Wählen Sie **Connection String erstellen-**
4. Kopieren Sie den Connection String in das entsprechende Feld ein und klicken Sie auf **Verbinden**.

19.1.5.1 Verfügbare ODBC-Treiber

Im ODBC-Datenquellen-Administrator können Sie die auf Ihrem Betriebssystem verfügbaren ODBC-Treiber anzeigen. Sie können den ODBC-Datenquellen-Administrator (**Odbcad32.exe**) über die Windows-Systemsteuerung unter **Verwaltung** aufrufen. Auf 64-Bit-Betriebssystemen gibt es zwei Versionen dieser ausführbaren Datei:

- Die 32-Bit-Version der Datei **Odbcad32.exe** befindet sich im Verzeichnis **C:\Windows\SysWoW64** (wenn **C:** Ihr Systemlaufwerk ist).
- Die 64-Bit-Version der Datei **Odbcad32.exe** befindet sich im Verzeichnis **C:\Windows\System32**.

Die installierten 32-Bit-Datenbanktreiber sind in der 32-Bit-Version des ODBC-Datenquellen-Administrators zu sehen, während die 64-Bit-Treiber in der 64-Bit-Version angezeigt werden. Vergewissern Sie sich daher, dass Sie die richtige Version des ODBC-Datenquellen-Administrators geöffnet haben, wenn Sie die Datenbanktreiber überprüfen.



ODBC-Datenquellen-Administrator

Wenn der Treiber für die gewünschte Datenbank in der Liste nicht vorhanden ist oder wenn Sie eine anderen Treiber hinzufügen möchten, müssen Sie diesen von der Webseite des Datenbankanbieters herunterladen (siehe Übersicht über Datenbanktreiber 972). Sobald der ODBC-Treiber auf Ihrem System verfügbar ist, können Sie damit ODBC-Verbindungen herstellen (siehe Einrichten der ODBC-Verbindung 989).

19.1.6 JDBC-Verbindung

JDBC (Java Database Connectivity) ist eine Datenbankzugriffsschnittstelle, die Teil der Java-Software-Plattform von Oracle ist. JDBC-Verbindungen beanspruchen im Allgemeinen mehr Ressourcen als ODBC-Verbindungen, bieten aber Funktionen, die über ODBC nicht zur Verfügung stehen.

Voraussetzungen

- JRE (Java Runtime Environment) oder Java Development Kit (JDK) muss installiert sein. Dabei muss es sich entweder um Oracle JDK oder einen Open Source Build wie Oracle OpenJDK handeln.XMLSpy

 © 2016-2022 Altova GmbH

ermittelt den Pfad zur Java Virtual Machine (JVM) anhand der folgenden Ordner und zwar in folgender Reihenfolge: a) anhand des benutzerdefinierten JVM-Pfads, den Sie eventuell in den **Applikationsoptionen** definiert haben; b) anhand des JVM-Pfads in der Windows Registry; c) anhand der `JAVA_HOME`-Umgebungsvariablen.

- Stellen Sie sicher, dass die Plattform von XMLSpy (32-Bit, 64-Bit) mit der des JRE/JDK übereinstimmt.
- Die JDBC-Treiber des Datenbankanbieters müssen installiert sein. Dabei kann es sich um JDBC-Treiber, die im Rahmen der Datenbankclient-Installation installiert wurden oder um separat heruntergeladene JDBC-Bibliotheken (.jar-Dateien) (falls verfügbar und von der Datenbank unterstützt) handeln, siehe auch Beispiele für Datenbankverbindungen 1003.
- Die `CLASSPATH`-Umgebungsvariable muss den Pfad zum JDBC-Treiber auf Ihrem Windows-Betriebssystem enthalten. Diese Variable wird unter Umständen bei der Installation einiger Datenbank Clients automatisch konfiguriert, siehe auch Konfigurieren des CLASSPATH 995.

Verbinden mit SQL Server über JDBC mit Windows-Anmeldeinformationen

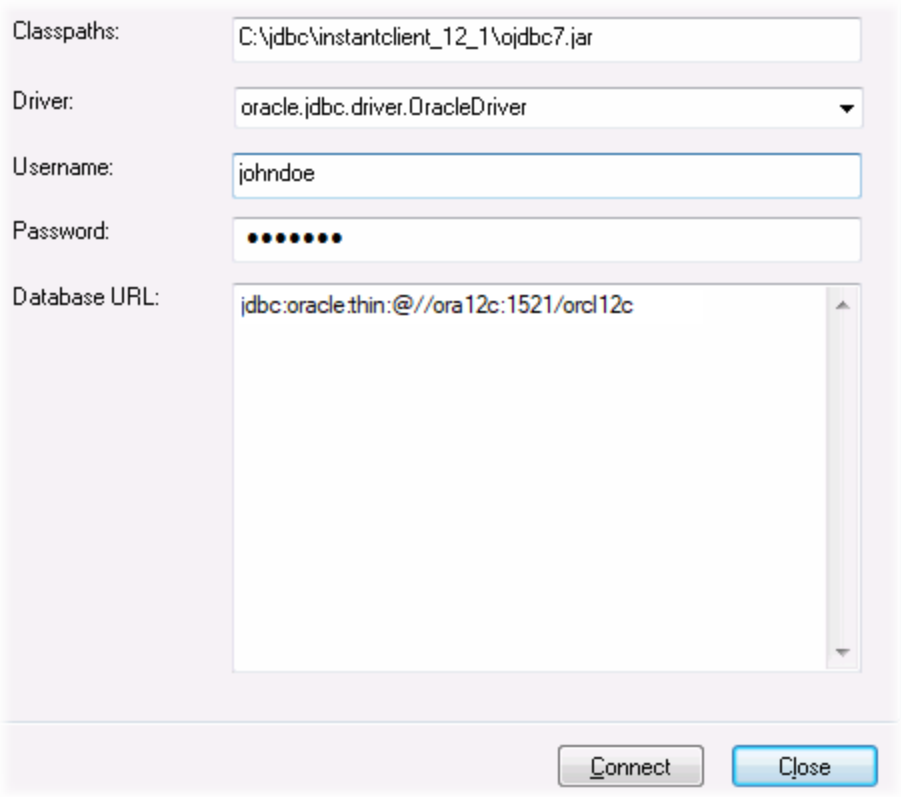
Wenn Sie über JDBC mit Windows-Anmeldeinformationen (integrierte Sicherheit) eine Verbindung zu SQL Server herstellen, beachten Sie die folgenden Punkte:

- Die im JDBC-Treiberpaket enthaltene Datei **sqljdbc_auth.dll** muss in ein Verzeichnis kopiert werden, das sich in der System PATH-Umgebungsvariablen befindet. Es gibt zwei solche Dateien, eine für die x86- und eine für die x64-Plattform. Vergewissern Sie sich, dass Sie die Ihrer JDK-Plattform entsprechende zum PATH hinzufügen.
- Der JDBC Connection String muss die Eigenschaft **`integratedSecurity=true`** enthalten.

Nähere Informationen dazu finden Sie in der *Microsoft-Dokumentation zu JDBC-Treibern für SQL Server* unter https://docs.microsoft.com/de-de/sql/connect/jdbc/building-the-connection-url?view=sql-server-ver15.

Einrichten einer JDBC-Verbindung

1. Starten Sie den Datenbank-Verbindungsassistenten 970.
2. Klicken Sie auf **JDBC-Verbindungen**.
3. Geben Sie optional in das Textfeld "Classpaths" eine durch Semikola getrennte Liste von .jar-Dateipfaden ein. Die hier eingegebenen .jar-Bibliotheken werden zusätzlich zu den bereits in der Umgebungsvariablen `CLASSPATH` definierten in die Umgebung geladen. Nachdem Sie Ihre Eingaben ins Textfeld "Classpaths" beendet haben, werden alle in den .jar-Quellbibliotheken gefundenen JDBC-Treiber automatisch zur Liste "Treiber" (siehe nächster Schritt) hinzugefügt.



4. Wählen Sie neben "Treiber" einen JDBC-Treiber aus der Liste aus oder geben Sie einen Java-Klassennamen ein. Beachten Sie, dass diese Liste alle über die Umgebungsvariable CLASSPATH konfigurierten JDBC-Treiber (siehe Konfigurieren des CLASSPATH 995) sowie die im Textfeld "Classpaths" gefundenen JDBC-Treiber enthält.

   Die in der CLASSPATH-Variablen definierten JDBC-Treiberpfade sowie alle direkt in das Datenbankverbindungsdialogfeld eingegebenen Pfade zu .jar-Dateien werden alle der Java Virtual Machine (JVM) zur Verfügung gestellt. Die JVM entscheidet anschließend, welche Treiber zur Herstellung einer Verbindung verwendet werden sollen. Es wird empfohlen, die in die JVM geladenen Java-Klassen im Auge zu behalten, damit es zu keinen potenziellen JDBC-Treiberkonflikten und unerwarteten Ergebnissen bei der Herstellung der Datenbankverbindung kommt.

5. Geben Sie den Benutzernamen und das Passwort für die Datenbank in die entsprechenden Felder ein.
6. Geben Sie im Textfeld "Datenbank-URL" die JDBC Connection-URL (String) im datenbanktypspezifischen Format ein. In der folgenden Tabelle sehen Sie die Syntaxvorgaben für die JDBC Connection-URLs (Strings) für gebräuchliche Datenbanktypen.

Datenbank	JDBC-Verbindungs-URL
Firebird	`jdbc:firebirdsql://<host>[:<port>]/<database path or alias>`
IBM DB2	`jdbc:db2://hostName:port/databaseName`

© 2016-2022 Altova GmbH

IBM DB2 for i	jdbc:as400://**[host]**
IBM Informix	jdbc:informix-sqli://**hostName**:**port**/**databaseName**:INFORMIXSERVER=**myserver**
MariaDB	jdbc:mariadb://**hostName**:**port**/**databaseName**
Microsoft SQL Server	jdbc:sqlserver://**hostName**:**port**;**databaseName**=name
MySQL	jdbc:mysql://**hostName**:**port**/**databaseName**
Oracle	jdbc:oracle:thin:@**hostName**:**port**:**SID** jdbc:oracle:thin:@//**hostName**:**port**/**service**
Oracle XML DB	jdbc:oracle:oci:@//**hostName**:**port**:**service**
PostgreSQL	jdbc:datadirect:openedge://**host**:**port**;databaseName=**db_name**
Progress OpenEdge	jdbc:datadirect:openedge://**host**:**port**;databaseName=**db_name**
Sybase	jdbc:sybase:Tds:**hostName**:**port**/**databaseName**
Teradata	jdbc:teradata://**databaseServerName**

Anmerkung: Bei den oben aufgelisteten Formaten sind auch Syntaxvarianten möglich (die Datenbank-URL kann eventuell ohne Port oder einschließlich Benutzernamen und Datenbank-Passwort angegeben werden). Nähere Informationen dazu finden Sie in der Dokumentation des jeweiligen Datenbankanbieters.

7. Klicken Sie auf **Verbinden**.

19.1.6.1 Konfigurieren des CLASSPATH

Mit Hilfe der CLASSPATH-Umgebungsvariablen findet das Java Runtime Environment (JRE) bzw. der Java Development Kit (JDK) Java-Klassen und andere Ressourcendateien auf Ihrem Betriebssystem. Bei Herstellung einer Datenbankverbindung über JDBC muss diese Variable so konfiguriert werden, dass sie den Pfad zum JDBC-Treiber auf Ihrem Betriebssystem und in einigen Fällen den Pfad zur zusätzlichen datenbanktypspezifischen Bibliotheken enthält.

In der folgenden Tabelle sind typische Beispieldateipfade aufgelistet, die in der CLASSPATH-Variablen enthalten sein müssen. Sie müssen diese Informationen eventuell anhand des JDBC-Treibers auf Ihrem System, des JDBC-Treibernamens sowie der JRE/JDK-Version auf Ihrem Betriebssystem anpassen. Um Verbindungsprobleme zu vermeiden, lesen Sie die Installationsanleitung zu dem auf Ihrem Betriebssystem installierten JDBC-Treiber und führen Sie alle vor oder nach der Installation erforderlichen Schritte durch.

Datenbank	**CLASSPATH-Beispieleinträge**
Firebird	C:\Programme\Firebird\Jaybird-2.2.8-JDK_1.8\jaybird-full-2.2.8.jar

IBM DB2	`C:\Programme (x86)\IBM\SQLLIB\java\db2jcc.jar;C:\Programme (x86)\IBM\SQLLIB\java\db2jcc_license_cu.jar;`
IBM DB2 for i	`C:\jt400\jt400.jar;`
IBM Informix	`C:\Informix_JDBC_Driver\lib\ifxjdbc.jar;`
Microsoft SQL Server	`C:\Programme\Microsoft JDBC Driver 4.0 for SQL Server\sqljdbc_4.0\enu\sqljdbc.jar`
MariaDB	**`<installation directory>`**`\mariadb-java-client-2.2.0.jar`
MySQL	`mysql-connector-java-`*`version`*`-bin.jar;`
Oracle	**`ORACLE_HOME`**`\jdbc\lib\ojdbc6.jar;`
Oracle (mit XML DB)	**`ORACLE_HOME`**`\jdbc\lib\ojdbc6.jar;`**`ORACLE_HOME`**`\LIB\xmlparserv2.jar;` **`ORACLE_HOME`**`\RDBMS\jlib\xdb.jar;`
PostgreSQL	**`<installation directory>`**`\postgresql.jar`
Progress OpenEdge	`%DLC%\java\openedge.jar;%DLC%\java\pool.jar;` Anmerkung: Angenommen, Progress OpenEdge SDK ist auf dem Rechner installiert, so ist `%DLC%` das Verzeichnis, in dem OpenEdge installiert ist.
Sybase	`C:\sybase\jConnect-7_0\classes\jconn4.jar`
Teradata	**`<installation directory>`**`\tdgssconfig.jar;`**`<installation directory>`**`\terajdbc4.jar`

- Wenn Sie die CLASSPATH-Variable ändern, kann sich dies auf das Verhalten von Java-Applikationen auf Ihrem Rechner auswirken. Lesen Sie dazu die Java-Dokumentation.
- Umgebungsvariablen können benutzer- oder systemspezifisch sein. Um System-Umgebungsvariablen zu ändern, benötigen Sie Administratorrechte auf Ihrem Betriebssystem.
- Nachdem Sie die Umgebungsvariable geändert haben, starten Sie alle laufenden Programme neu, damit die Änderungen wirksam werden. Alternativ dazu können Sie sich am Betriebssystem auch ab- und wieder anmelden oder dieses neu starten.

So konfigurieren Sie den CLASSPATH unter Windows 7:

1. Öffnen Sie das **Startmenü** und klicken Sie mit der rechten Maustaste auf **Computer**.
2. Klicken Sie auf **Eigenschaften**.
3. Klicken Sie auf **Erweiterte Systemeinstellungen**.
4. Klicken Sie auf dem Register **Erweitert** auf **Umgebungsvariablen.**
5. Gehen Sie unter Benutzer- oder System-Umgebungsvariablen zur CLASSPATH-Variablen und klicken Sie anschließend auf **Bearbeiten**. Wenn die CLASSPATH-Variable nicht vorhanden ist, klicken Sie auf **Neu**, um sie zu erstellen.
6. Bearbeiten Sie den Wert der Variablen, damit sie den Pfad enthält, auf dem sich auf Ihrem Betriebssystem der JDBC-Treiber befindet. Verwenden Sie das Semikolon (;), um den JDBC-Treiberpfad von anderen in der CLASSPATH-Variablen bereits vorhandenen Pfaden zu trennen.

© 2016-2022 Altova GmbH

So konfigurieren Sie den CLASSPATH unter Windows 10:

1. Drücken Sie die Windows-Taste und beginnen Sie mit der Eingabe von "Umgebungsvariablen".
2. Klicken Sie auf den Vorschlag **Systemumgebungsvariablen bearbeiten**.
3. Klicken Sie auf **Umgebungsvariablen.**
4. Gehen Sie unter Benutzer- oder System-Umgebungsvariablen zur CLASSPATH-Variablen und klicken Sie anschließend auf **Bearbeiten**. Wenn die CLASSPATH-Variable nicht vorhanden ist, klicken Sie auf **Neu**, um sie zu erstellen.
5. Bearbeiten Sie den Wert der Variablen, damit sie den Pfad enthält, auf dem sich auf Ihrem Betriebssystem der JDBC-Treiber befindet. Verwenden Sie das Semikolon (;), um den JDBC-Treiberpfad von anderen in der CLASSPATH-Variablen bereits vorhandenen Pfaden zu trennen.

19.1.7 PostgreSQL-Verbindung

Verbindungen zu PostgreSQL-Datenbanken können entweder als native Verbindung oder über ODBC, JDBC und andere Treiber hergestellt werden. Der Vorteil einer nativen Verbindung ist, dass keine Treiber auf Ihrem System installiert werden müssen.

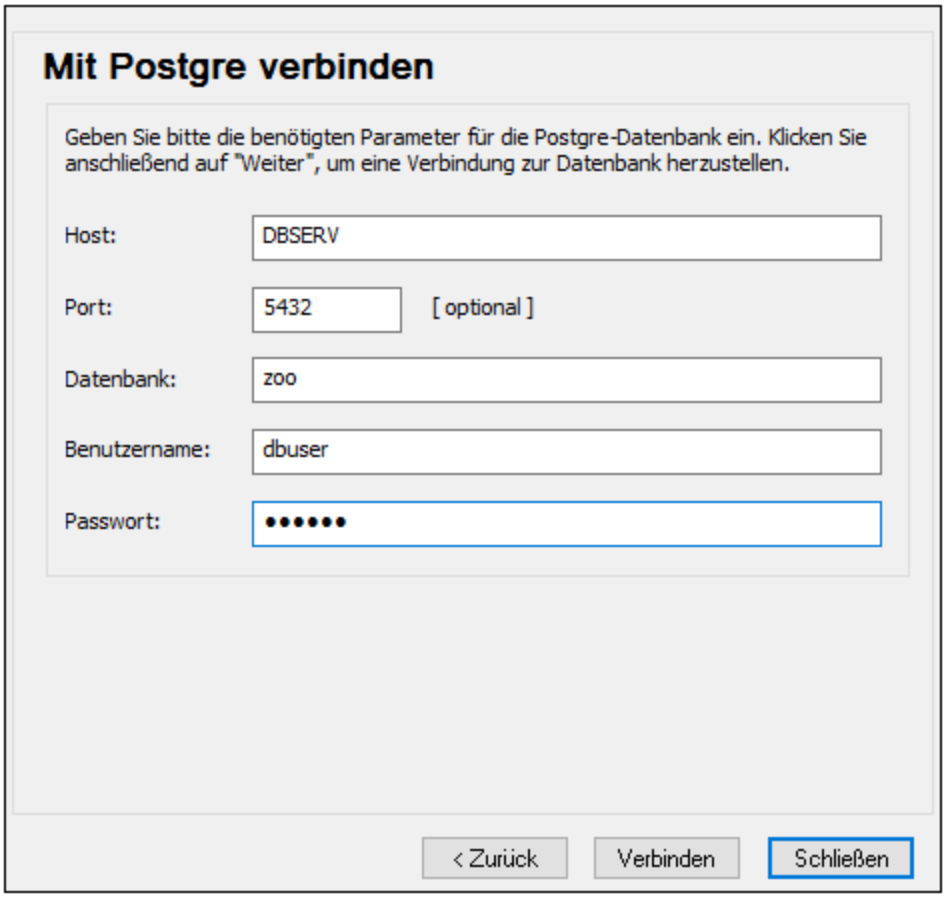
Wenn Sie die Verbindung lieber über einen nicht nativen Treiber herstellen möchten, lesen Sie in den folgenden Kapiteln nach:

- Einrichten einer JDBC-Verbindung 992
- Verbinden mit PostgreSQL (ODBC) 1043

Wenn Sie eine native Verbindung zu PostgreSQL herstellen möchten, befolgen Sie die unten stehenden Anleitung. Um fortzufahren, benötigen Sie die folgenden Informationen: Host-Name, Port, Datenbankname, Benutzername und Passwort.

So richten Sie eine native PostgreSQL-Verbindung ein:

1. Starten Sie den Datenbank-Verbindungsassistenten 970.
2. Klicken Sie auf **PostgreSQL-Verbindungen**.
3. Geben Sie den Host (*localhost*, wenn PostgreSQL auf demselben Rechner ausgeführt wird), Port (normalerweise 5432, dies ist optional), den Datenbanknamen, den Benutzernamen und das Passwort in die entsprechenden Felder ein.



4. Klicken Sie auf **Verbinden**.

Wenn sich der PostgreSQL-Datenbankserver auf einem anderen Rechner befindet, beachten Sie die folgenden Punkte:

- Der PostgreSQL-Datenbankserver muss so konfiguriert sein, dass er Verbindungen von Clients zulässt. Insbesondere muss die Datei **pg_hba.conf** so konfiguriert werden, dass sie nicht lokale Verbindungen zulässt. Außerdem muss die Datei **postgresql.conf** so konfiguriert werden, dass sie eine listen-Verbindung zu bestimmten ID-Adressen und einem bestimmten Port hat. Nähere Informationen dazu finden Sie in der Dokumentation zu PostgreSQL (https://www.postgresql.org/docs/9.5/static/client-authentication-problems.html).
- Der Server-Rechner muss so konfiguriert sein, dass er Verbindungen am angegebenen Port (normalerweise 5432) durch die Firewall zulässt. So müssen Sie z.B. eventuell auf einem Windows-Datenbankserver eine Regel erstellen, damit Verbindungen über Port 5432 durch die Firewall zugelassen werden. Wählen Sie dazu **Systemsteuerung > Windows Firewall > Erweiterte Einstellungen > Eingehende Regeln**.

19.1.8 SQLite-Verbindung

SQLite ist ein dateibasierter, eigenständiger Datenbanktyp, der dadurch ideal für Szenarien geeignet ist, in denen Portabilität und einfache Konfiguration wichtig sind. Da SQLite-Datenbanken von XMLSpy nativ

© 2016-2022 Altova GmbH

unterstützt werden, müssen zur Herstellung einer Verbindung mit SQLite-Datenbanken keine Treiber installiert werden.

Anmerkungen zur Unterstützung von SQLite-Datenbanken

- Für SQLite-Datenbanken kann auf Linux-Systemen kein Timeout für eine Anweisungsausführung definiert werden.
- Die Volltextsuche in Tabellen wird nicht unterstützt.
- Bei SQLite können in jeder Zeile einer Tabelle Werte unterschiedliche Datentypen verwendet werden. Alle verarbeiteten Werte müssen mit dem deklarierten Spaltentyp kompatibel sein; daher können unerwartete Werte abgerufen werden und Laufzeitfehler auftreten, wenn der Wert in der Zeile einer SQLite-Datenbank nicht mit dem deklarierten Spaltentyp übereinstimmt.

Achtung

Es wird empfohlen, Tabellen mit dem Schlüsselwort `STRICT` zu erstellen, um ein besser vorhersehbares Verhalten Ihrer Daten sicherzustellen. Andernfalls können Daten eventuell nicht korrekt gelesen oder geschrieben werden, wenn eine Spalte Werte unterschiedlichen Typs enthält. Nähere Informationen zu STRICT-Tabellen finden Sie in der SQLite-Dokumentation.

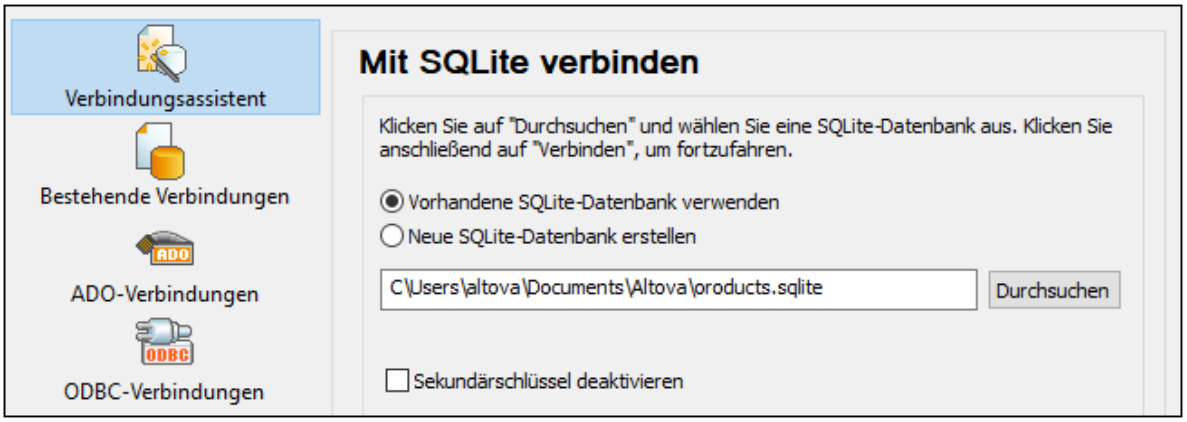
19.1.8.1 Herstellen einer Verbindung zu einer bestehenden SQLite-Datenbank

Sie können entweder mit Hilfe des Verbindungsassistenten oder direkt über den Windows Explorer mit dem Befehl **Öffnen mit...** eine Verbindung zu einer bestehenden SQLite-Datenbank herstellen.

Mit dem Verbindungsassistenten

So stellen Sie eine Verbindung zu einer vorhandenen SQLite-Datenbank her:

1. Starten Sie den Datenbankverbindungsassistenten (siehe Starten des Datenbankverbindungsassistenten 970).
2. Wählen Sie **SQLite** aus und klicken Sie auf **Weiter**.



3. Wählen Sie "**Vorhandene SQLite-Datenbank verwenden**" und navigieren Sie anschließend zur SQLite-Datenbankdatei oder geben Sie den (relativen oder absoluten) Pfad zur Datenbank ein. Die Schaltfläche **Verbinden** wird aktiv, sobald Sie den Pfad zu einer SQLite-Datenbankdatei eingegeben haben.
4. Aktivieren Sie optional das Kontrollkästchen **Sekundärschlüssel deaktivieren**, siehe Sekundärschlüssel-Constraints[1001].
5. Klicken Sie auf **Verbinden.**

Über Windows Explorer

Sie können eine SQLite-Datenbank auch direkt über Windows Explorer öffnen. Gehen Sie dazu folgendermaßen vor:

1. Klicken Sie in Windows Explorer mit der rechten Maustaste auf eine vorhandene Datenbankdatei und wählen Sie im Kontextmenü den Befehl **Öffnen mit**.
2. Wählen Sie aus der Liste der Vorschläge **DatabaseSpy** aus. Falls dieses Programm nicht vorgeschlagen wird, klicken Sie auf **Andere App auswählen** und navigieren Sie im Installationsverzeichnis zur ausführbaren DatabaseSpy-Datei.
3. Wenn der Datenbankverbindungsassistent angezeigt wird, klicken Sie auf **Schließen**, damit dieser nicht aufgerufen wird.

Sie können die Datenbankdatei auch mit der Maus in DatabaseSpy ziehen, falls das Programm bereits geöffnet ist.

Anmerkungen:

- Sie haben die Möglichkeit, DatabaseSpy über **Extras | Optionen** so zu konfigurieren, dass der Datenbankverbindungsassistent beim Start der Applikation nicht aufgerufen wird. Deaktivieren Sie dazu in der Einstellungsgruppe **Allgemein** das Kontrollkästchen **Dialogfeld "Datenbankverbindung erstellen" anzeigen**.
- Wenn DatabaseSpy offen ist und bereits eine Datenquelle zur selben Datenbankdatei vorhanden ist, wird diese wiederverwendet. D.h. Datenquellen zur selben Datenbank werden nicht doppelt erstellt. Dies gilt sowohl für das Öffnen einer Datenbankdatei mit **Öffnen mit** als auch für das Öffnen über Drag-and-Drop.

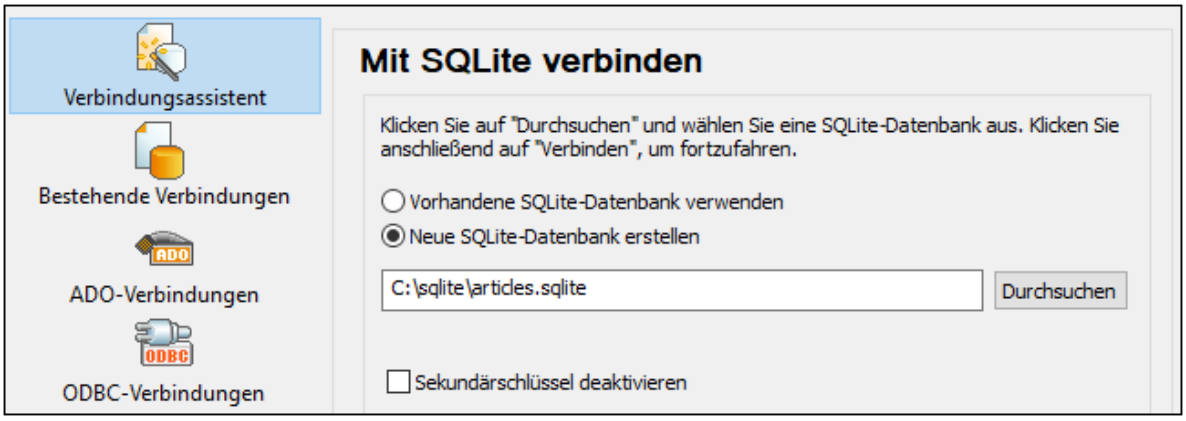
19.1.8.2 Erstellen einer neuen SQLite-Datenbank

Anstatt eine Verbindung zu einer bestehenden Datenbankdatei herzustellen, können Sie eine neue SQLite-Datenbankdatei erstellen und zu dieser eine Verbindung herstellen. Die von XMLSpy erstellte Datenbankdatei ist leer; erstellen Sie mit Hilfe von Abfragen oder Skripts die gewünschte Datenbankstruktur und befüllen Sie diese mit Daten.

So erstellen Sie eine neue SQLite-Datenbank:

1. Starten Sie den Datenbankverbindungsassistenten (siehe Starten des Datenbankverbindungsassistenten[970]).
2. Wählen Sie **SQLite** aus und klicken Sie auf **Weiter**.

© 2016-2022 Altova GmbH



3. Wählen Sie die Option **Neue SQLite-Datenbank erstellen** und geben Sie anschließend den gewünschten (relativen oder absoluten) Pfad ein (z.B., **c:\users\public\products.sqlite**). Alternativ dazu können Sie auf **Durchsuchen** klicken, um einen Ordner auszuwählen, den Namen der Datenbankdatei in das Textfeld "Dateiname" eingeben (z.B.**products.sqlite**) und auf **Speichern** klicken.

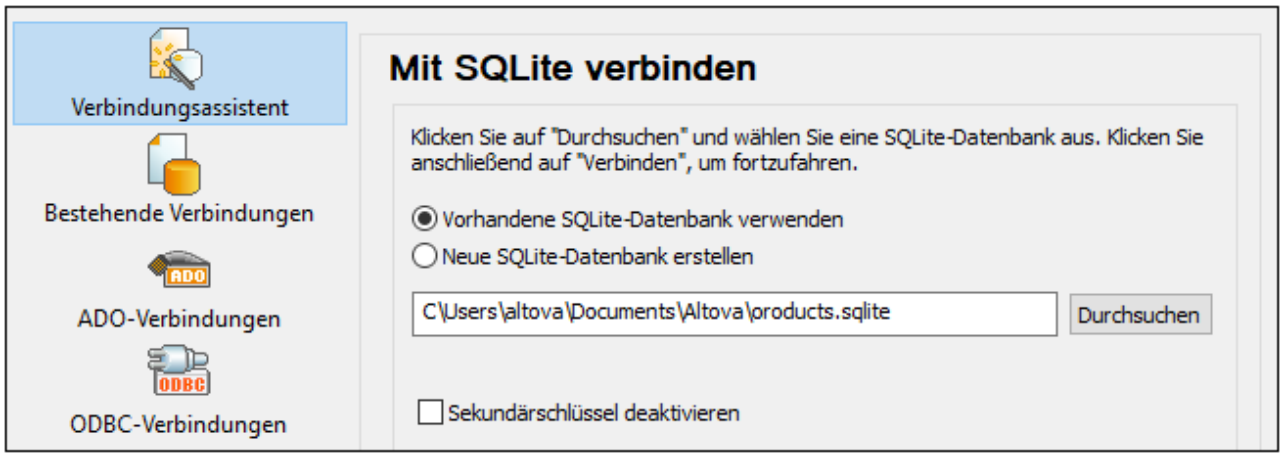
   Stellen Sie sicher, dass Sie für den Ordner, in dem die SQLite-Datenbankdatei erstellt werden soll, Schreibrechte haben.

4. Aktivieren Sie optional das Kontrollkästchen **Sekundärschlüssel deaktivieren**, siehe Sekundärschlüssel-Constraints[998].
5. Klicken Sie auf **Verbinden**.

19.1.8.3 Sekundärschlüssel-Constraints

Wenn Sie von XMLSpy aus eine Verbindung zu einer vorhandenen SQLite-Datenbank herstellen oder wenn Sie eine neue Verbindung erstellen sind Sekundärschlüssel-Constraints standardmäßig aktiviert. Mit Hilfe von Sekundärschlüssel-Constraints können Sie die Integrität der Daten in Ihrer Datenbank sicherstellen. So ist es z.B. nicht möglich, einen Datensatz aus einer Tabelle zu löschen, wenn er Abhängigkeiten in einer anderen Tabelle hat, wenn Sekundärschlüssel aktiviert sind.

In einigen Fällen muss dieses Verhalten vorübergehend außer Kraft gesetzt werden und Sie müssen die Sekundärschlüssel deaktivieren, z.B. um mehrere Datenzeilen einfügen oder aktualisieren zu können, ohne Datenvalidierungsfehler zu erhalten. Um Sekundärschlüssel vor dem Herstellen der Verbindung zur SQLite-Datenbank explizit zu deaktivieren, aktivieren Sie im Datenbankverbindungsassistenten die Option **Sekundärschlüssel deaktivieren**.



Assistentenseite "Mit SQLite verbinden"

Wenn Sekundärschlüssel deaktiviert sind, können Sie Operationen an Daten durchführen, die andernfalls aufgrund von Validierungsfehlern nicht möglich wären. Dabei besteht allerdings das Risiko, dass fehlerhafte Daten in die Datenbank eingeschleust, oder "verwaiste" Zeilen erstellt werden (Ein Beispiel für eine solche Zeile wäre eine Adresse in der Tabelle "addresses", die mit keiner Person in der Tabelle "person" verknüpft ist, da die Person gelöscht wurde, die damit verknüpfte Adresse aber nicht.)

19.1.9 Globale Ressourcen

Nachdem Sie eine Datenbank als globale Ressource erstellt haben, werden ihre Verbindungsinformationen gespeichert und können in allen auf Ihrem Rechner installierten Altova-Produkten verwendet werden.

Erstellen einer Datenbank als globale Ressource

Um eine Datenbank als globale Ressource zu erstellen, gehen Sie folgendermaßen vor:

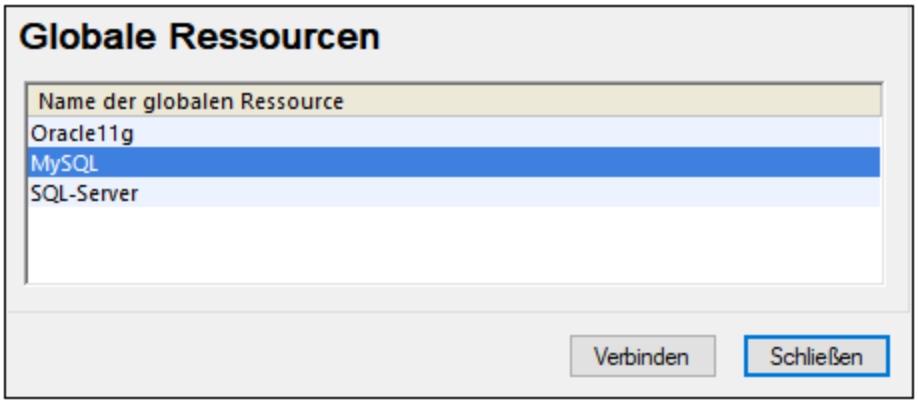
1. Wählen Sie im Menü **Extras** von XMLSpy den Befehl **Globale Ressourcen**.
2. Klicken Sie auf **Hinzufügen** und anschließend auf Datenbank.
3. Geben Sie in das Feld *Ressourcen-Alias* einen Namen für die globale Ressource ein.
4. Klicken Sie auf **Datenbank auswählen**. Daraufhin wird der Verbindungsassistent 970 angezeigt.
5. Fügen Sie mit Hilfe des Verbindungsassistenten, wie oben beschrieben, eine Datenbankverbindung hinzu.

Verwenden einer Datenbank aus den globalen Ressourcen

Um eine als globale Ressource erstellte Datenbank (*siehe oben*) zu verwenden, gehen Sie folgendermaßen vor:

1. Starten Sie den Verbindungsassistenten, wie oben beschrieben.
2. Wählen Sie Globale Ressourcen aus. Alle als globale Ressourcen erstellten Datenbanken werden im Fenster "Globale Ressourcen" nach ihrem Namen aufgelistet (*siehe Abbildung unten*).

© 2016-2022 Altova GmbH



3. Wählen Sie die gewünschte globale Ressource aus. Tipp: Wenn Sie den Mauszeiger über eine globale Ressource in der Liste platzieren, werden Informationen über die Datenbank angezeigt.

19.1.10 Beispiele für Datenbankverbindungen

In diesem Abschnitt sind Beispiele dafür beschrieben, wie Sie von XMLSpy aus über ADO, ODBC oder JDBC eine Verbindung zu einer Datenbank herstellen. Die Beispiele für ADO.NET-Verbindungen sind separat aufgelistet, siehe Beispiele für ADO.NET Connection Strings[987]. Anleitungen dazu, wie Sie eine native Verbindung zu PostgreSQL und SQLite herstellen, finden Sie unter Einrichten einer PostgreSQL-Verbindung[997] bzw. Einrichten einer SQLite-Verbindung[998].

Beachten Sie die folgenden Punkte:

- Wenn Ihre Windows-Konfiguration, Ihre Netzwerkumgebung und Ihre Datenbank Client- oder Server-Software nicht genau der in den Beispielen beschriebenen Konfiguration entsprechen, weicht die Vorgangsweise eventuell etwas von der in den Beispielen beschriebenen ab.
- Bei den meisten Datenbanktypen kann die Verbindung über unterschiedliche Datenzugriffstechnologien (ADO, ADO.NET, ODBC, JDBC) oder Treiber hergestellt werden. Das Verhalten der Datenbankverbindung, die verfügbaren Funktionalitäten und Einschränkungen hängen vom ausgewählten Treiber, (gegebenenfalls) der Datenbank Client-Software und eventuellen zusätzlich außerhalb von XMLSpy konfigurierten Verbindungsparametern ab.

19.1.10.1 Firebird (JDBC)

Dieses Kapitel enthält eine Beispielanleitung für das Herstellen einer Verbindung zu einem Firebird Datenbankserver mittels JDBC.

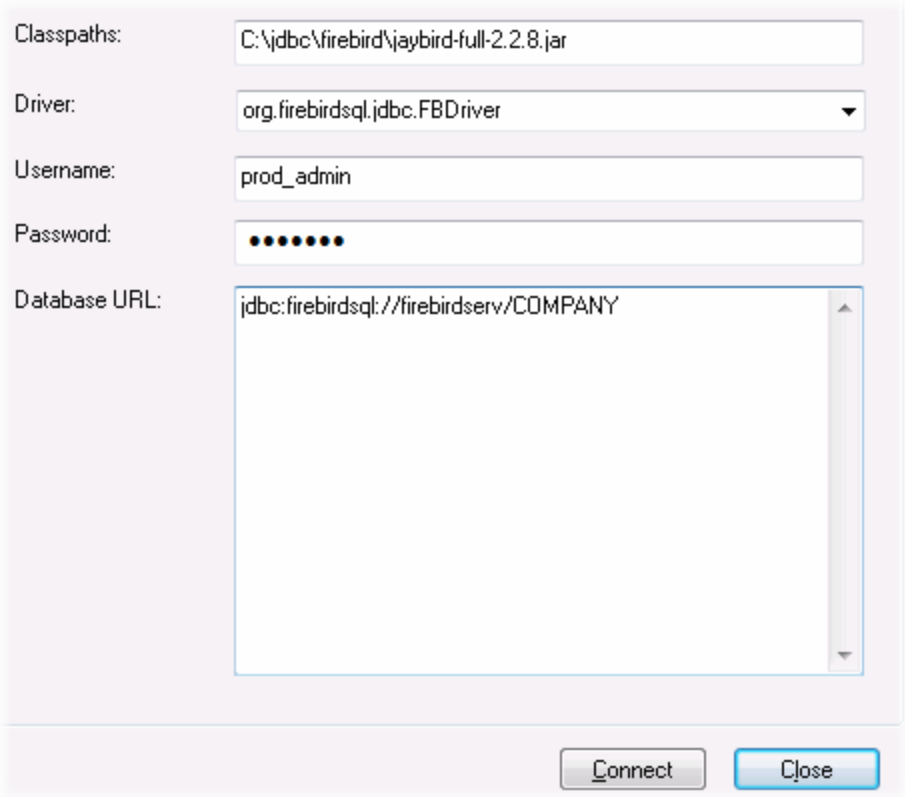
Voraussetzungen:

- Auf Ihrem Betriebssystem muss JRE (Java Runtime Enviroment) oder Java Development KIT (JDK) installiert sein. Dabei muss es sich entweder um Oracle JDK oder einen Open Source Build wie Oracle OpenJDK handeln.XMLSpy ermittelt den Pfad zur Java Virtual Machine (JVM) anhand der folgenden Ordner und zwar in folgender Reihenfolge: a) anhand des benutzerdefinierten JVM-Pfads, den Sie

eventuell in den **Applikationsoptionen** definiert haben; b) anhand des JVM-Pfads in der Windows Registry; c) anhand der `JAVA_HOME`-Umgebungsvariablen.
- Stellen Sie sicher, dass die Plattform von XMLSpy (32-Bit, 64-Bit) mit der des JRE/JDK übereinstimmt.
- Der Firebird JDBC-Treiber muss auf Ihrem Betriebssystem verfügbar sein (Er hat die Form einer .jar-Datei, die eine Verbindung zur Datenbank herstellt). Der Treiber kann von der Firebird Website (https://www.firebirdsql.org/) heruntergeladen werden. In diesem Beispiel wird *Jaybird 2.2.8* verwendet.
- Sie haben die folgenden Datenbankverbindungsinformationen zur Verfügung: Host, Datenbankpfad oder Alias, Benutzername und Passwort.

So stellen Sie über JDBC eine Verbindung zu Firebird her:

1. Starten Sie den Datenbank-Verbindungsassistenten 970.
2. Klicken Sie auf **JDBC-Verbindungen**.
3. Geben Sie neben "Classpaths" den Pfad zur .jar-Datei , die die Verbindung zur Datenbank bereitstellt, ein. Falls nötig, können Sie auch eine durch Semikola getrennte Liste von .jar-Dateipfaden eingeben. Die benötigte .jar-Datei in diesem Beispiel befindet sich unter dem folgenden Pfad: **C:\jdbc\firebird\jaybird-full-2.2.8.jar**. Beachten Sie, dass Sie das Textfeld "Classpaths" leer lassen können, wenn Sie den/die .jar-Dateipfad(e) zur Umgebungsvariablen CLASSPATH des Betriebssystems hinzugefügt haben (siehe auch Konfigurieren des CLASSPATH 995).
4. Wählen Sie im Feld "Treiber" **org.firebirdsql.jdbc.FBDriver** aus. Beachten Sie, dass dieser Eintrag zur Verfügung steht, wenn entweder im Textfeld "Classpaths" oder in der Umgebungsvariablen `CLASSPATH` des Betriebssystems eine gültige .jar-Datei gefunden wird (siehe vorheriger Schritt).



5. Geben Sie den Benutzernamen und das Passwort in die entsprechenden Textfelder ein.
6. Geben Sie in das Textfeld "Datenbank-URL" den Connection String zum Datenbankserver ein, indem Sie die hervorgehobenen Werte durch die entsprechenden Werte für Ihren Datenbankserver ersetzen.

 © 2016-2022 Altova GmbH

```
jdbc:firebirdsql://<host>[:<port>]/<database path or alias>
```

7. Klicken Sie auf **Verbinden**.

19.1.10.2 Firebird (ODBC)

Dieses Kapitel enthält eine Beispielanleitung für das Herstellen einer Verbindung zu einer Firebird 2.5.4-Datenbank auf einem Linux Server.

Voraussetzungen:

- Der Firebird-Datenbankserver akzeptiert aufgrund seiner Konfiguration TCP/IP-Verbindungen von Clients.
- Der Firebird ODBC-Treiber muss auf Ihrem Betriebssystem installiert sein. In diesem Beispiel wird der Firebirg ODBC-Treiber Version 2.0.3.154 verwendet, der von der Firebird Website (https://www.firebirdsql.org/) heruntergeladen wurde.
- Der Firebird Client muss auf Ihrem Betriebssystem installiert sein. Beachten Sie, dass für den Firebird 2.5.4 Client kein eigenständiger Installer verfügbar ist; der Client ist Teil des Firebird Server-Installationspakets, welches Sie unter "Windows executable installer for full Superclassic/Classic or Superserver" von der Firebird Website (https://www.firebirdsql.org/) herunterladen können. Um nur die Client-Dateien zu installieren, wählen Sie im Zuge der Installation die Option "**Minimum client install - no server, no tools**" aus.

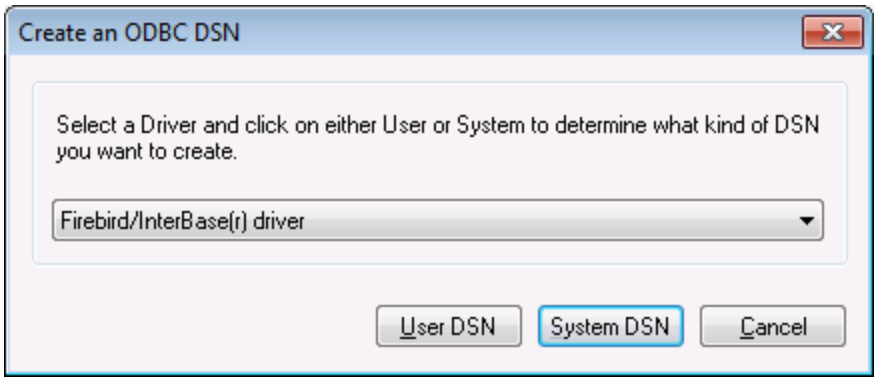
  Wichtig:

  - Die Plattform des Firebird ODBC-Treibers und Client (32-Bit oder 64-Bit) muss mit der von XMLSpy übereinstimmen.
  - Die Version des Firebird Client muss mit der des Firebird Servers, zu dem Sie die Verbindung herstellen, übereinstimmen.

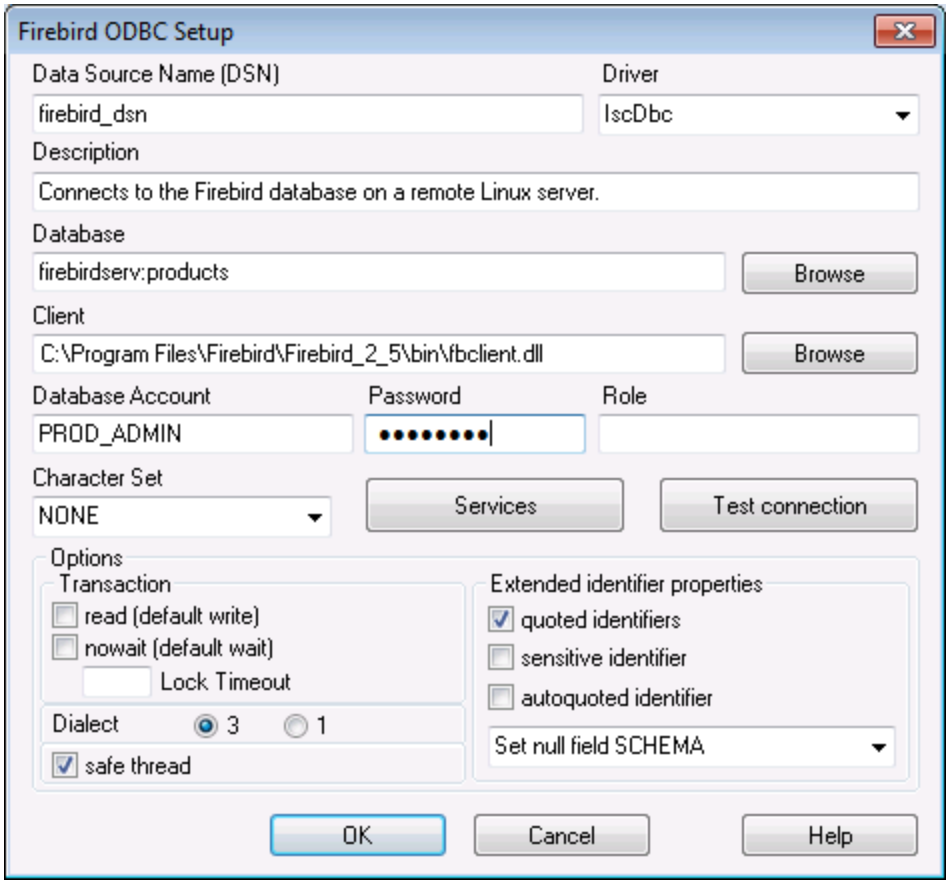
- Sie verfügen über die folgenden Datenbankverbindungsinformationen: Name oder IP-Adresse des Server Host, Datenbankpfad (oder Alias) auf dem Server, Benutzername und Passwort.

So stellen Sie über ODBC eine Verbindung zu Firebird her:

1. Starten Sie den Datenbank-Verbindungsassistenten 970.
2. Klicken Sie auf **ODBC-Verbindungen**.
3. Wählen Sie Benutzer-DSN (oder **System-DSN**, wenn Sie Administratorrechte haben) und klicken Sie anschließend auf **Hinzufügen** .



4. Wählen Sie den Firebird-Treiber aus und klicken Sie anschließend, je nachdem, was Sie im vorherigen Schritt ausgewählt haben, auf **Benutzer-DSN** oder **System-DSN**. Wenn der Firebird-Treiber in der Liste nicht zur Verfügung steht, stellen Sie sicher, dass er auf Ihrem Betriebssystem installiert ist (siehe auch Anzeigen der verfügbaren ODBC-Treiber 991).



5. Geben Sie die folgenden Datenbankverbindungsinformationen ein:

Data Source Name (DSN)	Geben Sie einen beschreibenden Namen für die zu erstellende Datenquelle ein.

 © 2016-2022 Altova GmbH

Database	Geben Sie den Namen oder die IP-Adresse des Server Host, gefolgt von einem Doppelpunkt, gefolgt vom Datenbank-Alias (oder Pfad) ein. In diesem Beispiel lautet der Host-Name `firebirdserv` und der Datenbank-Alias `products`. Geben Sie daher den folgenden String ein:
	
	`firebirdserv:products`
	
	Wenn Sie einen Datenbank-Alias verwenden, wird davon ausgegangen, dass der Datenbankadministrator den Alias *products* auf dem Server so konfiguriert hat, dass er auf die tatsächliche Firebird (.fdb) Datenbankdatei verweist (nähere Informationen dazu siehe Firebird-Dokumentation).
	
	Sie können anstelle des Host-Namens auch die Server IP-Adresse und anstelle eines Pfads einen Alias verwenden; daher ist jeder der folgenden Beispiel Connection-Strings gültig:
	
	`firebirdserver:/var/Firebird/databases/butterflies.fdb`
	`127.0.0.1:D:\Misc\Lenders.fdb`
	
	Wenn sich die Datenbank auf dem lokalen Windows-Rechner befindet, klicken Sie auf **Durchsuchen** und wählen Sie die Firebird (.fdb) Datenbank direkt aus.

Client	Geben Sie den Pfad zur Datei **fbclient.dll** ein. Standardmäßig ist dies das Unterverzeichnis `bin` des Firebird-Installationsverzeichnisses.
Database Account	Geben Sie den vom Datenbankadministrator bereitgestellten Datenbank-Benutzernamen ein (in diesem Beispiel `PROD_ADMIN`).
Password	Geben Sie das vom Datenbankadministrator bereitgestellte Datenbank-Passwort ein

1. Klicken Sie auf **OK**.

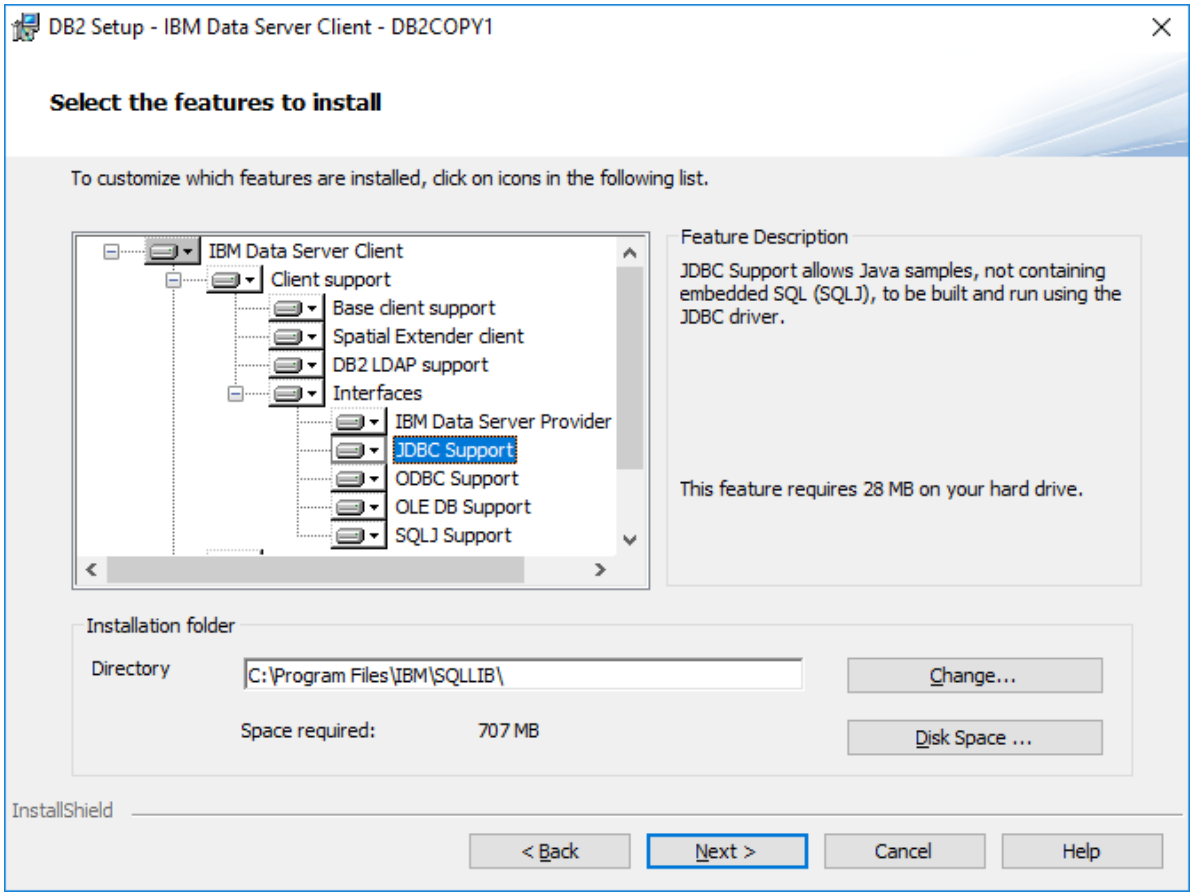
19.1.10.3 IBM DB2 (JDBC)

Dieses Kapitel enthält eine Beispielanleitung für das Herstellen einer Verbindung zu einem **IBM DB2** Datenbankserver über JDBC.

Voraussetzungen:

- Auf Ihrem Betriebssystem muss JRE (Java Runtime Enviroment) oder Java Development KIT (JDK) installiert sein. Dabei muss es sich entweder um Oracle JDK oder einen Open Source Build wie Oracle OpenJDK handeln.XMLSpy ermittelt den Pfad zur Java Virtual Machine (JVM) anhand der folgenden Ordner und zwar in folgender Reihenfolge: a) anhand des benutzerdefinierten JVM-Pfads, den Sie eventuell in den **Applikationsoptionen** definiert haben; b) anhand des JVM-Pfads in der Windows Registry; c) anhand der `JAVA_HOME`-Umgebungsvariablen.

- Stellen Sie sicher, dass die Plattform von XMLSpy (32-Bit, 64-Bit) mit der des JRE/JDK übereinstimmt. In diesem Beispiel wird Oracle OpenJDK 11.0 64-Bit verwendet, folglich wird auch die 64-Bit-Version von XMLSpy verwendet.
- Der Firebird JDBC-Treiber (eine oder mehrere .jar-Dateien, die die Verbindung zur Datenbank herstellen) muss auf Ihrem Betriebssystem verfügbar sein. In diesem Beispiel wird der JDBC-Treiber verwendet, der zur Verfügung steht, nachdem Sie die **IBM Data Server Client** Version 10.1 (64-Bit) installiert haben. Um die JDBC-Treiber zu installieren, wählen Sie eine **Standardinstallation** oder wählen Sie diese Option explizit im Installationsassistenten aus.



  Wenn Sie den Standardinstallationspfad nicht geändert haben, befinden sich die erforderlichen .jar-Dateien nach der Installation im Verzeichnis **C:\Programme\IBM\SQLLIB\java**.
- Sie benötigen die folgenden Datenbankverbindungsinformationen: Host, Port, Datenbankname, Benutzername und Passwort.

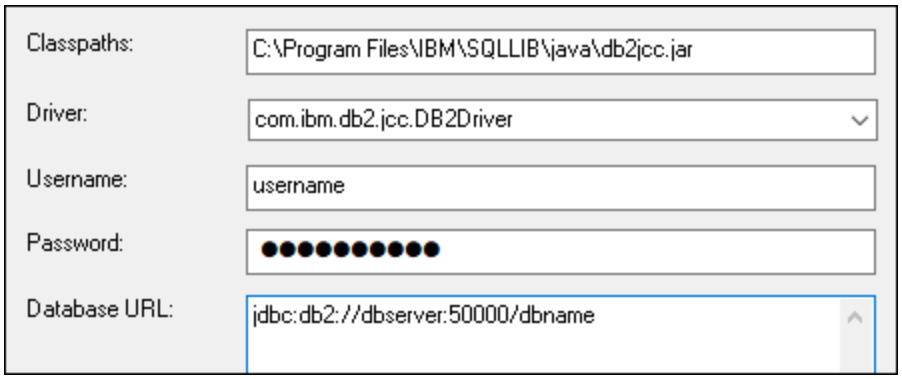
So stellen Sie über JDBC eine Verbindung zu IBM DB2 her:

1. Starten Sie den Datenbank-Verbindungsassistenten 970.
2. Klicken Sie auf **JDBC-Verbindungen**.
3. Geben Sie neben "Classpaths" den Pfad zur .jar-Datei , die die Verbindung zur Datenbank bereitstellt, ein. In diesem Beispiel wird der Pfad **C:\Programme\IBM\SQLLIB\java\db2jcc.jar** referenziert. Je nach Datenbankversion müssen Sie eventuell den Treiber **db2jcc4.jar** referenzieren. Informationen zur Treiberkompatibilität finden Sie in der IBM-Dokumentation unter (http://www-01.ibm.com/support/docview.wss?uid=swg21363866). Beachten Sie, dass Sie das Textfeld

 © 2016-2022 Altova GmbH

"Classpaths" leer lassen können, wenn Sie den/die .jar-Dateipfad(e) zur Umgebungsvariablen CLASSPATH des Betriebssystems hinzugefügt haben (siehe auch Konfigurieren des CLASSPATH 995).

4. Wählen Sie im Feld "Treiber" **com.ibm.db2.jcc.DB2Driver** aus. Beachten Sie, dass dieser Eintrag zur Verfügung steht, wenn entweder im Textfeld "Classpath" oder in der Umgebungsvariablen `CLASSPATH` des Betriebssystems eine gültige .jar-Datei gefunden wird (siehe vorheriger Schritt).



5. Geben Sie den Benutzernamen und das Passwort des Datenbankbenutzers in die entsprechenden Textfelder ein.
6. Geben Sie in das Textfeld **Datenbank-URL** den Connection String zum Datenbankserver ein. Dabei müssen Sie die Verbindungsinformationen durch die entsprechenden Werte für Ihren Datenbankserver ersetzen.

```
jdbc:db2://hostName:port/databaseName
```

7. Klicken Sie auf **Verbinden**.

19.1.10.4 IBM DB2 (ODBC)

Dieses Kapitel enthält eine Beispielanleitung für das Herstellen einer Verbindung zu einer IBM DB2-Datenbank über ODBC.

Voraussetzungen:

- Auf Ihrem Betriebssystem muss IBM Data Server Client installiert und konfiguriert sein (in diesem Beispiel wird IBM Data Server Client 9.7 verwendet). Eine Installationsanleitung dazu finden Sie in der Dokumentation zu Ihrer IBM DB2 Software. Nachdem Sie IBM Data Server Client installiert haben, überprüfen Sie, ob die ODBC-Treiber auf Ihrem Rechner verfügbar sind (siehe Anzeigen der verfügbaren ODBC-Treiber 991).
- Erstellen Sie einen Datenbank-Alias. Es gibt mehrere Methoden, dies zu tun:
  - über den IBM DB2-Konfigurationsassistenten
  - über den IBM DB2-Befehlszeilenprozessor
  - über den ODBC-Datenquellenassistenten (die Anleitung dazu finden Sie weiter unten)
- Sie haben die folgenden Datenbankverbindungsinformationen zur Verfügung: Host, Datenbank, Port, Benutzername und Passwort.

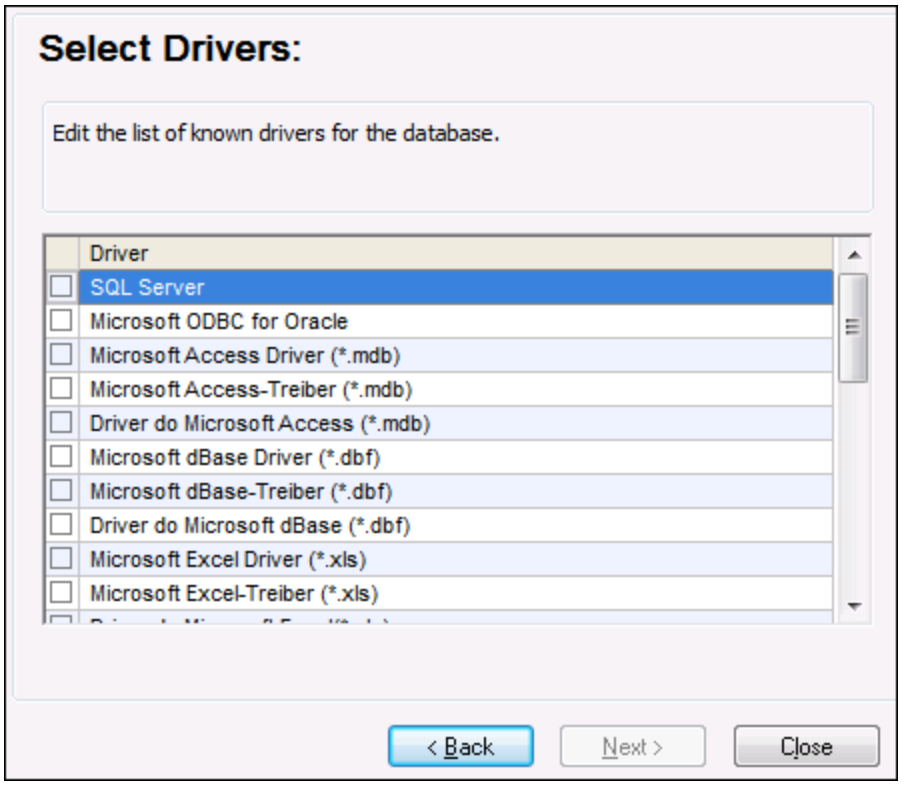
So stellen Sie eine Verbindung zu IBM DB2 her:

1. Starten Sie den Datenbank-Verbindungsassistenten[970] und wählen Sie **IBM DB2 (ODBC/JDBC)** aus.
2. Klicken Sie auf **Weiter**.



3. Wählen Sie **ODBC** aus und klicken Sie auf **Weiter.** Wenn Sie aufgefordert werden, die Liste der bekannten Treiber für die Datenbank zu bearbeiten, wählen Sie die Datenbanktreiber für IBM DB2 aus (siehe Voraussetzungen[1009]) und klicken Sie auf **Weiter**.

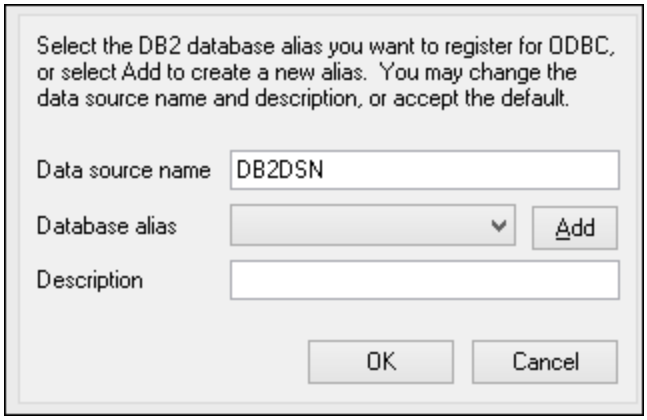
© 2016-2022 Altova GmbH



4. Wählen Sie den IBM DB2-Treiber aus der Liste aus und klicken Sie anschließend auf **Verbinden**. (Um die Liste der verfügbare Treiber zu bearbeiten, klicken Sie auf **Treiber bearbeiten** und aktivieren bzw. deaktivieren Sie anschließend die IBM DB2-Treiber, die Sie hinzufügen bzw. entfernen möchten.)

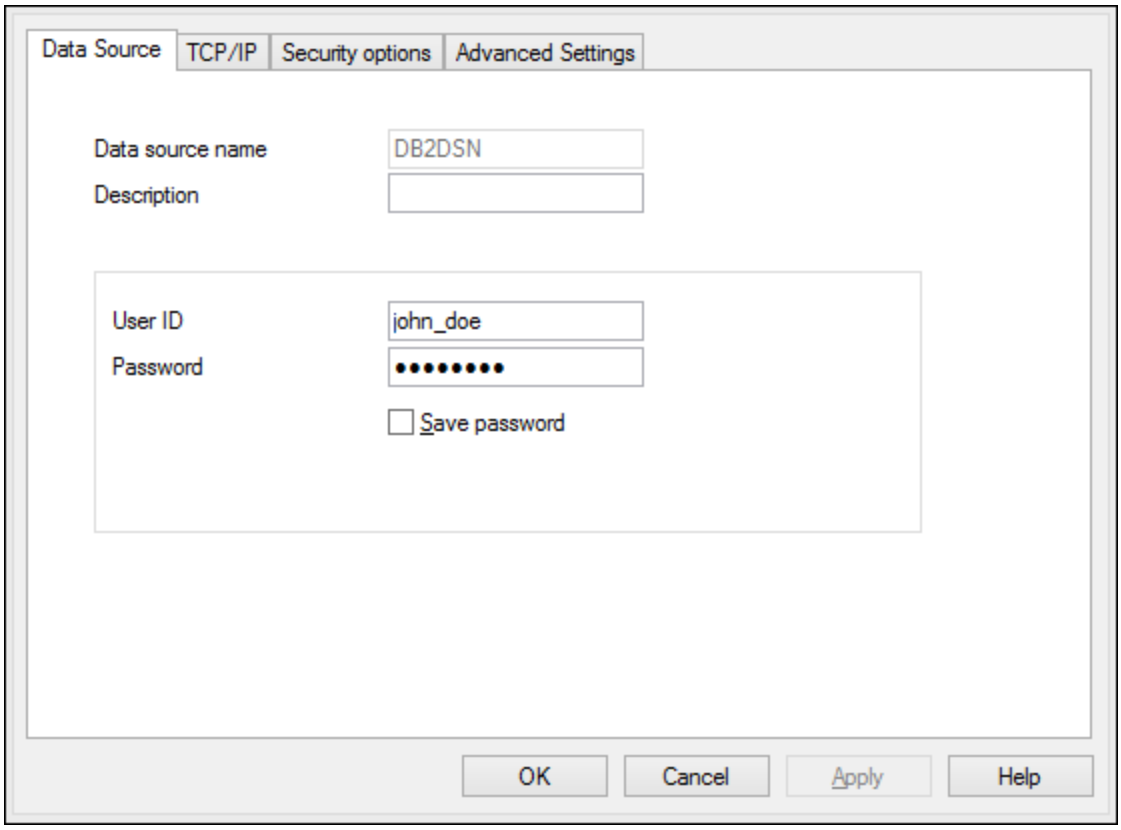


5. Geben Sie einen Datenquellennamen ein (in diesem Beispiel **DB2DSN**) und klicken Sie auf **Hinzufügen**.

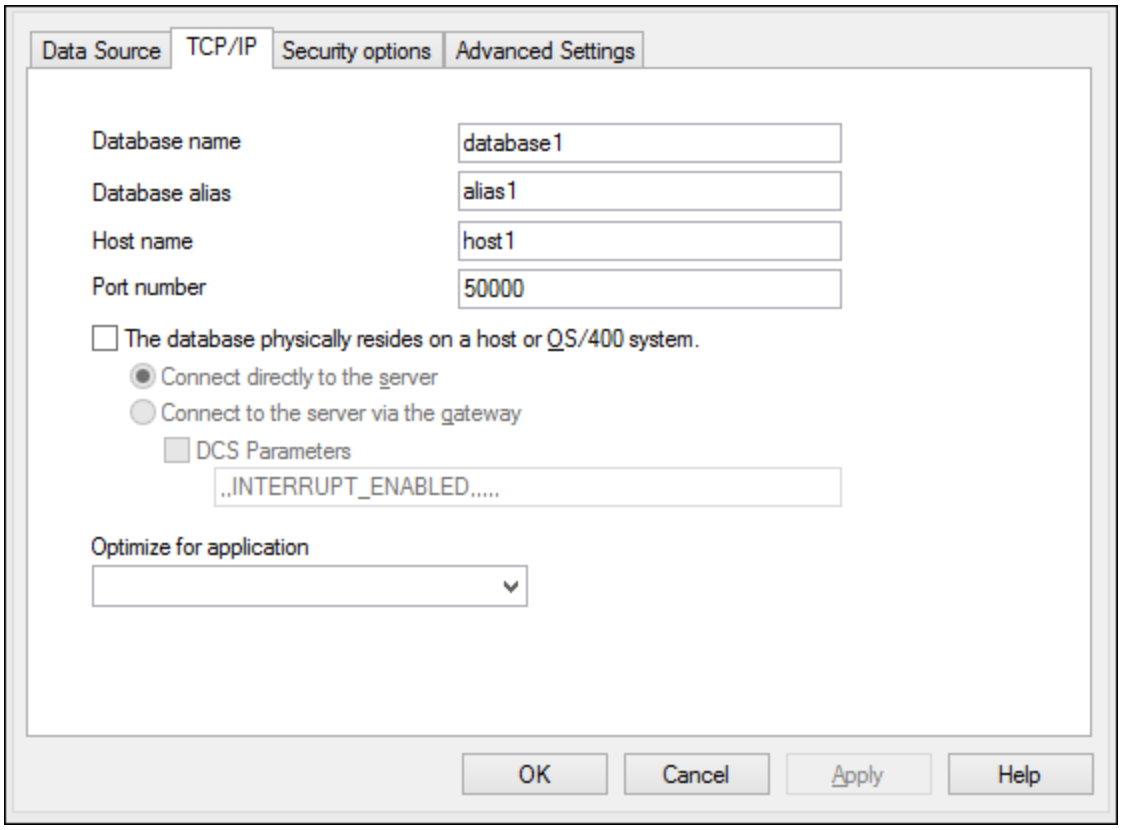


6. Geben Sie auf dem Register **Datenquelle** den Benutzernamen und das Passwort für die Datenbank ein.

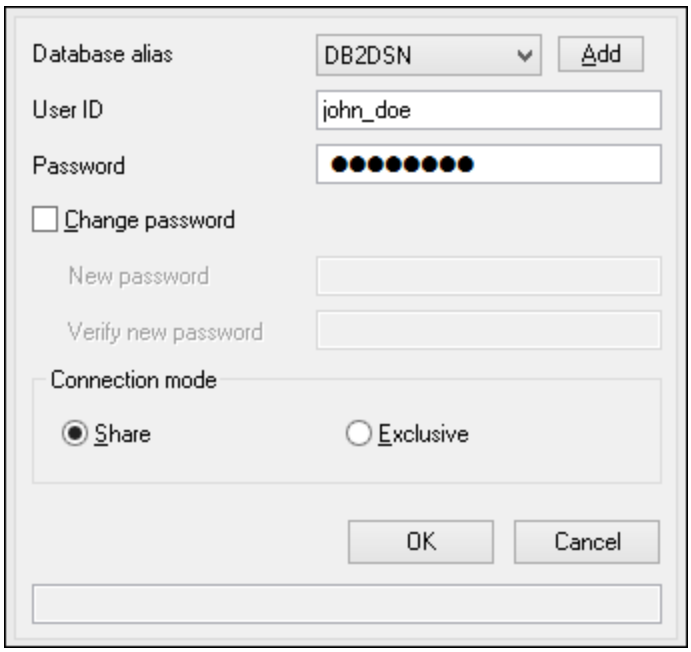
© 2016-2022 Altova GmbH



7. Geben Sie auf dem Register **TCP/IP** den Datenbanknamen, einen Namen für den Alias, den Host-Namen und die Port-Nummer ein und klicken Sie auf OK.



8. Geben Sie den Benutzernamen und das Passwort erneut ein und klicken Sie auf **OK**.



 © 2016-2022 Altova GmbH

19.1.10.5 IBM DB2 für i (JDBC)

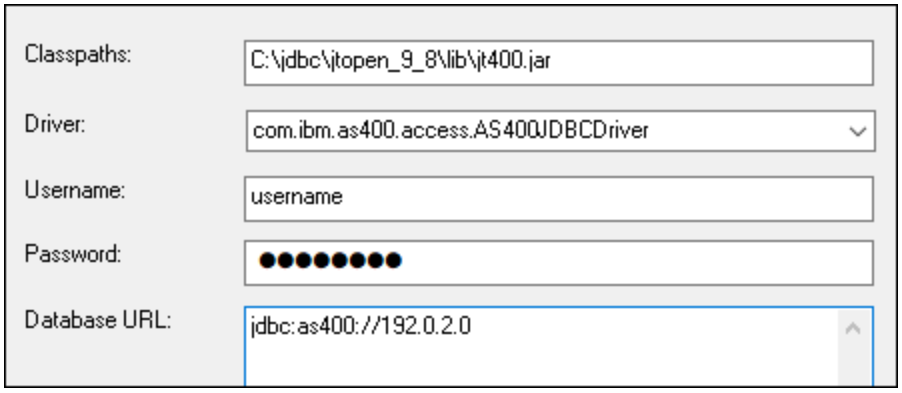
Dieses Kapitel enthält eine Beispielanleitung für das Herstellen einer Verbindung zu einem **IBM DB2 für i**-Datenbankserver über JDBC.

Voraussetzungen:

- Auf Ihrem Betriebssystem muss JRE (Java Runtime Enviroment) oder Java Development KIT (JDK) installiert sein. Dabei muss es sich entweder um Oracle JDK oder einen Open Source Build wie Oracle OpenJDK handeln.XMLSpy ermittelt den Pfad zur Java Virtual Machine (JVM) anhand der folgenden Ordner und zwar in folgender Reihenfolge: a) anhand des benutzerdefinierten JVM-Pfads, den Sie eventuell in den **Applikationsoptionen** definiert haben; b) anhand des JVM-Pfads in der Windows Registry; c) anhand der `JAVA_HOME`-Umgebungsvariablen.
- Stellen Sie sicher, dass die Plattform von XMLSpy (32-Bit, 64-Bit) mit der des JRE/JDK übereinstimmt. In diesem Beispiel wird Oracle OpenJDK 11.0 64-Bit verwendet, folglich wird auch die 64-Bit-Version von XMLSpy verwendet.
- Der Firebird JDBC-Treiber (eine oder mehrere .jar-Dateien, die die Verbindung zur Datenbank herstellen) muss auf Ihrem Betriebssystem verfügbar sein. In diesem Beispiel wird die Open Source **Toolbox for Java/JTOpen** Version 9.8 (http://jt400.sourceforge.net/) verwendet. Nachdem Sie das Paket heruntergeladen und in ein lokales Verzeichnis entpackt haben, stehen die erforderlichen .jar-Dateien im Unterverzeichnis **lib** zur Verfügung.
- Sie benötigen die folgenden Datenbankverbindungsinformationen: Host, Benutzername und Passwort.

So stellen Sie über JDBC eine Verbindung zu IBM DB2 für i her:

1. Starten Sie den Datenbank-Verbindungsassistenten[970].
2. Klicken Sie auf **JDBC-Verbindungen**.
3. Geben Sie neben "Classpaths" den Pfad zur .jar-Datei , die die Verbindung zur Datenbank bereitstellt, ein. In diesem Beispiel wird der Pfad **C:\jdbc\jtopen_9_8\jt400.jar** referenziert. Beachten Sie, dass Sie das Textfeld "Classpaths" leer lassen können, wenn Sie den/die .jar-Dateipfad(e) zur Umgebungsvariablen CLASSPATH des Betriebssystems hinzugefügt haben (siehe auch Konfigurieren des CLASSPATH[995]).
4. Wählen Sie im Feld "Treiber" **com.ibm.as400.access.AS400JDBCDriver** aus. Beachten Sie, dass dieser Eintrag zur Verfügung steht, wenn entweder im Textfeld "Classpath" oder in der Umgebungsvariablen `CLASSPATH` des Betriebssystems eine gültige .jar-Datei gefunden wird (siehe vorheriger Schritt).



5. Geben Sie den Benutzernamen und das Passwort des Datenbankbenutzers in die entsprechenden Textfelder ein.
6. Geben Sie in das Textfeld **Datenbank-URL** den JDBC-Connection String zum Datenbankserver ein. Dabei müssen Sie `host` durch den Hostnamen oder die IP-Adresse Ihres Datenbankservers ersetzen.

```
jdbc:as400://host
```

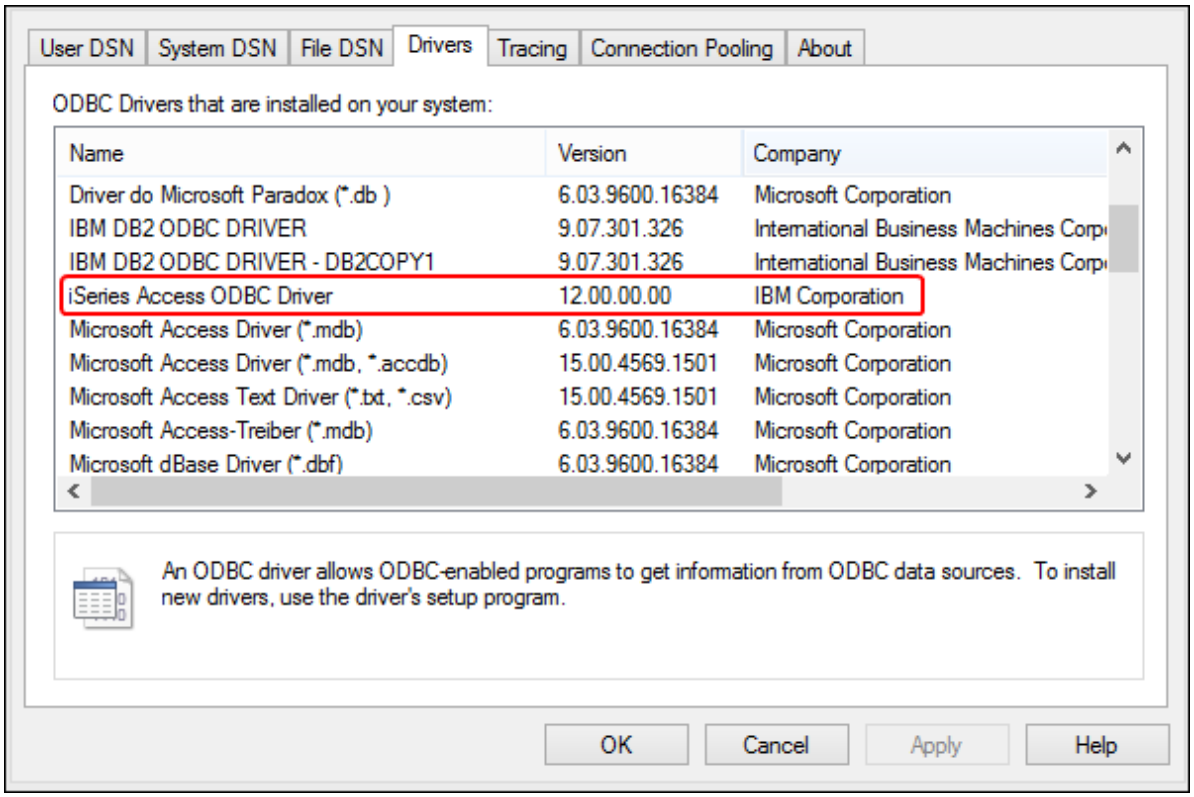
7. Klicken Sie auf **Verbinden**.

19.1.10.6 IBM DB2 für i (ODBC)

Dieses Kapitel enthält eine Beispielanleitung für das Herstellen einer Verbindung zu einer *IBM DB2 für i*-Datenbank über ODBC.

Voraussetzungen:

- Auf Ihrem Betriebssystem muss *IBM System i Access für Windows* installiert sein (in diesem Beispiel wird *IBM System i Access für Windows V6R1M0 verwendet*). Eine Installationsanleitung finden Sie in der Dokumentation zu Ihrer *IBM DB2 für i*-Software. Überprüfen Sie nach der Installation, ob der ODBC-Treiber auf Ihrem Rechner verfügbar ist (siehe Anzeigen der verfügbaren OBBC-Treiber 991).



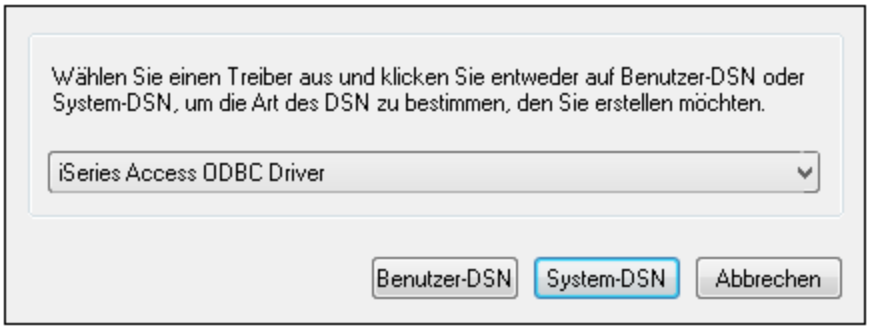
- Sie haben die folgenden Datenbankverbindungsinformationen zur Verfügung: die IP-Adresse des Datenbankservers, den Datenbank-Benutzernamen und das Passwort.
- Führen Sie den *System i Navigator* aus und befolgen Sie die Anweisungen des Assistenten, um eine neue Verbindung zu erstellen. Wenn Sie nach einem System gefragt werden, geben Sie die IP-

 © 2016-2022 Altova GmbH

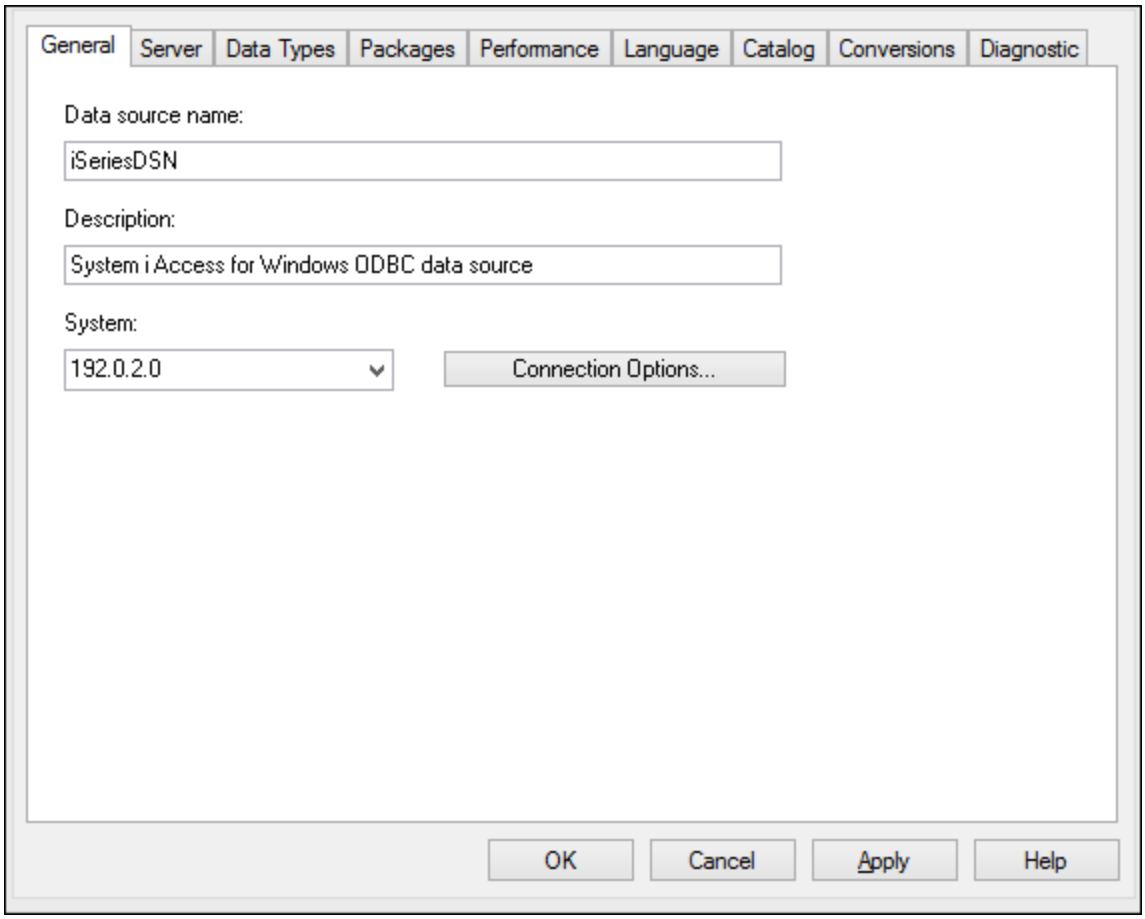
Adresse des Datenbankservers ein. Nachdem Sie die Verbindung hergestellt haben, sollten Sie diese überprüfen (Klicken Sie auf den Verbindung und wählen Sie **Datei** > **Diagnose** > **Verbindung überprüfen**). Wenden Sie sich an den Datenbankserver-Administrator, wenn Sie Verbindungsfehler erhalten.

So stellen Sie eine Verbindung zu IBM DB2 für i ein:

1. Starten Sie den Datenbank-Verbindungsassistenten 970.
2. Klicken Sie auf **ODBC-Verbindungen**.
3. Klicken Sie auf **Benutzer-DSN** (Klicken Sie alternativ dazu auf **System-DSN** oder **Datei-DSN** - die darauf folgende Anleitung ist ähnlich).
4. Klicken Sie auf **Hinzufügen** .
5. Wählen Sie aus der Liste **iSeries Access ODBC Driver** aus und klicken Sie auf **Benutzer-DSN** (bzw. gegebenenfalls auf **System- DSN**).

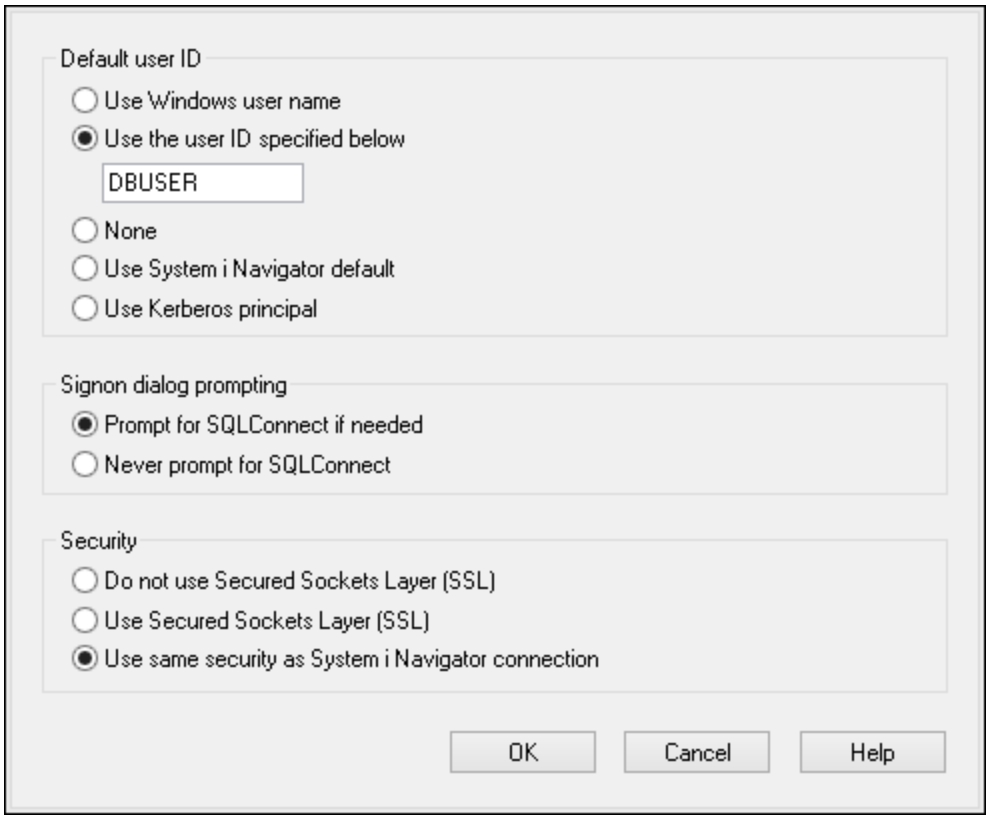


6. Geben Sie den Datenquellennamen ein und wählen Sie die Verbindung aus der System-Auswahlliste aus. In diesem Beispiel lautet der Datenquellenname **iSeriesDSN** und das System ist **192.0.2.0**.



7. Klicken Sie auf **Verbindungsoptionen** und wählen Sie **unten angeführte Benutzer-ID verwenden** und geben Sie den Namen des Datenbankbenutzers ein (in diesem Beispiel **DBUSER**).

© 2016-2022 Altova GmbH



8. Klicken Sie auf **OK**. Die neue Datenquelle steht nun in der Liste der DSNs zur Verfügung.
9. Klicken Sie auf **Verbinden**.
10. Geben Sie den Benutzernamen und das Passwort für die Datenbank ein, wenn Sie dazu aufgefordert werden und klicken Sie auf **OK**.

19.1.10.7 IBM Informix (JDBC)

Dieses Kapitel enthält eine Beispielanleitung für das Herstellen einer Verbindung zu einer IBM Informix-Datenbank über JDBC.

Voraussetzungen:

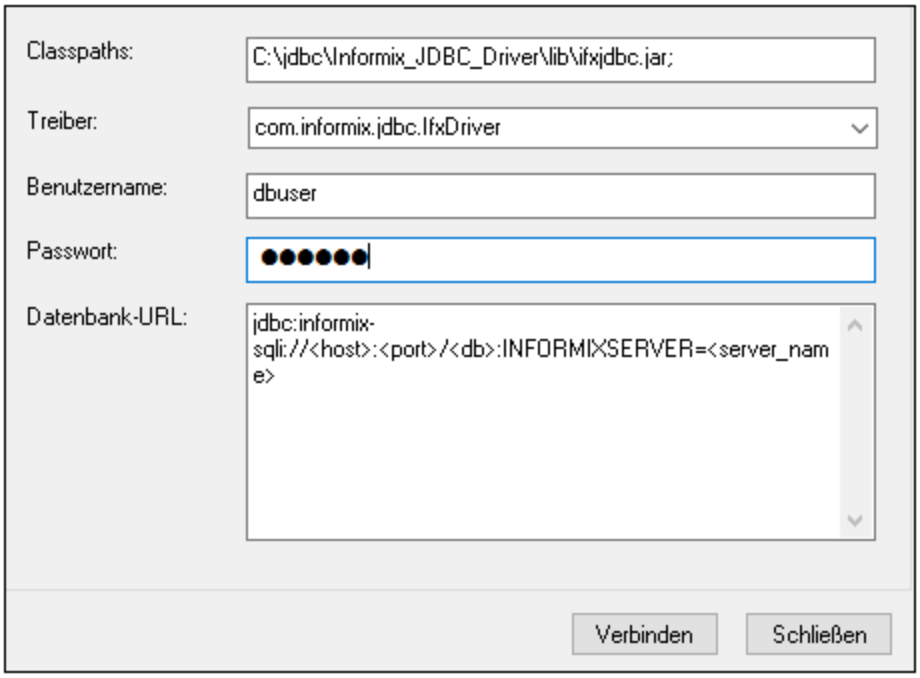
- Auf Ihrem Betriebssystem muss JRE (Java Runtime Enviroment) oder Java Development KIT (JDK) installiert sein. Dabei muss es sich entweder um Oracle JDK oder einen Open Source Build wie Oracle OpenJDK handeln.XMLSpy ermittelt den Pfad zur Java Virtual Machine (JVM) anhand der folgenden Ordner und zwar in folgender Reihenfolge: a) anhand des benutzerdefinierten JVM-Pfads, den Sie eventuell in den **Applikationsoptionen** definiert haben; b) anhand des JVM-Pfads in der Windows Registry; c) anhand der `JAVA_HOME`-Umgebungsvariablen.
- Stellen Sie sicher, dass die Plattform von XMLSpy (32-Bit, 64-Bit) mit der des JRE/JDK übereinstimmt.
- Der JDBC-Treiber (eine oder mehrere .jar-Dateien, die die Verbindung zur Datenbank herstellen) muss auf Ihrem Betriebssystem installiert sein. In diesem Beispiel wird die IBM Informix JDBC-Treiberversion 3.70 verwendet. Die Installationsanleitung zum Treiber finden Sie in der dazugehörigen Dokumentation bzw. im "IBM Informix JDBC Driver Programmer's Guide").

- Sie haben die folgenden Datenbankinformationen zur Verfügung: Host, Name des Informix-Servers, Datenbank, Port. Benutzername und Passwort.

So stellen Sie über JDBC eine Verbindung zu IBM Informix her:

1. Starten Sie den Datenbank-Verbindungsassistenten[970].
2. Klicken Sie auf **JDBC-Verbindungen**.
3. Geben Sie neben "Classpaths" den Pfad zur .jar-Datei , die die Verbindung zur Datenbank bereitstellt, ein. Falls nötig, können Sie auch eine durch Semikola getrennte Liste von .jar-Dateipfaden eingeben. Die benötigte .jar-Datei in diesem Beispiel befindet sich unter dem folgenden Pfad: **C:\Informix_JDBC_Driver\lib\ifxjdbc.jar**. Beachten Sie, dass Sie das Textfeld "Classpaths" leer lassen können, wenn Sie den/die .jar-Dateipfad(e) zur Umgebungsvariablen CLASSPATH des Betriebssystems hinzugefügt haben (siehe auch Konfigurieren des CLASSPATH[995]).

4. Wählen Sie im Feld "Treiber" **com.informix.jdbc.IfxDriver** aus. Beachten Sie, dass dieser Eintrag zur Verfügung steht, wenn entweder im Textfeld "Classpaths" oder in der Umgebungsvariablen `CLASSPATH` des Betriebssystems eine gültige .jar-Datei gefunden wird (siehe vorheriger Schritt).



5. Geben Sie den Benutzernamen und das Passwort für die Datenbank in die entsprechenden Textfelder ein.
6. Geben Sie den Connection String zum Datenbankserver in das Textfeld "Datenbank-URL" ein, indem Sie die hervorgehobenen Werte durch die entsprechenden Werte für Ihren Datenbankserver ersetzen.

```
jdbc:informix-sqli://hostName:port/databaseName:INFORMIXSERVER=myserver;
```

7. Klicken Sie auf **Verbinden.**

© 2016-2022 Altova GmbH

19.1.10.8 MariaDB (ODBC)

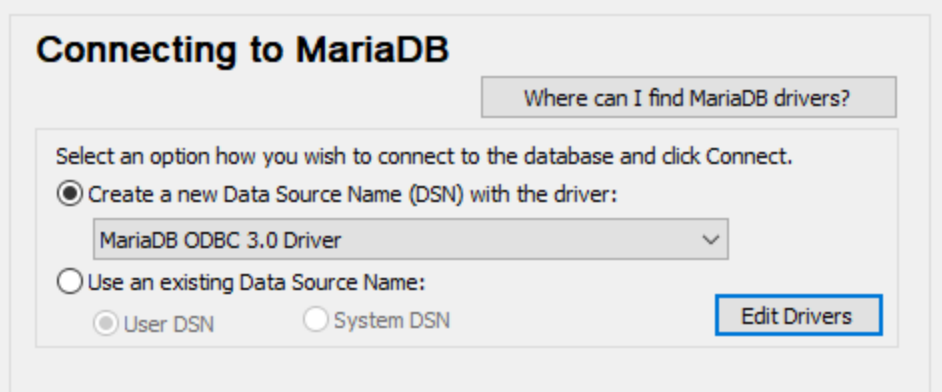
In diesem Beispiel wird gezeigt, wie Sie über ODBC eine Verbindung zu einem MariaDB-Datenbankserver herstellen.

Voraussetzungen:

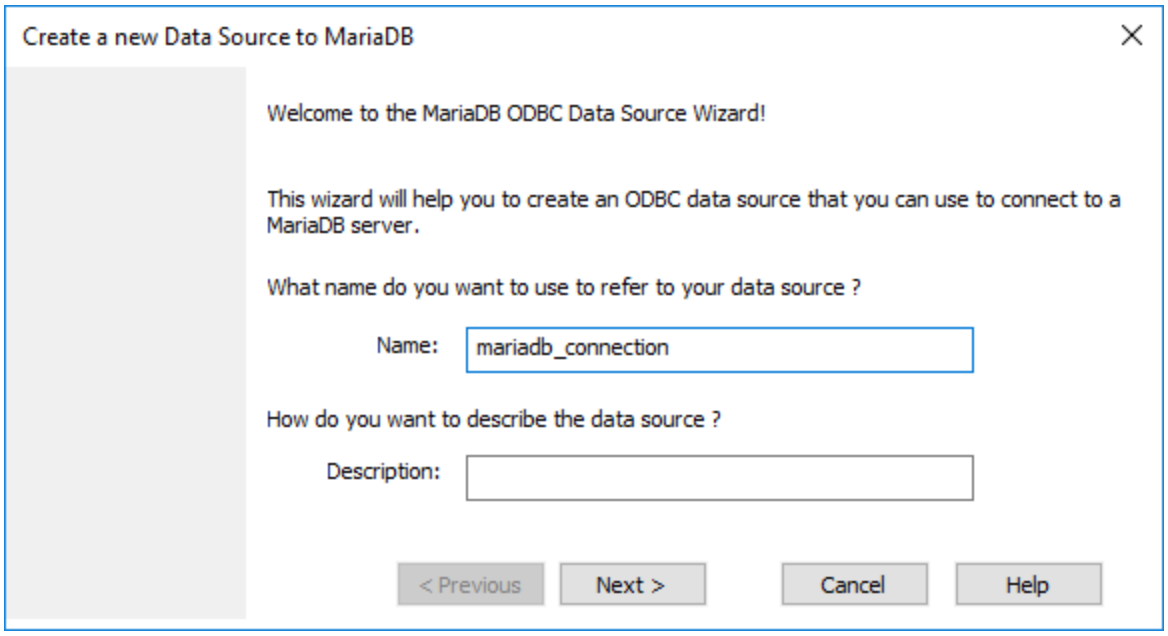
- Der MariaDB Connector/ODBC (https://downloads.mariadb.org/connector-odbc/) muss installiert sein.
- Sie haben die folgenden Datenbankinformationen zur Verfügung: Host, Datenbank, Port. Benutzername und Passwort.

So stellen Sie über ODBC eine Verbindung zu MariaDB her:

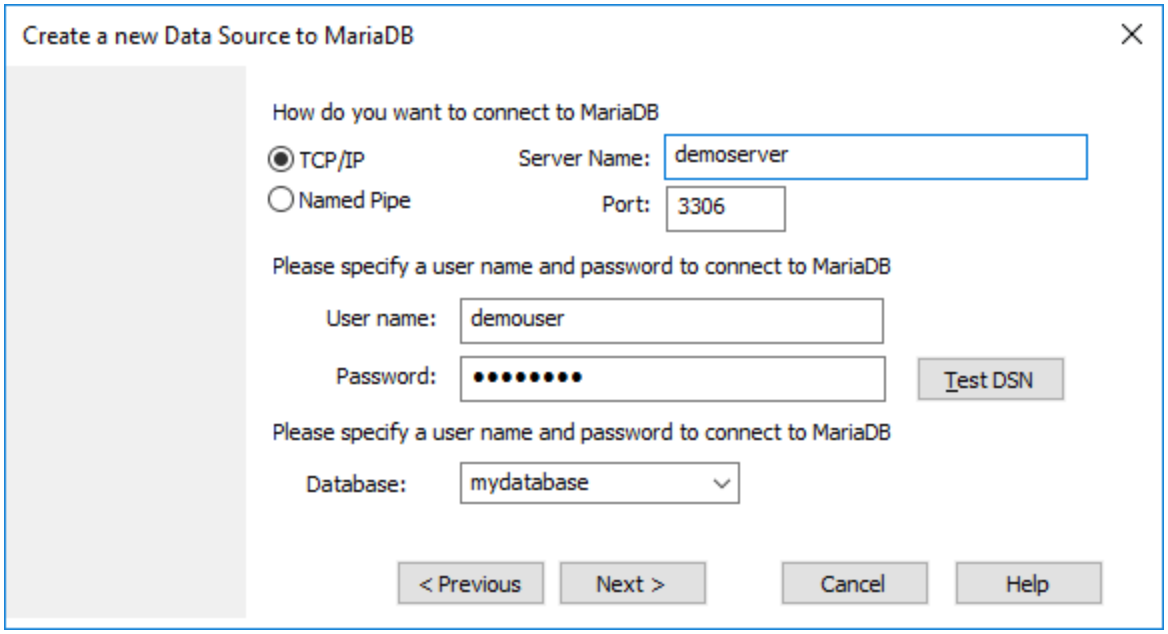
1. Starten Sie den Datenbank-Verbindungsassistenten 970.
2. Wählen Sie **MariaDB (ODBC)** aus und klicken Sie auf **Weiter**.



3. Wählen Sie **Erstelle neuen Data Source Name (DSN) mit dem Treiber** und wählen Sie **MariaDB ODBC 3.0 Driver** aus. Wenn in der Liste kein MySQL-Treiber verfügbar ist, klicken Sie auf **Treiber bearbeiten** und wählen Sie einen beliebigen verfügbaren MariaDB-Treiber aus (die Liste enthält alle auf Ihrem Betriebssystem installierten ODBC-Treiber).
4. Klicken Sie auf **Verbinden**.

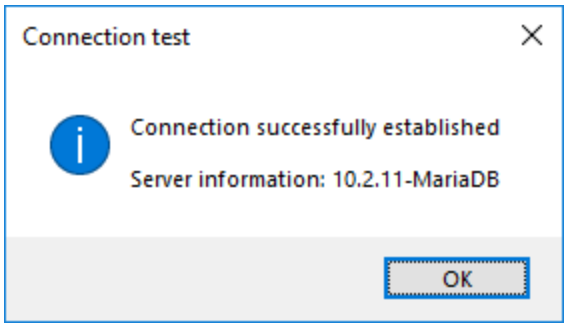


5. Geben Sie einen Namen und optional eine Beschreibung für diese ODBC-Datenquelle ein.



6. Füllen Sie die Anmeldeinformationen für die Datenbankverbindung aus (TCP/IP Server, Benutzer, Passwort), wählen Sie eine Datenbank aus und klicken Sie auf **DSN testen**. Wenn die Verbindung erfolgreich hergestellt werden konnte, wird eine Meldung angezeigt:

© 2016-2022 Altova GmbH



7. Klicken Sie auf **Weiter** und schließen Sie den Vorgang ab. Von Fall zu Fall werden eventuell auch andere Parameter benötigt (z.B. SSL-Zertifikate, wenn Sie eine sichere Verbindung zu MariaDB herstellen möchten).

Anmerkung: Wenn es sich um einen entfernten Datenbankserver handelt, muss er vom Server-Administrator so konfiguriert werden, dass er entfernte Verbindungen von der IP-Adresse Ihres Rechners aus zulässt.

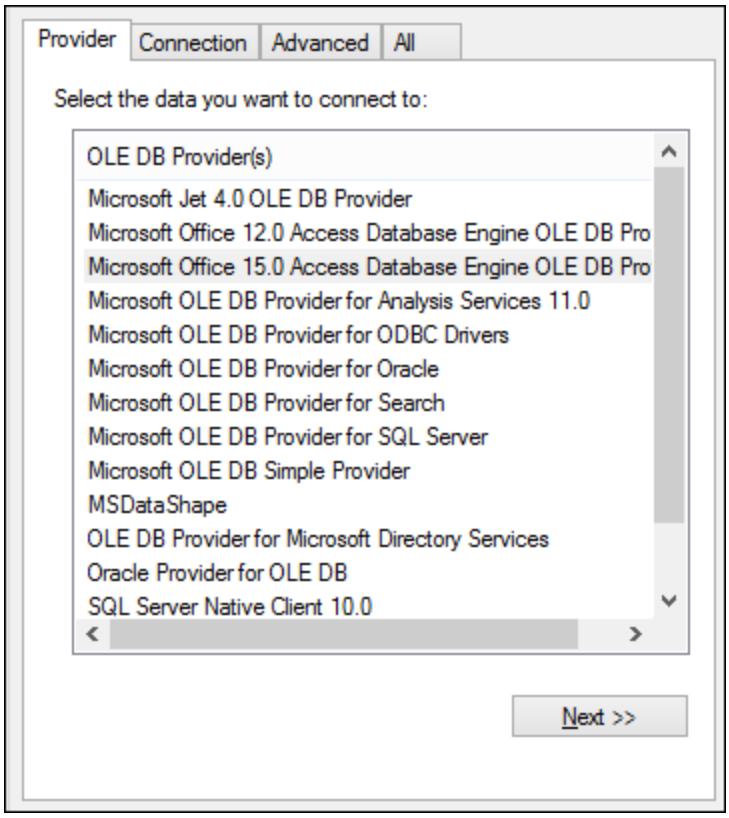
19.1.10.9 Microsoft Access (ADO)

Eine einfache Methode, eine Verbindung zu einer Microsoft Access-Datenbank herzustellen, ist, den Anweisungen des Assistenten zu folgen und zur Datenbankdatei zu navigieren, wie unter Herstellen einer Verbindung zu einer vorhandenen Microsoft Access-Datenbank[978] beschrieben. Alternativ dazu können Sie explizit eine ADO-Verbindung definieren, wie im Folgenden gezeigt. Diese Methode empfiehlt sich, wenn Ihre Datenbank durch ein Passwort geschützt ist.

Sie können zwar auch über ODBC eine Verbindung zu Microsoft Access herstellen, doch sollte diese Methode vermieden werden, da sich dadurch einige Einschränkungen ergeben.

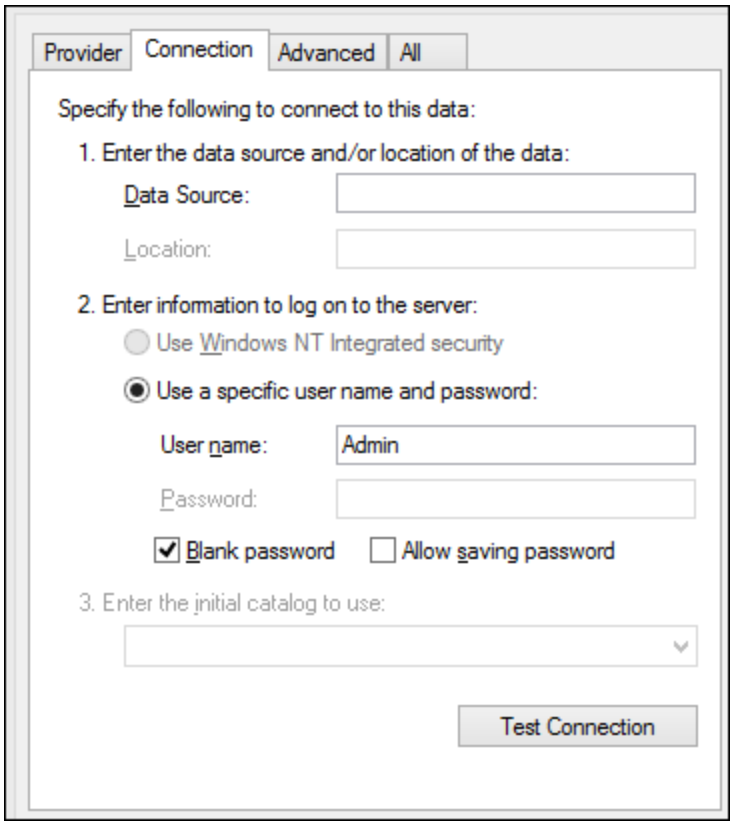
Herstellen einer Verbindung zu einer durch ein Passwort geschützten Microsoft Access-Datenbank:

1. Starten Sie den Datenbank-Verbindungsassistenten[970].
2. Klicken Sie auf **ADO-Verbindungen**.
3. Klicken Sie auf **Erzeugen**.

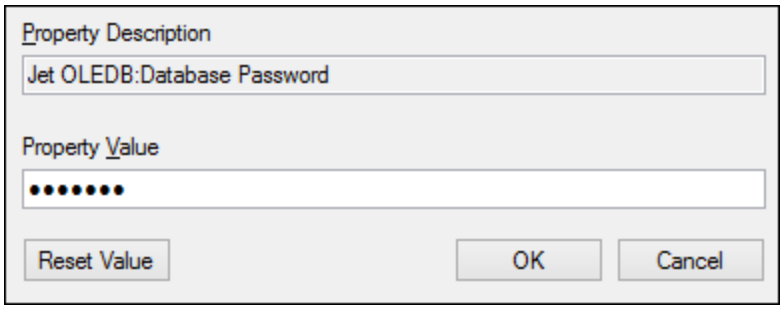


4. Wählen Sie den **Microsoft Office 15.0 Access Database Engine OLE DB Provider** aus und klicken Sie auf **Weiter**.

© 2016-2022 Altova GmbH



5. Geben Sie in das Feld "Datenquelle" den Pfad zur Microsoft Access-Datei im UNC-Format ein, z.B. \ **\myserver\\mynetworkshare\Reports\Revenue.accdb**, wobei **myserver** der Name des Servers und **mynetworkshare** der Name des gemeinsamen Netzwerklaufwerks ist.
6. Doppelklicken Sie auf dem Register **Alle** auf die Eigenschaft **Jet OLEDB:Database Password** und geben Sie das Datenbank-Passwort als Eigenschaftswert ein.



Anmerkung: Wenn die Verbindung immer noch nicht hergestellt werden kann, suchen Sie die Arbeitsgruppen-Informationsdatei (**System.MDW**) für Ihr Benutzerprofil und setzen Sie den Wert der -Eigenschaft **Jet OLEDB: System** auf den Pfad der Datei **System.MDW**.

19.1.10.10 Microsoft SQL Server (ADO)

In diesem Beispiel wird gezeigt, wie Sie über ADO eine Verbindung zu einer SQL Server-Datenbank herstellen. Diese Anleitung gilt für die Verwendung des empfohlenen **Microsoft OLE DB-Treibers für SQL Server**

(MSOLEDBSQL), der von https://docs.microsoft.com/en-us/sql/connect/oledb/download-oledb-driver-for-sql-server?view=sql-server-ver15 heruntergeladen werden kann.

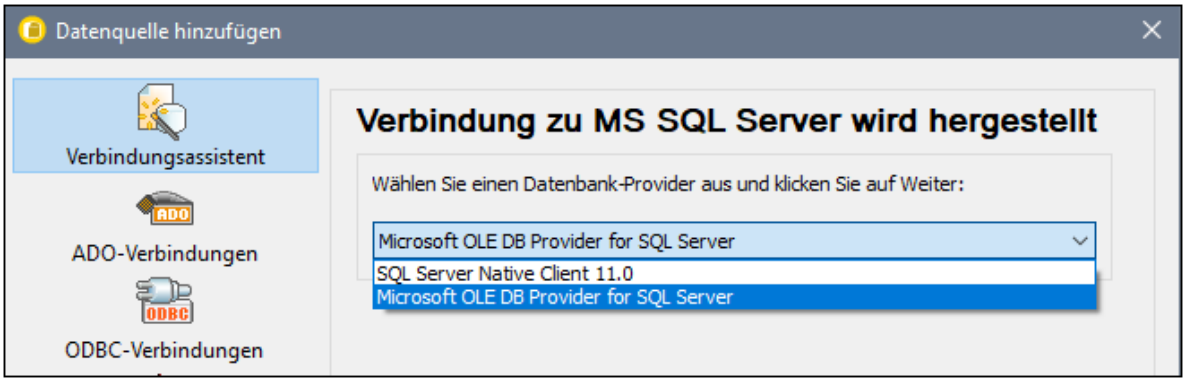
Bevor Sie die Schritte aus dieser Anleitung durchführen, stellen Sie sicher, dass Sie den oben genannten Anbieter heruntergeladen und auf Ihrem Rechner installiert haben. Der ADO-Anbieter muss mit der Plattform von XMLSpy (32-Bit oder 64-Bit) übereinstimmen.

Wenn Sie andere ADO-Anbieter wie **SQL Server Native Client (SQLNCLI)** oder **Microsoft OLE DB-Anbieter für SQL Server (SQLOLEDB)** verwenden möchten, ist die Vorgangsweise ähnlich, doch sind diese Anbieter veraltet und werden daher nicht empfohlen. Damit die Verbindung zu einem veralteten Anbieter hergestellt werden kann, müssen Sie außerdem eventuell zusätzliche Verbindungseigenschaften konfigurieren, wie unter Einrichten der SQL Server-Datenverknüpfungseigenschaften[979] beschrieben.

Es ist bekannt, dass es beim **Microsoft OLE DB-Anbieter für SQL Server (SQLOLEDB)** zu Problemen mit der Parameterbindung komplexer Abfragen wie Common Table Expressions (CTE) und verschachtelten SELECT-Anweisungen kommt.

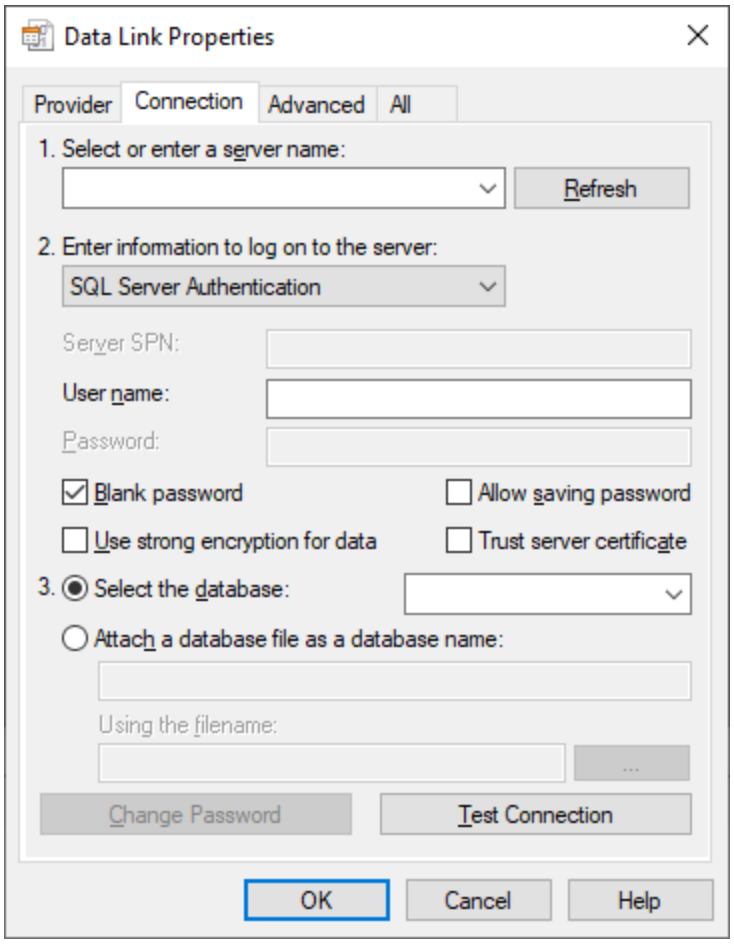
So stellen Sie eine Verbindung zu SQL Server her:

1. Starten Sie den Datenbank-Verbindungsassistenten[970].
2. Wählen Sie **Microsoft SQL Server (ADO)** und klicken Sie auf **Weiter**. Daraufhin wird die Liste der verfügbaren ADO-Anbieter angezeigt. In diesem Beispiel wird der **Microsoft OLE DB-Treiber für SQL Server** verwendet. Falls er in der Liste nicht enthalten ist, überprüfen Sie, ob er, wie oben erwähnt, auf Ihrem Rechner installiert ist.

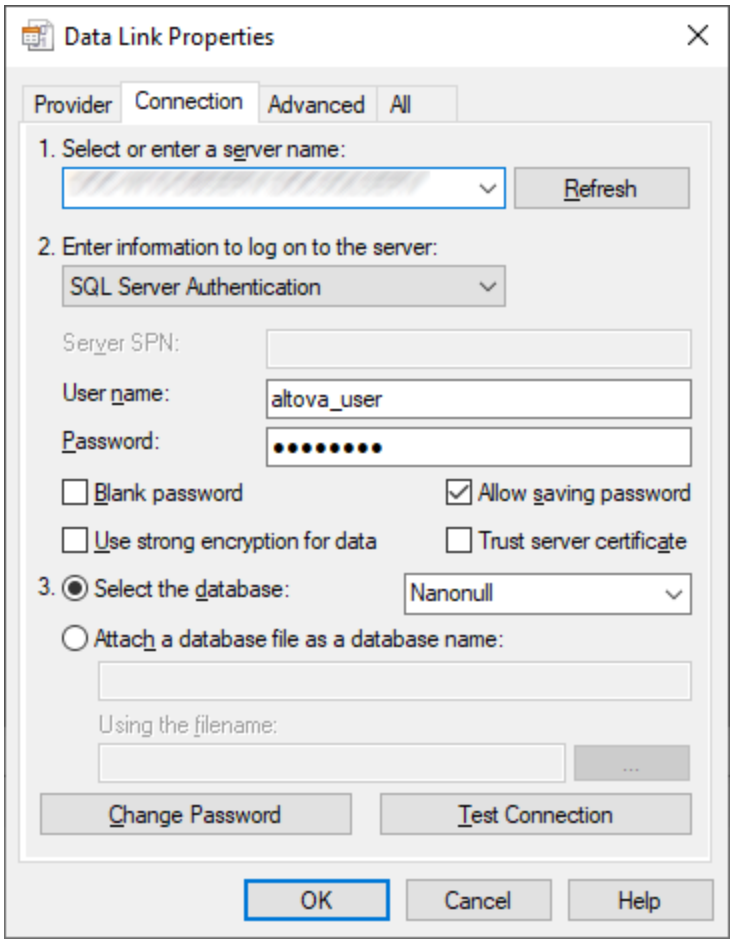


3. Klicken Sie auf **Weiter**. Daraufhin wird das Dialogfeld "Datenverknüpfungseigenschaften" angezeigt.

© 2016-2022 Altova GmbH



4. Wählen Sie den Namen des Datenbankservers aus oder geben Sie ihn ein (z.B. **SQLSERV01**). Wenn Sie eine Verbindung zu einer benannten SQL-Server-Instanz herstellen, sieht der Servername folgendermaßen aus: **SQLSERV01\INSTANZ**.
5. Wenn der Datenbankserver so konfiguriert ist, dass er Verbindungen von bei der Windows Domain angemeldeten Benutzern gestattet, wählen Sie **Windows-Authentifizierung**. Wählen Sie andernfalls **SQL Server-Authentifizierung**, deaktivieren Sie das Kontrollkästchen **Leeres Kennwort** und geben Sie die Anmeldeinformationen für die Datenbank in die entsprechenden Felder ein.
6. Aktivieren Sie das Kontrollkästchen **Passwort speichern zulassen** und wählen Sie die gewünschte Datenbank aus (in diesem Beispiel "Nanonull").



7. Um die Verbindung zu diesem Zeitpunkt zu überprüfen, klicken Sie auf **Verbindung testen**. Dieser Schritt ist optional, wird aber empfohlen.
8. Klicken Sie auf **OK.**

19.1.10.11 Microsoft SQL Server (ODBC)

In diesem Beispiel wird gezeigt, wie Sie über ODBC eine Verbindung zu einer SQL Server-Datenbank herstellen.

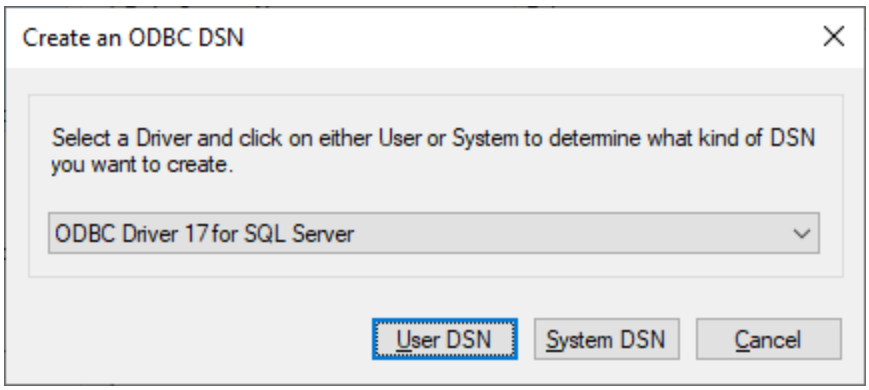
Voraussetzungen:

- Laden Sie den **Microsoft ODBC-Treiber für SQL Server** von der Microsoft-Website herunter und installieren Sie ihn, siehe https://docs.microsoft.com/en-us/SQL/connect/odbc/download-odbc-driver-for-sql-server. In diesem Beispiel wird für die Verbindung mit einer **SQL Server 2016**-Datenbank der **Microsoft ODBC-Treiber 17 für SQL Server** verwendet. Je nach gewünschter SQL-Server-Version müssen Sie eventuell eine andere ODBC-Treiberversion herunterladen. Informationen über von Ihrer SQL Server-Datenbank unterstützte ODBC-Treiberversionen schlagen Sie bitte unter den Systemanforderungen des Treibers nach.

© 2016-2022 Altova GmbH

So stellen Sie über ODBC eine Verbindung zu SQL Server her:

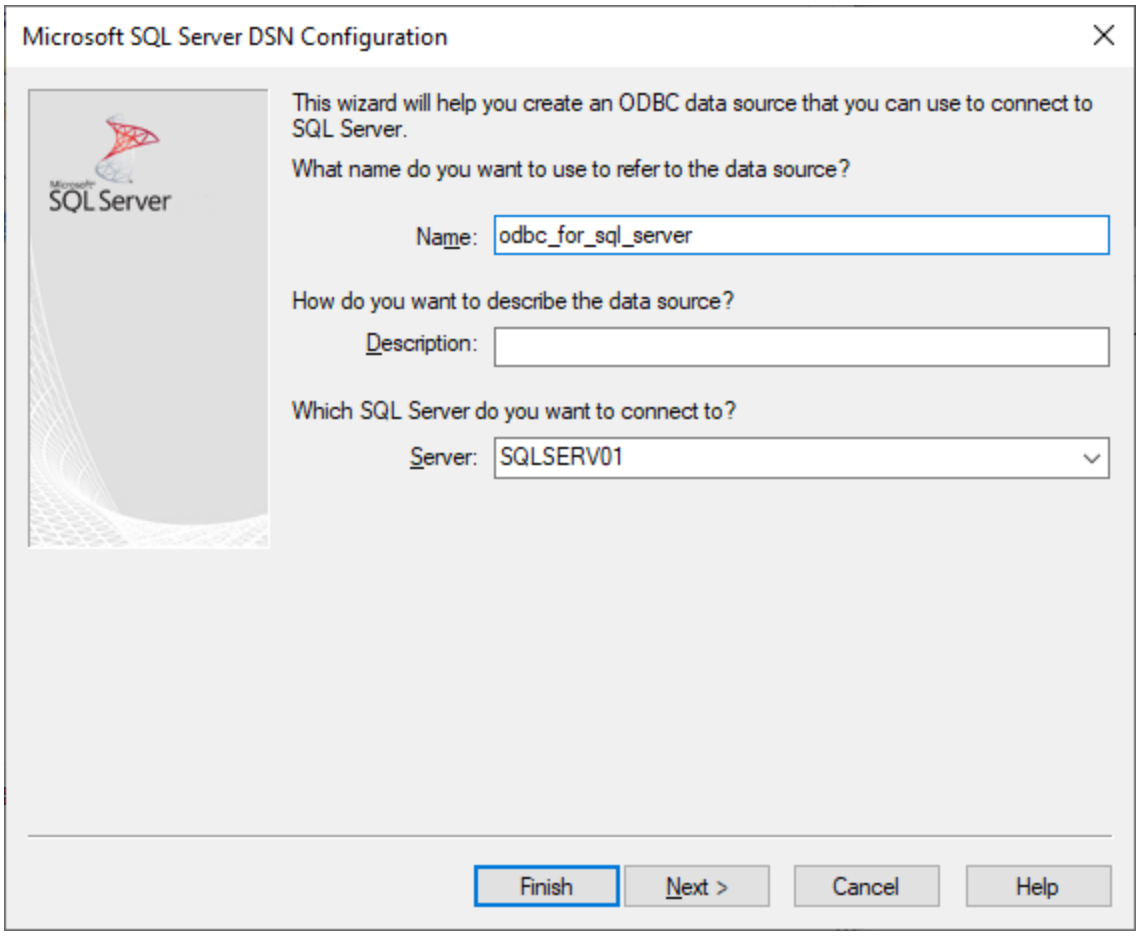
1. Starten Sie den Datenbank-Verbindungsassistenten 970.
2. Klicken Sie auf **ODBC-Verbindungen**.
3. Wählen Sie **Benutzer-DSN** (oder **System-DSN**, wenn Sie Administratorrechte haben) und klicken Sie auf **Hinzufügen** .
4. Wählen Sie den Treiber aus der Liste aus. Beachten Sie, dass der Treiber erst nach Installation auf der Liste angezeigt wird.



5. Wählen Sie **Benutzer-DSN** (oder **System-DSN**, wenn Sie einen System-DSN erstellen).

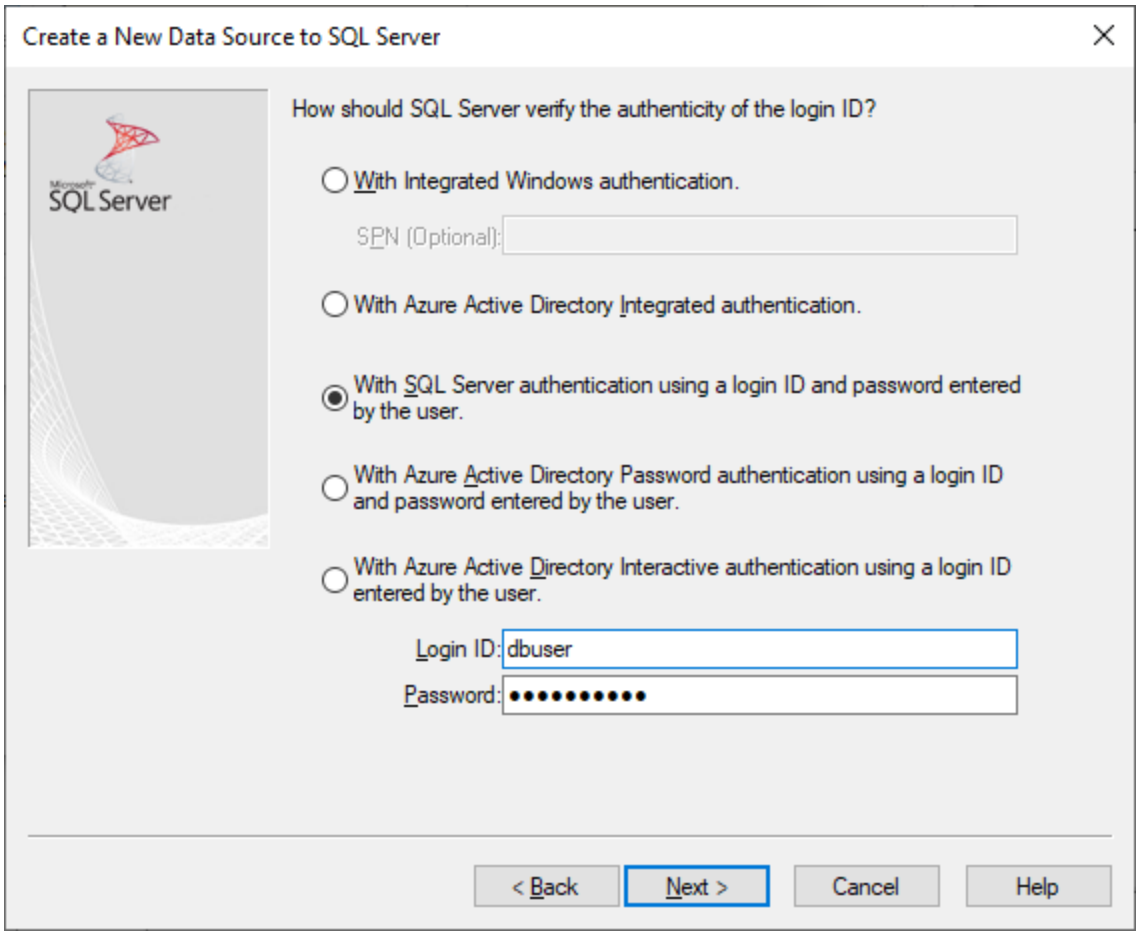
> Um einen **System-DSN** zu erstellen, müssen Sie XMLSpy als Administrator ausführen. Um daher einen **System-DSN** zu erstellen, brechen Sie den Assistenten ab, stellen Sie sicher, dass Sie XMLSpy als Administrator ausführen und führen Sie die obigen Schritte erneut durch.

6. Geben Sie einen Namen und optional eine Beschreibung für diese Verbindung ein und wählen Sie anschließend aus der Liste den gewünschten SQL Server aus (in diesem Beispiel **SQLSERV01**).

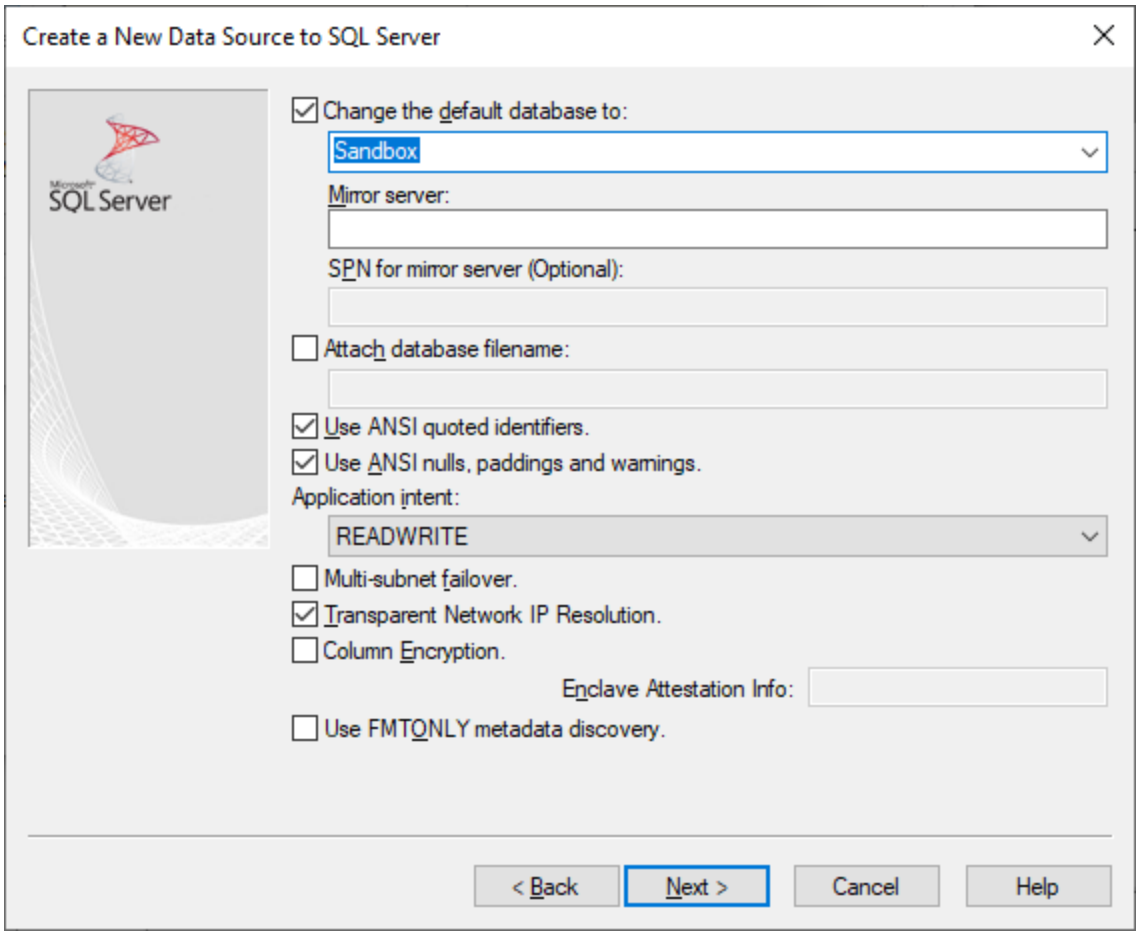


7. Wählen Sie die Option **Mit integrierter Windows NT-Authentifizierung**, wenn der Datenbankserver so konfiguriert ist, dass er Verbindungen von bei der Windows Domain angemeldeten Benutzern gestattet. Wählen Sie andernfalls je nach Bedarf eine der anderen Optionen aus. In diesem Beispiel wird **Mit SQL Server-Authentifizierung...** verwendet. In diesem Fall müssen Benutzername und Passwort in die entsprechenden Felder eingegeben werden.

© 2016-2022 Altova GmbH

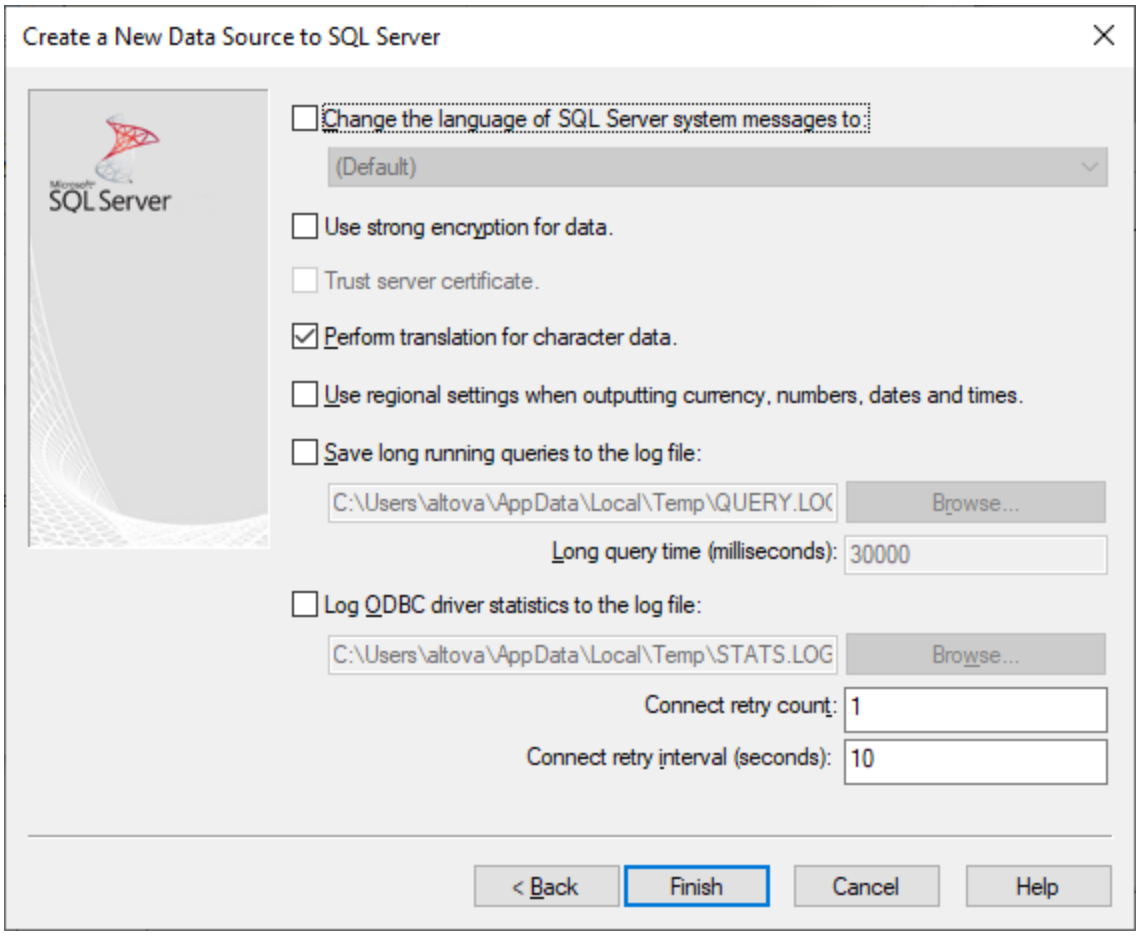


8. Aktivieren Sie optional das Kontrollkästchen **Standarddatenbank ändern in** und geben Sie den Namen der Datenbank, zu der Sie eine Verbindung herstellen (in diesem Beispiel **Sandbox**) ein.

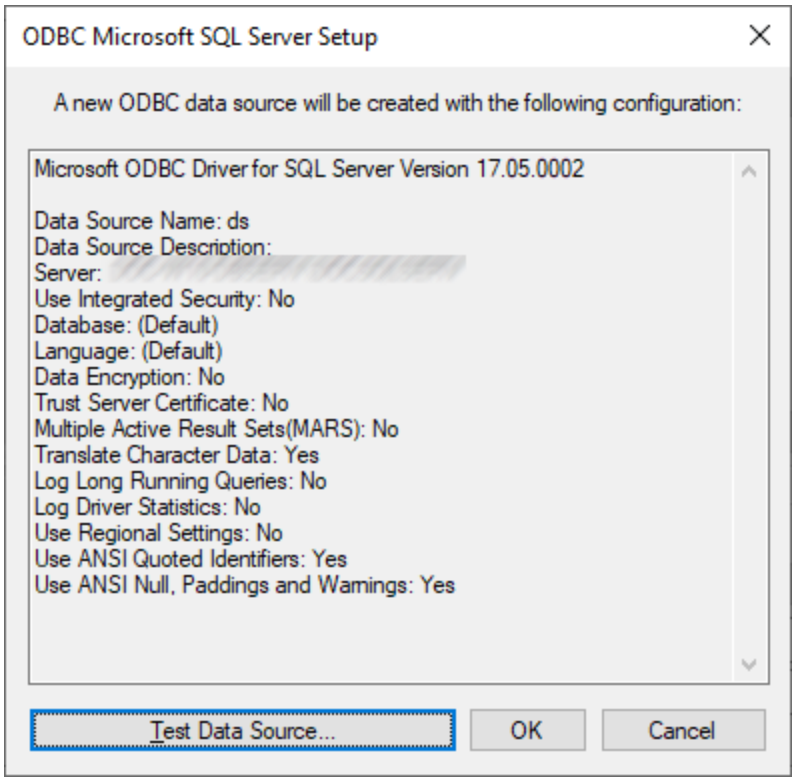


9. Klicken Sie auf **Weiter** und konfigurieren Sie optional weitere Parameter für diese Verbindung.

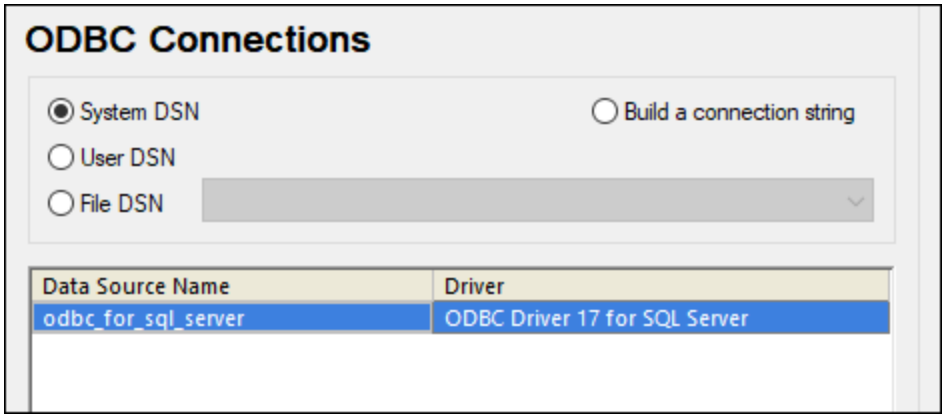
© 2016-2022 Altova GmbH



10. Klicken Sie auf **Fertig stellen**. Daraufhin wird ein Bestätigungsdialogfeld mit den Verbindungsinformationen angezeigt.



11. Klicken Sie auf **OK.** Daraufhin wird die Datenquelle je nach Konfiguration in der Liste der **Benutzer**- oder **System**-Datenquellen angezeigt, z.B:



19.1.10.12 MySQL (ODBC)

In diesem Kapitel wird gezeigt, wie Sie von einem Windows-Rechner aus über den ODBC-Treiber eine Verbindung zu einem MySQL-Datenbankserver herstellen. Der MySQL ODBC-Treiber steht auf Windows nicht zur Verfügung, daher müssen Sie ihn separat herunterladen und installieren. In diesem Beispiel wird die MySQL Connector/ODBC-Version 8.0 verwendet.

© 2016-2022 Altova GmbH

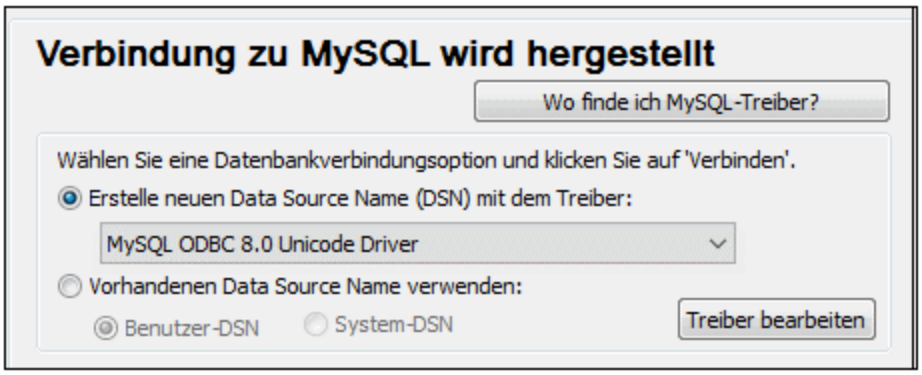
Voraussetzungen:

- Der MySQL ODBC-Treiber muss auf Ihrem Betriebssystem installiert sein. Lesen Sie nach in der Dokumentation zu MySQL, um zu ermittlen, welche Treiberversion für Ihre Datenbankserverversion empfohlen wird (siehe https://dev.mysql.com/downloads/connector/odbc/).
- Sie haben die folgenden Datenbankverbindungsinformationen zur Verfügung: Host, Datenbank, Port, Benutzername und Passwort.

Wenn Sie MySQL Connector/ODBC für die 64-Bit-Plattform installiert haben, stellen Sie sicher, dass auch XMLSpy für die 64-Bit-Plattform installiert ist.

So stellen Sie über ODBC eine Verbindung zu MySQL her:

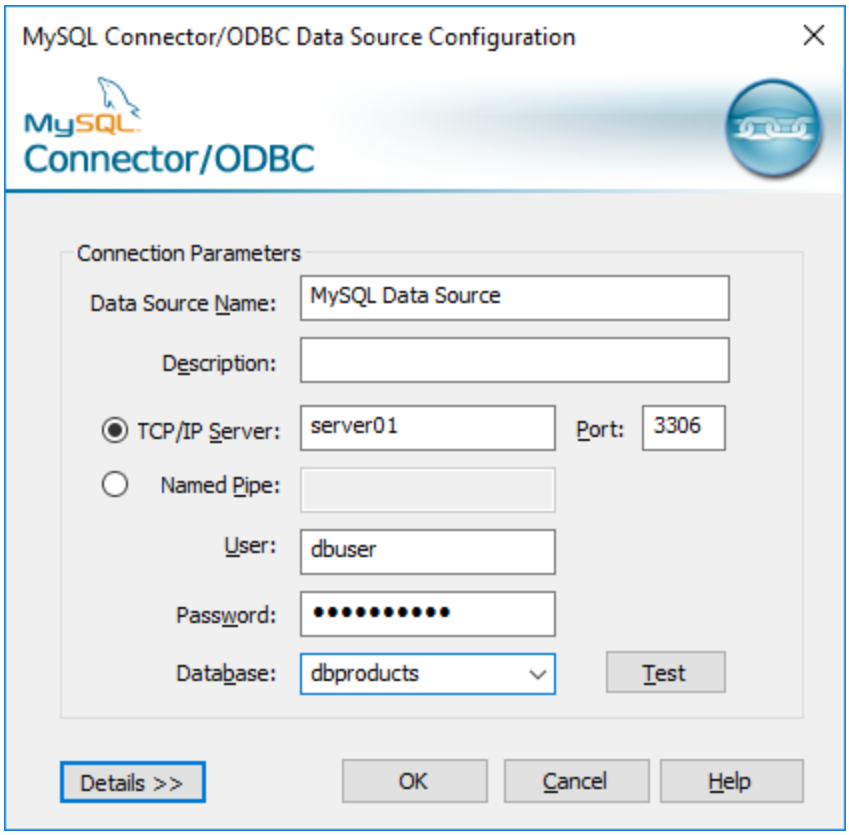
1. Starten Sie den Datenbank-Verbindungsassistenten[970].
2. Wählen Sie **MySQL (ODBC)** und klicken Sie auf **Weiter**.



3. Wählen Sie **Erstelle neuen Data Source Name (DSN) mit dem Treiber** und wählen Sie einen MySQL-Treiber aus. Wenn in der Liste kein MySQL-Treiber verfügbar ist, klicken Sie auf **Treiber bearbeiten** und wählen Sie einen beliebigen verfügbaren MySQL-Treiber aus (die Liste enthält alle auf Ihrem Betriebssystem installierten ODBC-Treiber).

   Wenn Sie XMLSpy 64-Bit installiert haben, werden in der Liste die 64-Bit-ODBC-Treiber angezeigt. Andernfalls werden die 32-Bit-ODBC-Treiber angezeigt. Siehe auch Anzeigen der verfügbaren ODBC-Treiber[991].

4. Klicken Sie auf **Verbinden**.



5. Geben Sie in das Feld "Datenquellenname" einen Namen ein, anhand dessen Sie diese ODBC-Datenquelle in Zukunft identifizieren können.
6. Füllen Sie die Anmeldeinformationen für die Datenbankverbindung aus (TCP/IP Server, Benutzer, Passwort), wählen Sie eine Datenbank aus und klicken Sie auf **OK**.

Anmerkung: Wenn es sich um einen Remote-Datenbankserver handelt, muss er vom Server-Administrator so konfiguriert sein, dass er remote-Verbindungen von der IP-Adresse Ihres Rechners zulässt. Wenn Sie außerdem auf **Details>>** klicken, können Sie eine Reihe zusätzlicher Parameter konfigurieren. Lesen Sie die Dokumentation zum Treiber, bevor Sie die Standardwerte ändern.

19.1.10.13 Oracle (JDBC)

Dieses Kapitel enthält eine Beispielanleitung für das Herstellen einer Verbindung von einem Client-Rechner zu einem Oracle Datenbankserver mittels JDBC. Die Verbindung wird mit Hilfe des auf der Oracle Website verfügbaren **Oracle Instant Client Package (Basic)** als reine Java-Verbindung hergestellt. Der Vorteil dieser Verbindungsart ist, dass nur die Java-Umgebung und die vom Oracle Instant Client Package bereitgestellten .jar-Bibliotheken benötigt werden, sodass Sie keinen komplexeren Datenbank-Client installieren und konfigurieren müssen.

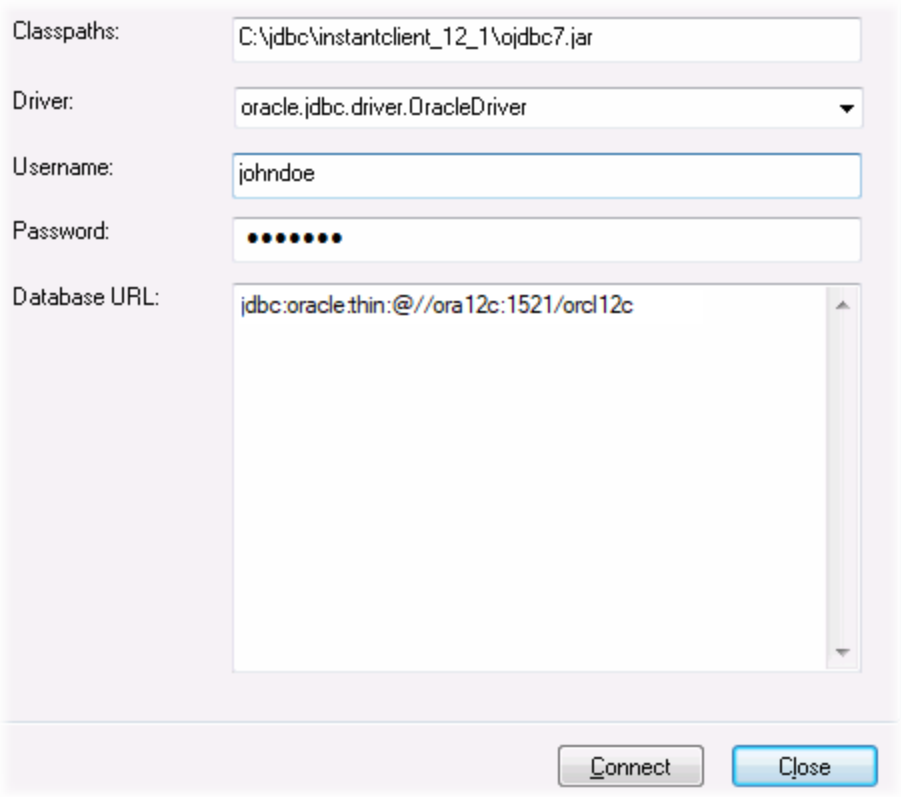
© 2016-2022 Altova GmbH

Voraussetzungen:

- Auf Ihrem Betriebssystem muss JRE (Java Runtime Enviroment) oder Java Development KIT (JDK) installiert sein. Dabei muss es sich entweder um Oracle JDK oder einen Open Source Build wie Oracle OpenJDK handeln.XMLSpy ermittelt den Pfad zur Java Virtual Machine (JVM) anhand der folgenden Ordner und zwar in folgender Reihenfolge: a) anhand des benutzerdefinierten JVM-Pfads, den Sie eventuell in den **Applikationsoptionen** definiert haben; b) anhand des JVM-Pfads in der Windows Registry; c) anhand der `JAVA_HOME`-Umgebungsvariablen.
- Stellen Sie sicher, dass die Plattform von XMLSpy (32-Bit, 64-Bit) mit der des JRE/JDK übereinstimmt.
- **Oracle Instant Client Package (Basic)** muss auf Ihrem Betriebssystem verfügbar sein. Das Paket kann von der offiziellen Oracle Website heruntergeladen werden. In diesem Beispiel wird Oracle Instant Client Package Version 12.1.0.2.0 für Windows 32-Bit und folglich Oracle JDK 32-Bit verwendet.
- Sie haben die folgenden Datenbankverbindungsinformationen zur Verfügung: Host, Port, Servicename, Benutzername und Passwort.

So stellen Sie über das Instant Client Package eine Verbindung zu Oracle her:

1. Starten Sie den Datenbank-Verbindungsassistenten[970].
2. Klicken Sie auf **JDBC-Verbindungen**.
3. Geben Sie neben "Classpaths" den Pfad zur .jar-Datei , die die Verbindung zur Datenbank bereitstellt, ein. Falls nötig, können Sie auch eine durch Semikola getrennte Liste von .jar-Dateipfaden eingeben. Die benötigte .jar-Datei in diesem Beispiel befindet sich unter dem folgenden Pfad: **C:\jdbc\instantclient_12_1\ojdbc7.jar**. Beachten Sie, dass Sie das Textfeld "Classpaths" leer lassen können, wenn Sie den/die .jar-Dateipfad(e) zur Umgebungsvariablen CLASSPATH des Betriebssystems hinzugefügt haben (siehe auch Konfigurieren des CLASSPATH[995]).
4. Wählen Sie im Feld "Treiber" **oracle.jdbc.driver.OracleDriver** oder **oracle.jdbc.driver.OracleDriver** aus. Beachten Sie, dass dieser Eintrag zur Verfügung steht, wenn entweder im Textfeld "Classpath" oder in der Umgebungsvariablen `CLASSPATH` des Betriebssystems eine gültige .jar-Datei gefunden wird (siehe vorheriger Schritt).
5. Geben Sie den Benutzernamen und das Passwort in die entsprechenden Textfelder ein.



6. Geben Sie in das Textfeld "Datenbank-URL" den Connection String zum Datenbankserver ein, indem Sie die hervorgehobenen Werte durch die entsprechenden Werte für Ihren Datenbankserver ersetzen.

```
jdbc:oracle:thin:@//host:port:service
```

7. Klicken Sie auf **Verbinden**.

19.1.10.14 Oracle (ODBC)

In diesem Beispiel wird ein häufig vorkommendes Szenario beschrieben: Sie stellen von XMLSpy aus über einen auf dem lokalen Betriebssystem installierten Oracle Datenbank Client eine Verbindung zu einer Oracle-Datenbank im Netzwerk her.

Das Beispiel enthält eine Anleitung, wie man mit Hilfe des Datenbankverbindungsassistenten in XMLSpy eine ODBC-Datenquelle (DSN) konfiguriert. Wenn Sie bereits einen DSN erstellt haben oder wenn Sie diesen lieber direkt über den ODBC-Datenquellen-Administrator in Windows erstellen, können Sie dies tun und den DSN dann auswählen, sobald Sie vom Assistenten dazu aufgefordert werden. Nähere Informationen zu ODBC-Datenquelle finden Sie unter Einrichten einer ODBC-Verbindung 989.

 © 2016-2022 Altova GmbH

Voraussetzungen:

- Der Oracle Datenbank Client (der den Oracle-ODBC-Treiber enthält) muss auf Ihrem Betriebssystem installiert und konfiguriert sein. Eine Anleitung zum Installieren und Konfigurieren eines Oracle Datenbank Client finden Sie in der Dokumentation zur Oracle-Software.
- Die Datei **tnsnames.ora** im Oracle-Startverzeichnis enthält einen Eintrag, der die Datenbankverbindungsparameter in etwa wie folgt beschreibt:

```
ORCL =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = server01)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SID = orcl)
      (SERVER = DEDICATED)
    )
  )
```

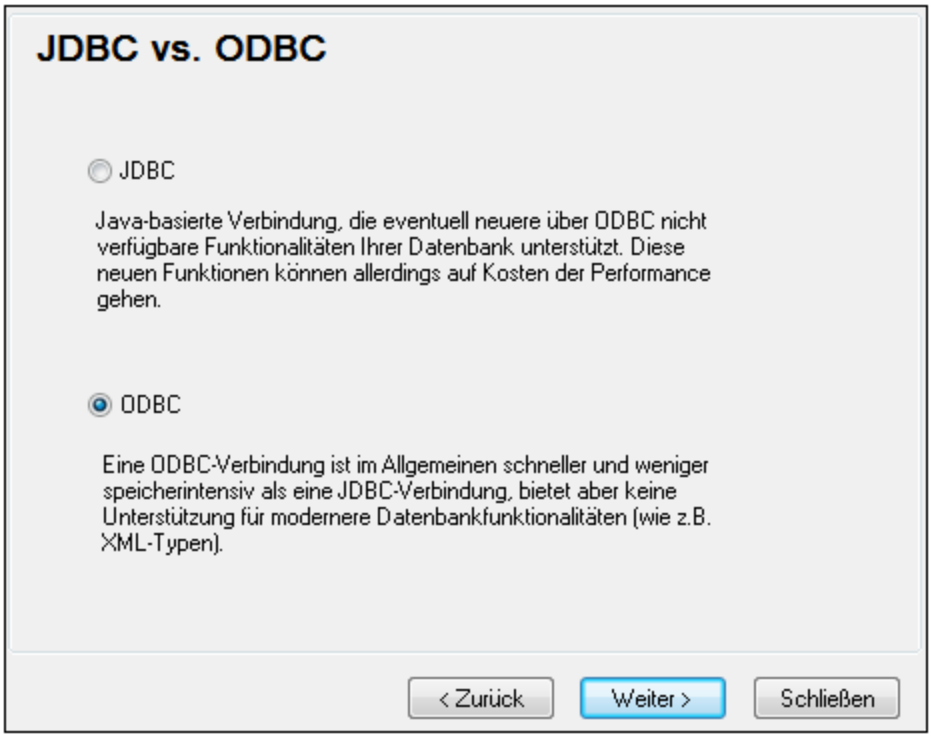
Der Pfad zur Datei **tnsnames.ora** hängt davon ab, wo das Oracle-Startverzeichnis installiert wurde. Beim Oracle-Datenbank-Client 11.2.0 könnte der Standardpfad folgendermaßen lauten:

```
C:\app\username\product\11.2.0\client_1\network\admin\tnsnames.ora
```

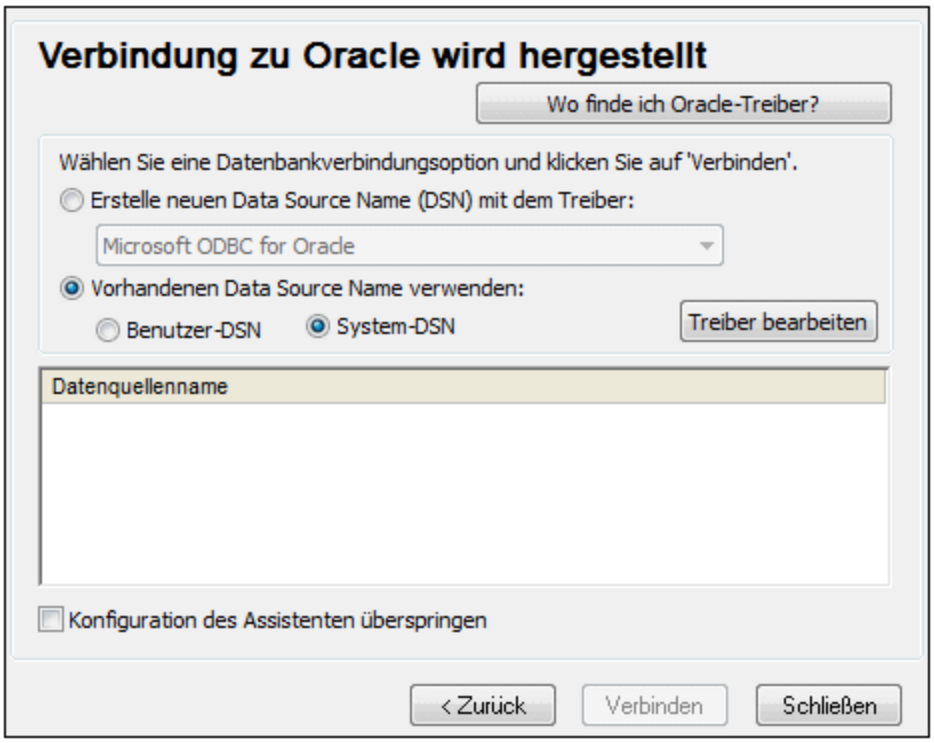
Sie können neue Einträge zur Datei **tnsnames.ora** hinzufügen, indem Sie die Verbindungsdetails entweder hineinkopieren und die Datei speichern oder indem Sie den Oracle *Net-Konfigurationsassistenten* ausführen (falls vorhanden). Wenn diese Werte bei der Konfiguration in Dropdown-Listen aufscheinen sollen, müssen Sie den Pfad zum admin-Ordner eventuell als **TNS_ADMIN**-Umgebungsvariable hinzufügen.

So stellen Sie über ODBC eine Verbindung zu Oracle her:

1. Starten Sie den Datenbank-Verbindungsassistenten 970.
2. Wählen Sie **Oracle (ODBC / JDBC)** und klicken Sie auf **Weiter**.

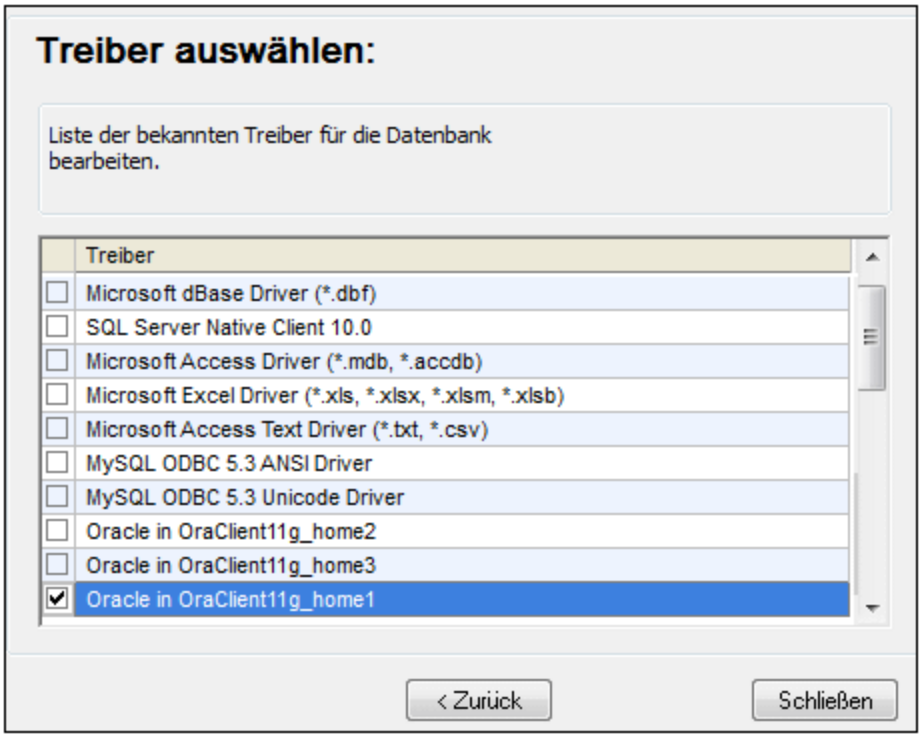


3. Wählen Sie **ODBC**.

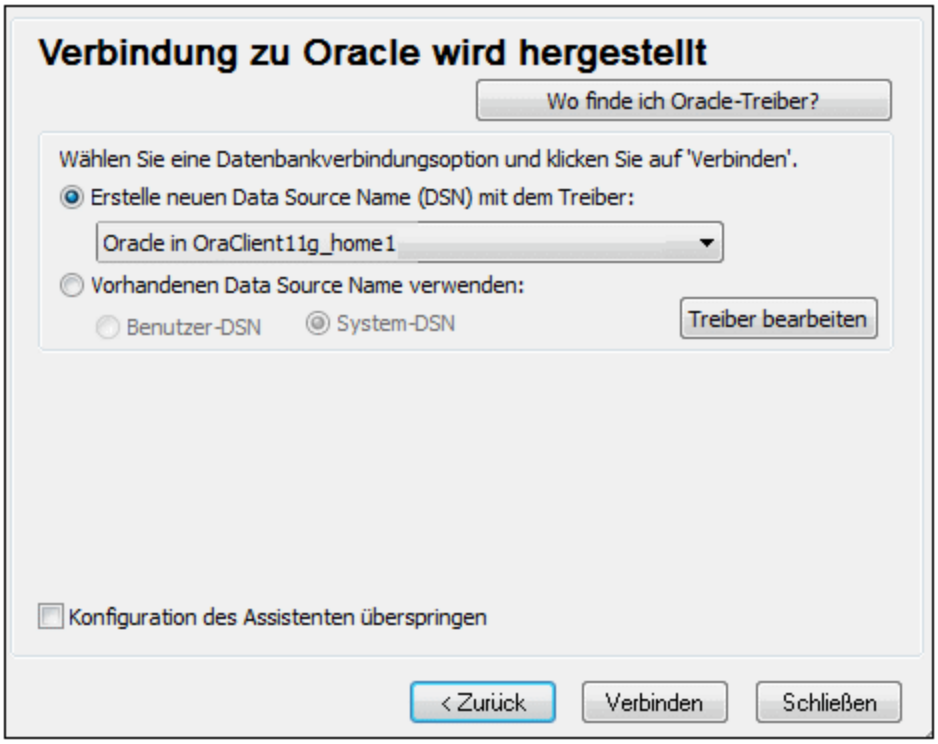


4. Klicken Sie auf **Treiber bearbeiten**.

 © 2016-2022 Altova GmbH



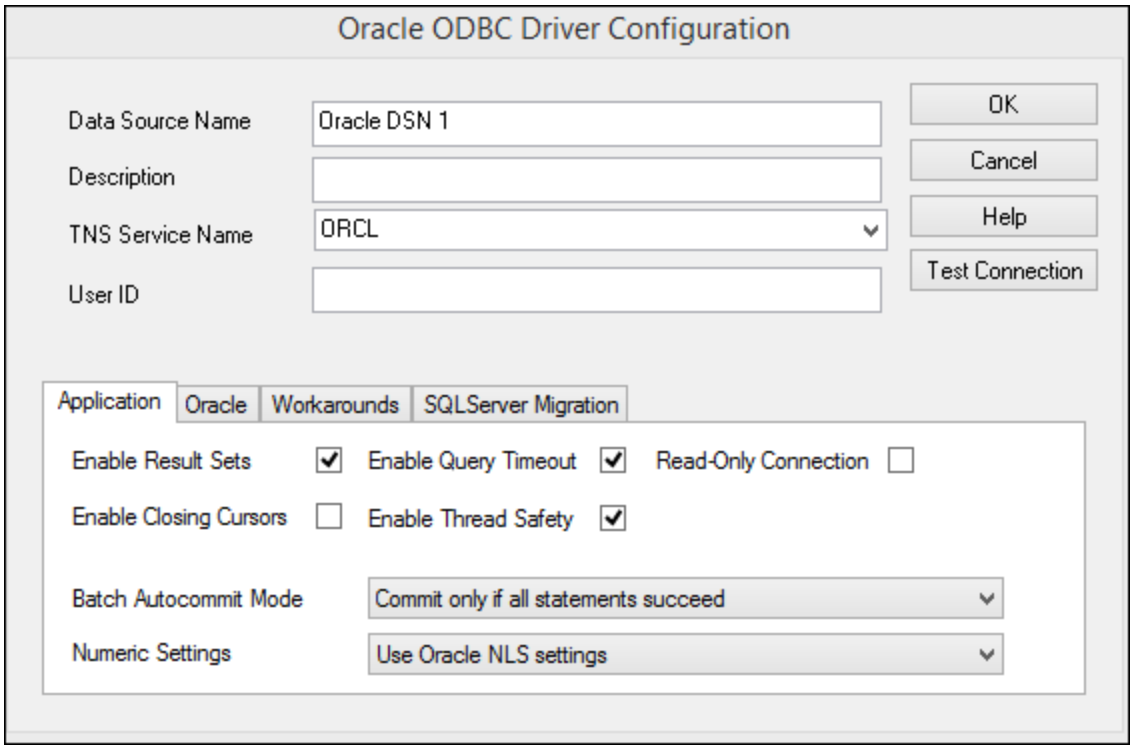
5. Wählen Sie die gewünschten Oracle-Treiber aus (in diesem Beispiel **Oracle in OraClient11g_home1**). In der Liste werden die nach der Installation des Oracle Client auf Ihrem System verfügbaren Oracle-Treiber angezeigt.
6. Klicken Sie auf **Zurück.**
7. Wählen Sie **Erstelle neuen Data Source Name (DSN) mit dem Treiber** und wählen Sie den in Schritt 4 ausgewählten Oracle-Treiber aus.



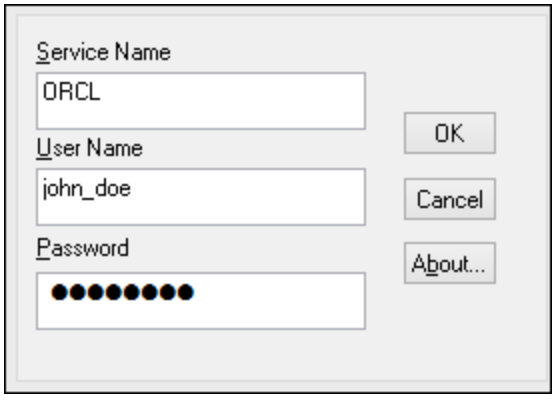
Verwenden Sie den von Microsoft bereitgestellten Treiber **Microsoft ODBC for Oracle** möglichst nicht. Microsoft empfiehlt, den von Oracle bereitgestellten ODBC-Treiber zu verwenden (siehe http://msdn.microsoft.com/en-us/library/ms714756%28v=vs.85%29.aspx)

8. Klicken Sie auf **Verbinden**.

© 2016-2022 Altova GmbH



9. Geben Sie im Textfeld "Datenquellenname" einen Namen für die Datenquelle ein (in diesem Beispiel **Oracle DSN 1**).
10. Geben Sie im Feld "TNS-Dienstname" den in der Datei **tnsnames.ora** definierten Verbindungsnamen ein (siehe Voraussetzungen[1039]). In diesem Beispiel lautet der Verbindungsname **ORCL**. *Anmerkung:* Wenn die Dropdown-Liste der Auswahlliste mit den Werten der **tnsnames.ora**-Datei befüllt werden soll, müssen Sie den Pfad zum admin-Ordner als **TNS_ADMIN**-Umgebungsvariable hinzufügen.
11. Klicken Sie auf **OK**.



12. Geben sie den Benutzernamen und das Passwort für die Datenbank ein und klicken Sie auf OK.

19.1.10.15 PostgreSQL (ODBC)

Dieses Kapitel enthält eine Beispielanleitung für das Herstellen einer Verbindung über den ODBC-Treiber von einem Windows-Rechner zu einem PostgreSQL Datenbankserver. Der PostgreSQL ODBC-Treiber ist auf

Windows nicht verfügbar und muss separat heruntergeladen und installiert werden. In diesem Beispiel wird der von der offiziellen Website heruntergeladene psqlODBC-Treiber (Version 11.0) verwendet (siehe auch Übersicht über Datenbanktreiber[972]).

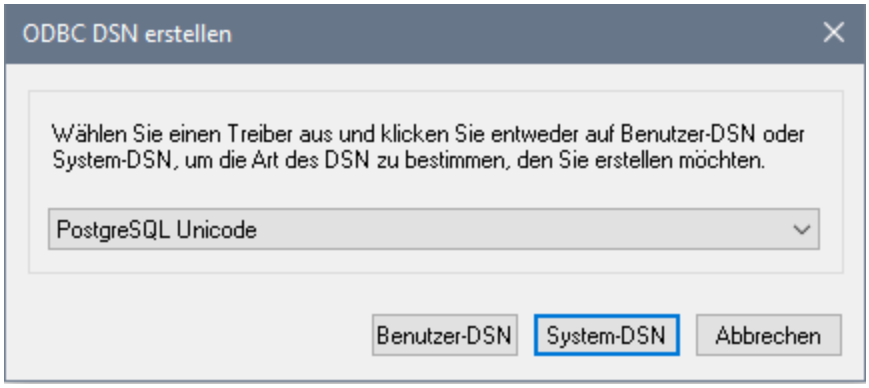
Anmerkung: Sie können die Verbindung zu einer PostgreSQL-Datenbank auch direkt (ohne ODBC-Treiber) herstellen, siehe Einrichten einer PostgreSQL-Verbindung[997].

Voraussetzungen:

- Der *psqlODBC*-Treiber muss auf Ihrem Betriebssystem installiert sein.
- Sie haben die folgenden Datenbankverbindungsinformationen zur Verfügung: Server, Port, Datenbank, Benutzername und Passwort.

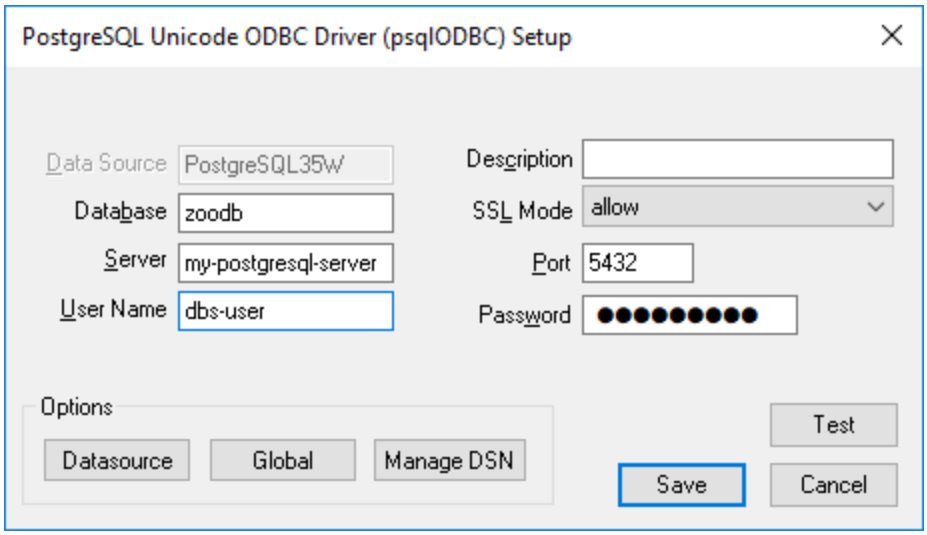
So richten Sie über ODBC eine Verbindung zu PostgreSQL ein:

1. Starten Sie den Datenbank-Verbindungsassistenten[970].
2. Klicken Sie auf **ODBC-Verbindungen**.
3. Aktivieren Sie die Option **Benutzer-DSN**.
4. Klicken Sie auf **Neuen DSN erstellen** und wählen Sie den Treiber aus der Dropdown-Liste aus. Wenn in der Liste kein PostgreSQL-Treiber zur Verfügung steht, stellen Sie sicher, dass der PostgreSQL ODBC-Treiber auf Ihrem Betriebssystem installiert ist, wie in den Voraussetzungen oben erwähnt.



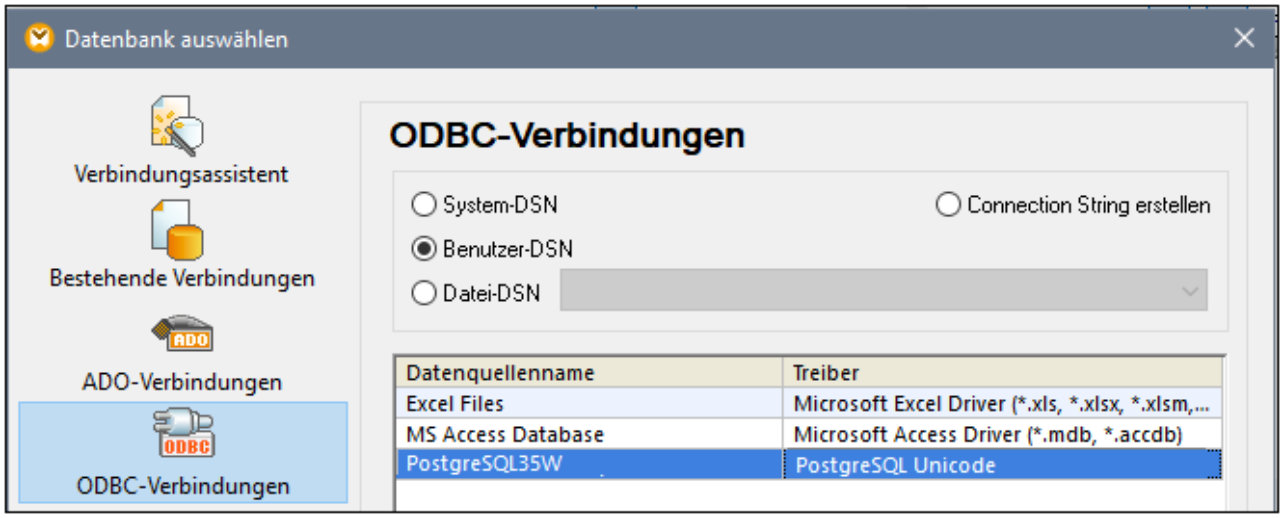
5. Klicken Sie auf **Benutzer-DSN**.

© 2016-2022 Altova GmbH



6. Füllen Sie die Anmeldeinformationen für die Datenbankverbindung aus (diese müssen vom Inhaber der Datenbank bereitgestellt werden) und klicken Sie anschließend auf **Speichern**.

Die Verbindung steht nun in der Liste der ODBC-Verbindungen zur Verfügung. Um eine Verbindung zur Datenbank herzustellen, können Sie entweder auf die Verbindung doppelklicken oder die Verbindung auswählen und auf **Verbinden** klicken.



19.1.10.16 Progress OpenEdge (JDBC)

Dieses Kapitel enthält eine Beispielanleitung für das Herstellen einer Verbindung zu einem Progress OpenEdge 11.6-Datenbankserver mittels JDBC.

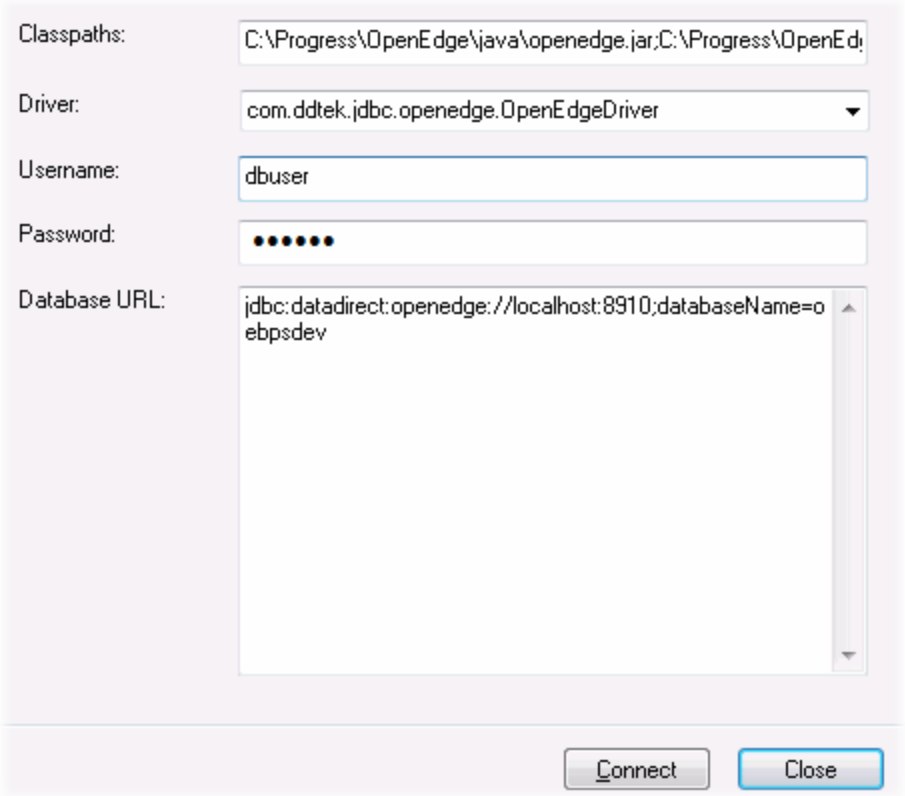
Voraussetzungen:

- Auf Ihrem Betriebssystem muss JRE (Java Runtime Enviroment) oder Java Development KIT (JDK) installiert sein. Dabei muss es sich entweder um Oracle JDK oder einen Open Source Build wie Oracle OpenJDK handeln.XMLSpy ermittelt den Pfad zur Java Virtual Machine (JVM) anhand der folgenden Ordner und zwar in folgender Reihenfolge: a) anhand des benutzerdefinierten JVM-Pfads, den Sie eventuell in den **Applikationsoptionen** definiert haben; b) anhand des JVM-Pfads in der Windows Registry; c) anhand der `JAVA_HOME`-Umgebungsvariablen.
- Stellen Sie sicher, dass die Plattform von XMLSpy (32-Bit, 64-Bit) mit der des JRE/JDK übereinstimmt.
- Die `PATH`-Umgebungsvariable des Betriebssystems muss den Pfad zum `bin`-Verzeichnis des JRE- bzw. JDK-Installationsverzeichnisses enthalten, z.B. `C:\Programe (x86)\Java\jre1.8.0_51\bin`.
- Der Progress OpenEdge JDBC-Treiber muss auf Ihrem Betriebssystem installiert sein. In diesem Beispiel erfolgt die JDBC-Verbindung über die Treiberkomponentendateien **openedge.jar** und **pool.jar,** die als Teil der OpenEdge SDK-Installation unter **C:\Progress\OpenEdge\java** zur Verfügung stehen.
- Sie haben die folgenden Datenbankverbindungsinformationen zur Verfügung: Host, Port, Datenbankname, Benutzername und Passwort.

So stellen Sie über JDBC eine Verbindung zu OpenEdge her:

1. Starten Sie den Datenbank-Verbindungsassistenten[970].
2. Klicken Sie auf **JDBC-Verbindungen**.
3. Geben Sie neben "Classpaths" den Pfad zur .jar-Datei , die die Verbindung zur Datenbank bereitstellt, ein. Falls nötig, können Sie auch eine durch Semikola getrennte Liste von .jar-Dateipfaden eingeben. Die benötigte .jar-Datei in diesem Beispiel befindet sich unter dem folgenden Pfad: **`C:\Progress\OpenEdge\java\openedge.jar;C:\Progress\OpenEdge\java\pool.jar;`**. Beachten Sie, dass Sie das Textfeld "Classpaths" leer lassen können, wenn Sie den/die .jar-Dateipfad(e) zur Umgebungsvariablen CLASSPATH des Betriebssystems hinzugefügt haben (siehe auch Konfigurieren des CLASSPATH[995]).
4. Wählen Sie im Feld "Treiber" **com.ddtek.jdbc.openedge.OpenEdgeDriver** aus. Beachten Sie, dass dieser Eintrag zur Verfügung steht, wenn entweder im Textfeld "Classpaths" oder in der Umgebungsvariablen `CLASSPATH` des Betriebssystems eine gültige .jar-Datei gefunden wird (siehe vorheriger Schritt).

© 2016-2022 Altova GmbH



5. Geben Sie den Benutzernamen und das Passwort für die Datenbank in die entsprechenden Textfelder ein.
6. Geben Sie in das Textfeld "Datenbank-URL" den Connection String zum Datenbankserver ein, indem Sie die hervorgehobenen Werte durch die entsprechenden Werte für Ihren Datenbankserver ersetzen.

```
jdbc:datadirect:openedge://host:port;databaseName=db_name
```

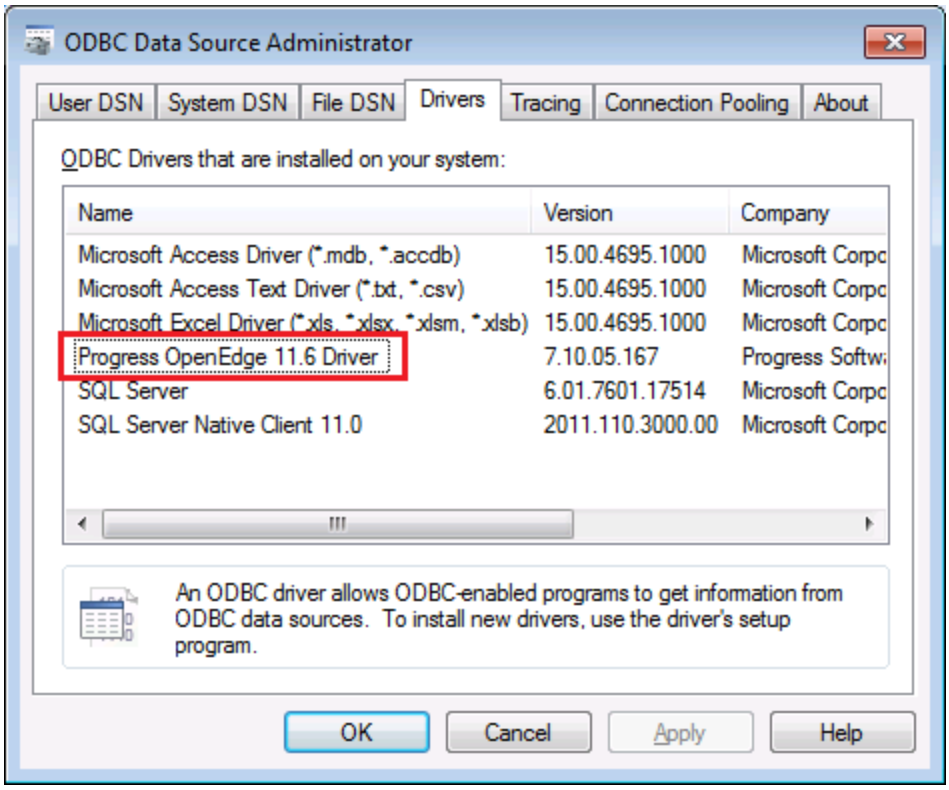
7. Klicken Sie auf **Verbinden**.

19.1.10.17 Progress OpenEdge (ODBC)

Dieses Kapitel enthält eine Beispielanleitung für das Herstellen einer Verbindung zu einer Progress OpenEdge-Datenbank über den Progress OpenEdge 11.6 ODBC-Treiber.

Voraussetzungen:

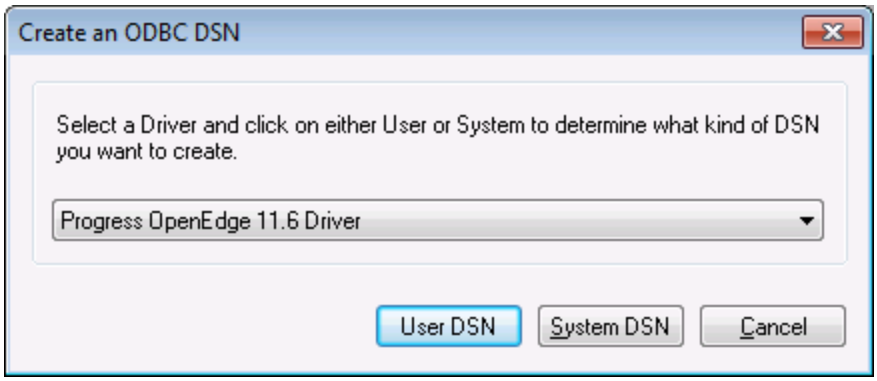
- Der *ODBC Connector for Progress OpenEdge*-Treiber muss auf Ihrem Betriebssystem installiert sein. Der Progress OpenEdge ODBC-Treiber kann von der Website des Anbieters heruntergeladen werden (siehe auch Übersicht über Datenbanktreiber[972]). Bei Ausführung der 32-Bit-Version von XMLSpy muss der 32-Bit-Treiber und bei Ausführung der 64-Bit-Version der 64-Bit-Treiber heruntergeladen werden. Überprüfen Sie nach Abschluss der Installation, ob der ODBC-Treiber auf Ihrem Rechner zur Verfügung steht (siehe auch Anzeigen der verfügbaren ODBC-Treiber[991]).



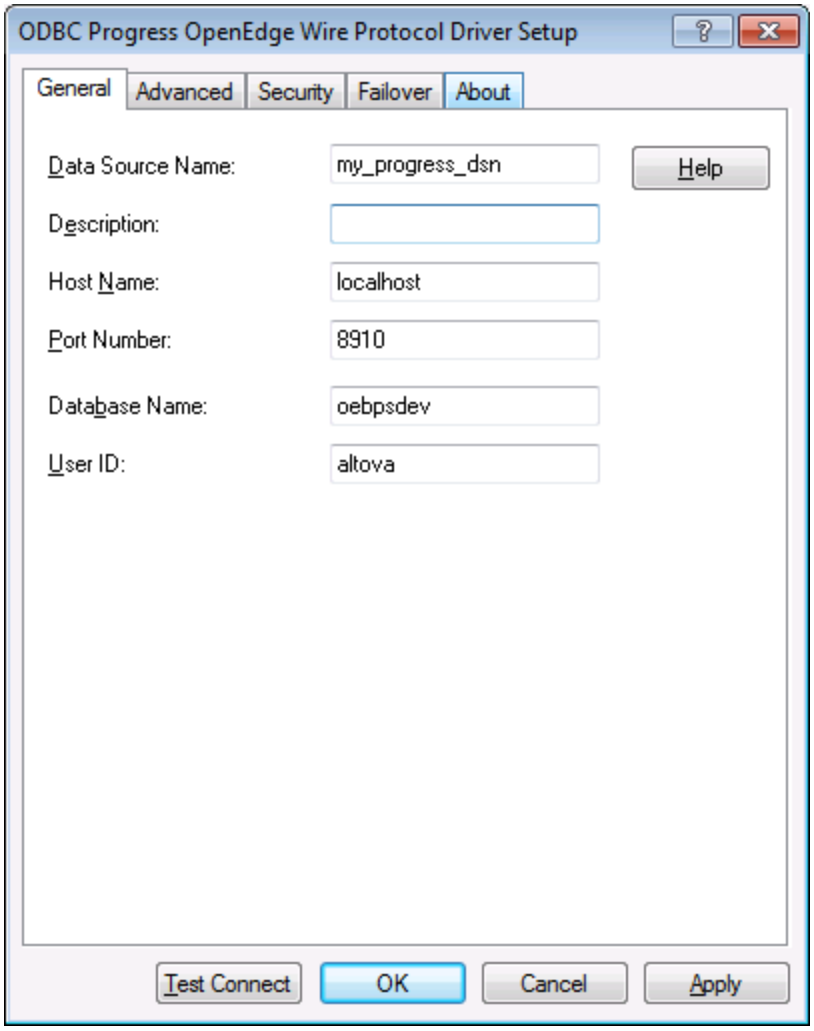
- Sie verfügen über die folgenden Datenbankverbindungsinformationen: Host-Name, Datenbankname, Benutzer-ID und Passwort.

So stellen Sie über ODBC eine Verbindung zu Progress OpenEdge her:

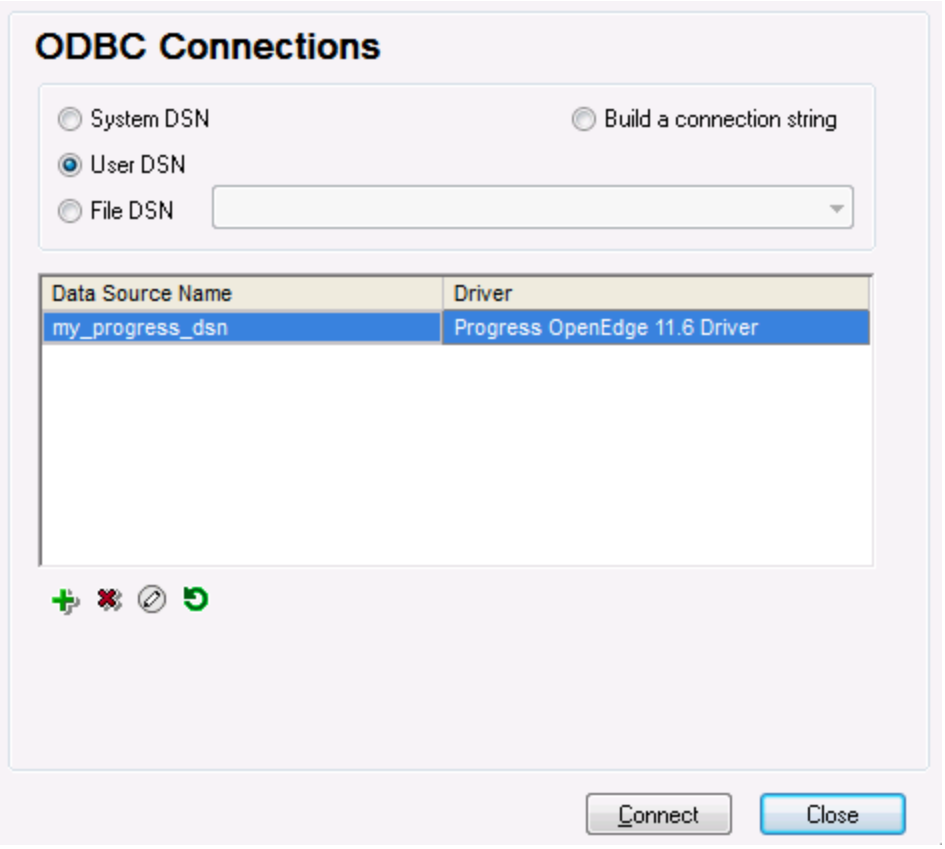
1. Starten Sie den Datenbank-Verbindungsassistenten 970.
2. Klicken Sie auf **ODBC-Verbindungen**.
3. Wählen Sie Benutzer-DSN (oder alternativ dazu **System-DSN** oder **Datei-DSN** - in diesem Fall sind die nun folgenden Schritte ähnlich).
4. Klicken Sie auf **Hinzufügen** .
2. Wählen Sie aus der Liste **Progress OpenEdge Driver** aus und klicken Sie auf **Benutzer-DSN** (oder gegebenenfalls auf **System-DSN**).



3. Füllen Sie die Anmeldeinformationen für die Datenbankverbindung aus (Datenbank, Server, Port, Benutzername und Passwort) und klicken Sie auf **OK**. Um die Verbindung zu überprüfen, bevor Sie die eingegebenen Daten speichern, klicken Sie auf **Verbindung testen**.



4. Klicken Sie auf OK. Die neue Datenquelle wird nun in der Liste der ODBC-Datenquellen angezeigt.



5. Klicken Sie auf **Verbinden**.

19.1.10.18 Sybase (JDBC)

Dieses Kapitel enthält eine Beispielanleitung, wie Sie über JDBC eine Verbindung zu einer Sybase-Datenbank herstellen.

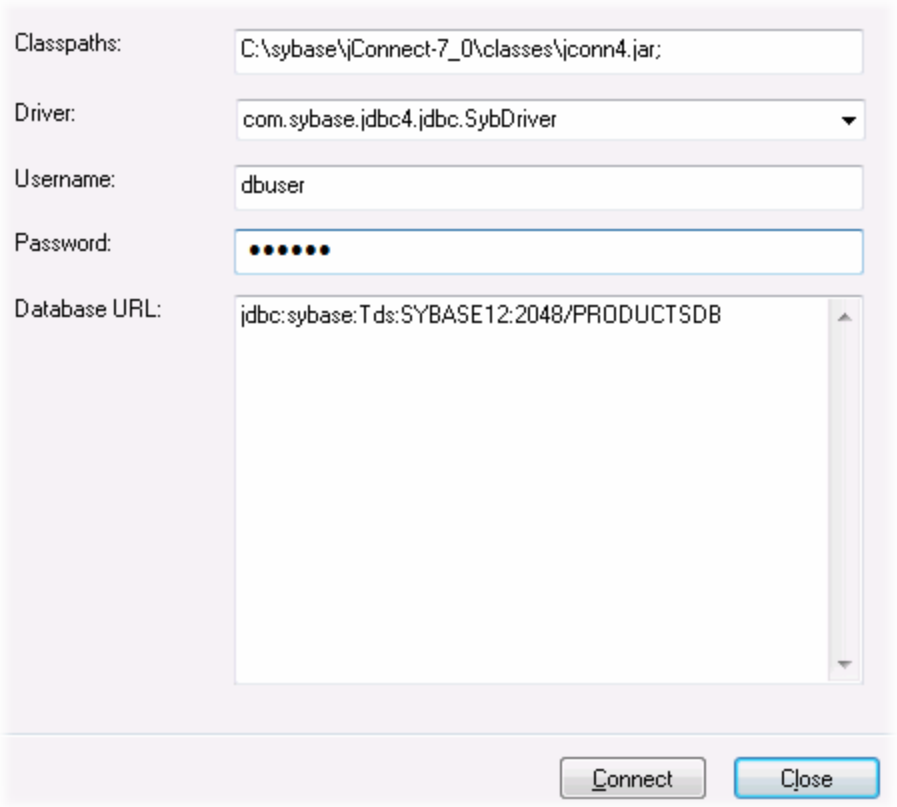
Voraussetzungen:

- Auf Ihrem Betriebssystem muss JRE (Java Runtime Enviroment) oder Java Development KIT (JDK) installiert sein. Dabei muss es sich entweder um Oracle JDK oder einen Open Source Build wie Oracle OpenJDK handeln.XMLSpy ermittelt den Pfad zur Java Virtual Machine (JVM) anhand der folgenden Ordner und zwar in folgender Reihenfolge: a) anhand des benutzerdefinierten JVM-Pfads, den Sie eventuell in den **Applikationsoptionen** definiert haben; b) anhand des JVM-Pfads in der Windows Registry; c) anhand der `JAVA_HOME`-Umgebungsvariablen.
- Stellen Sie sicher, dass die Plattform von XMLSpy (32-Bit, 64-Bit) mit der des JRE/JDK übereinstimmt.
- Auf Ihrem Betriebssystem muss die Sybase *jConnect*-Komponente installiert sein (in diesem Beispiel wird *jConnect 7.0* verwendet. Sie wird als Teil des *Sybase Adaptive Server Enterprise PC Client* installiert). Eine Anleitung zur Installation des Datenbank Client finden Sie in der Sybase-Dokumentation.
- Sie haben die folgenden Datenbankverbindungsinformationen zur Verfügung: Host, Port, Datenbankname, Benutzername und Passwort.

 © 2016-2022 Altova GmbH

So stellen Sie über JDBC eine Verbindung zu Sybase her:

1. Starten Sie den Datenbank-Verbindungsassistenten[970].
2. Klicken Sie auf **JDBC-Verbindungen**.
3. Geben Sie neben "Classpaths" den Pfad zur .jar-Datei , die die Verbindung zur Datenbank bereitstellt, ein. Falls nötig, können Sie auch eine durch Semikola getrennte Liste von .jar-Dateipfaden eingeben. Die benötigte .jar-Datei in diesem Beispiel befindet sich unter dem folgenden Pfad: **C:\sybase\jConnect-7_0\classes\jconn4.jar**. Beachten Sie, dass Sie das Textfeld "Classpaths" leer lassen können, wenn Sie den/die .jar-Dateipfad(e) zur Umgebungsvariablen CLASSPATH des Betriebssystems hinzugefügt haben (siehe auch Konfigurieren des CLASSPATH[995]).
4. Wählen Sie aus der Liste der verfügbaren JDBC-Treiber den Sybase JDBC-Treiber aus (in diesem Beispiel **com.sybase.jdbc4.jdbc.SybDriver**). Beachten Sie, dass dieser Eintrag zur Verfügung steht, wenn entweder im Textfeld "Classpaths" oder in der Umgebungsvariablen `CLASSPATH` des Betriebssystems eine gültige .jar-Datei gefunden wird (siehe vorheriger Schritt).



5. Geben Sie den Benutzernamen und das Passwort in die entsprechenden Textfelder ein.
6. Geben Sie in das Textfeld "Datenbank-URL" den Connection String zum Datenbankserver ein, indem Sie die hervorgehobenen Werte durch die entsprechenden Werte für Ihren Datenbankserver ersetzen.

```
jdbc:sybase:Tds:hostName:port/databaseName
```

7. Klicken Sie auf **Verbinden**.

19.1.10.19 Teradata (JDBC)

In diesem Beispiel wird gezeigt, wie Sie über JDBC eine Verbindung zu einem Teradata-Datenbankserver herstellen.

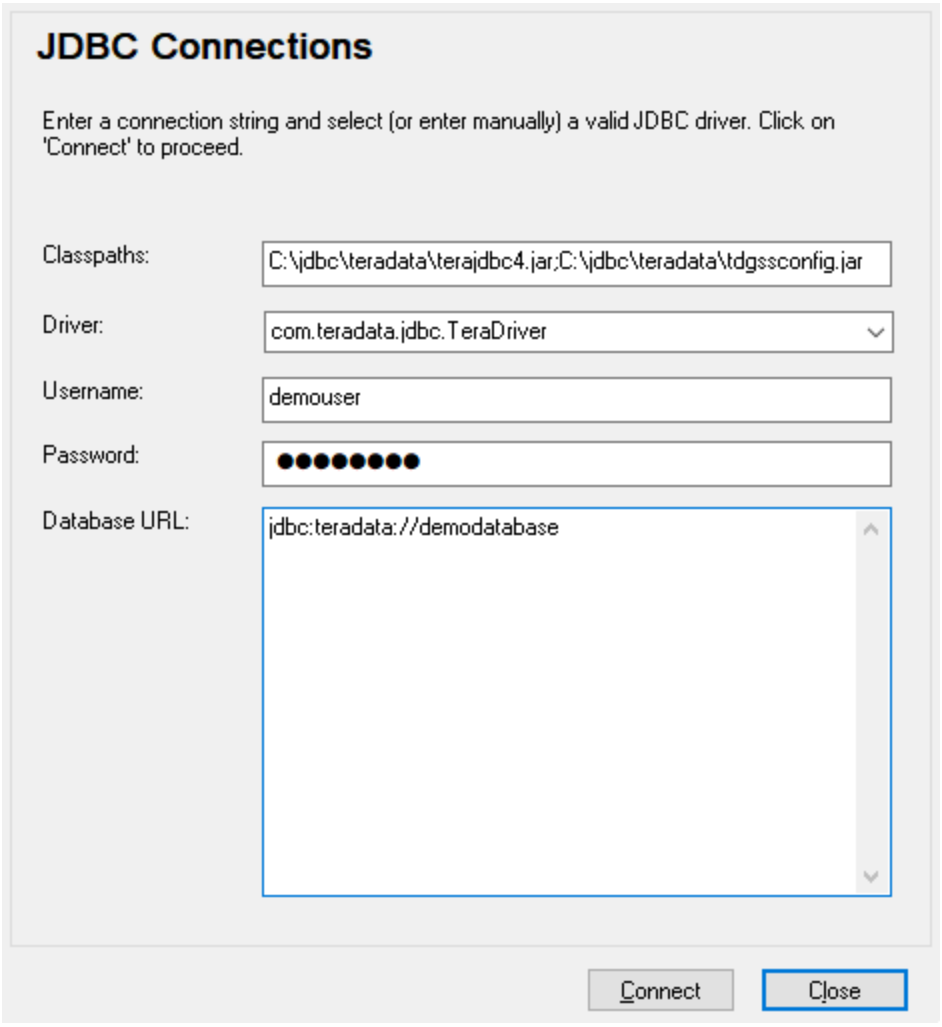
Voraussetzungen:

- Auf Ihrem Betriebssystem muss JRE (Java Runtime Enviroment) oder Java Development KIT (JDK) installiert sein. Dabei muss es sich entweder um Oracle JDK oder einen Open Source Build wie Oracle OpenJDK handeln.XMLSpy ermittelt den Pfad zur Java Virtual Machine (JVM) anhand der folgenden Ordner und zwar in folgender Reihenfolge: a) anhand des benutzerdefinierten JVM-Pfads, den Sie eventuell in den **Applikationsoptionen** definiert haben; b) anhand des JVM-Pfads in der Windows Registry; c) anhand der `JAVA_HOME`-Umgebungsvariablen.
- Stellen Sie sicher, dass die Plattform von XMLSpy (32-Bit, 64-Bit) mit der des JRE/JDK übereinstimmt.
- Der JDBC-Treiber (eine oder mehrere .jar-Dateien, die die Verbindung zur Datenbank herstellen) muss auf Ihrem Betriebssystem installiert sein. In diesem Beispiel wird der Teradata JDBC-Treiber 16.20.00.02 verwendet. Nähere Informationen dazu finden Sie unter https://downloads.teradata.com/download/connectivity/jdbc-driver.
- Sie haben die folgenden Datenbankinformationen zur Verfügung: Host, Datenbank, Port, Benutzername und Passwort.

So stellen Sie über JDBC eine Verbindung zu Teradata her:

1. Starten Sie den Datenbank-Verbindungsassistenten[970].
2. Klicken Sie auf **JDBC-Verbindungen**.
3. Geben Sie neben "Classpaths" den Pfad zur .jar-Datei , die die Verbindung zur Datenbank bereitstellt, ein. Falls nötig, können Sie auch eine durch Semikola getrennte Liste von .jar-Dateipfaden eingeben. Die benötigte .jar-Datei in diesem Beispiel befindet sich unter dem folgenden Pfad:**C:\jdbc\teradata**. Beachten Sie, dass Sie das Textfeld "Classpaths" leer lassen können, wenn Sie den/die .jar-Dateipfad(e) zur Umgebungsvariablen CLASSPATH des Betriebssystems hinzugefügt haben (siehe auch Konfigurieren des CLASSPATH[995]).
4. Wählen Sie im Feld "Treiber" **com.teradata.jdbc.TeraDriver** aus. Beachten Sie, dass dieser Eintrag zur Verfügung steht, wenn entweder im Textfeld "Classpath" oder in der Umgebungsvariablen `CLASSPATH` des Betriebssystems eine gültige .jar-Datei gefunden wird (siehe vorheriger Schritt).

© 2016-2022 Altova GmbH



5. Geben Sie den Benutzernamen und das Passwort in die entsprechenden Textfelder ein.
6. Geben Sie in das Textfeld "Datenbank-URL" den Connection String zum Datenbankserver ein, indem Sie die hervorgehobenen Werte durch die entsprechenden Werte für Ihren Datenbankserver ersetzen.

```
jdbc:teradata://databaseServerName
```

7. Klicken Sie auf **Verbinden**.

19.1.10.20 Teradata (ODBC)

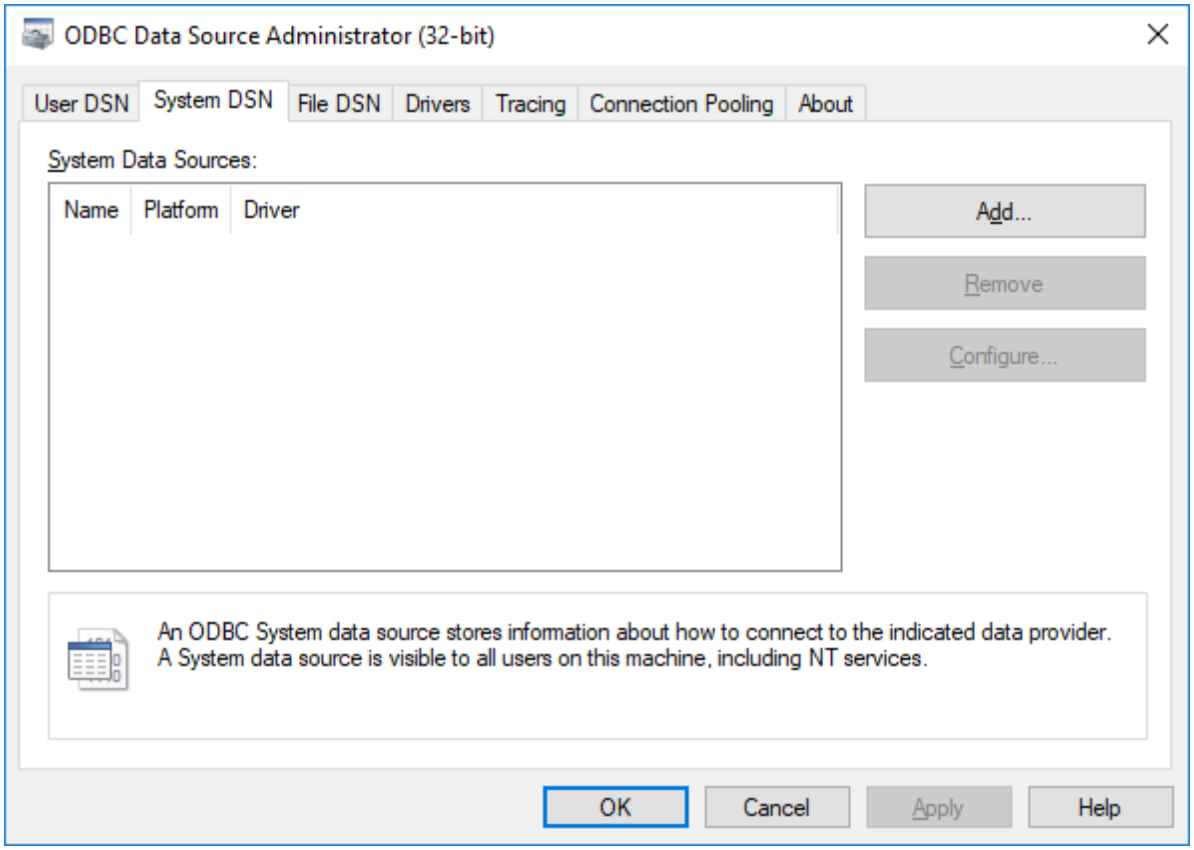
In diesem Beispiel wird gezeigt, wie Sie über ODBC eine Verbindung zu einem Teradata-Datenbankserver herstellen.

Voraussetzungen:

- Der Teradata ODBC-Treiber (https://downloads.teradata.com/download/connectivity/odbc-driver/windows) muss installiert sein. In diesem Beispiel wird der Teradata ODBC-Treiber für Windows Version 16.20.00 verwendet.
- Sie haben die folgenden Datenbankinformationen zur Verfügung: Host, Benutzername und Passwort.

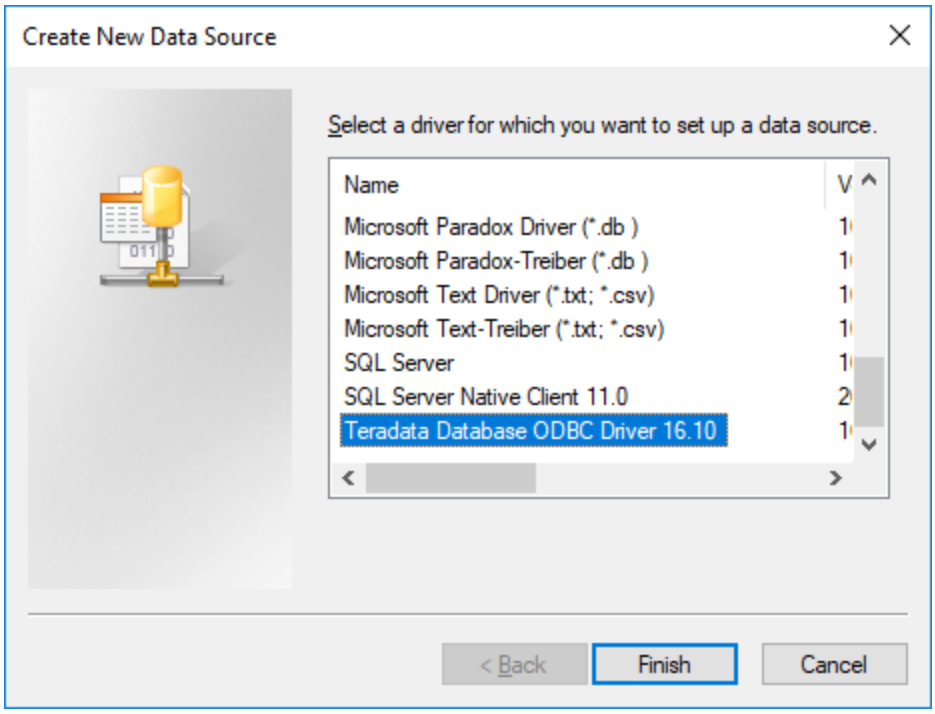
So stellen Sie über ODBC eine Verbindung zu Teradata her:

1. Drücken Sie die **Windows**-Taste, beginnen Sie mit der Eingabe von "ODBC" und wählen Sie aus der Liste der Vorschläge **ODBC Datenquellen einrichten (32-Bit)** . Wenn Sie einen 64-Bit-ODBC-Treiber haben, wählen Sie **ODBC Datenquellen einrichten (64-Bit)** und verwenden Sie in den nachfolgenden Schritten die 64-Bit-Version von XMLSpy.

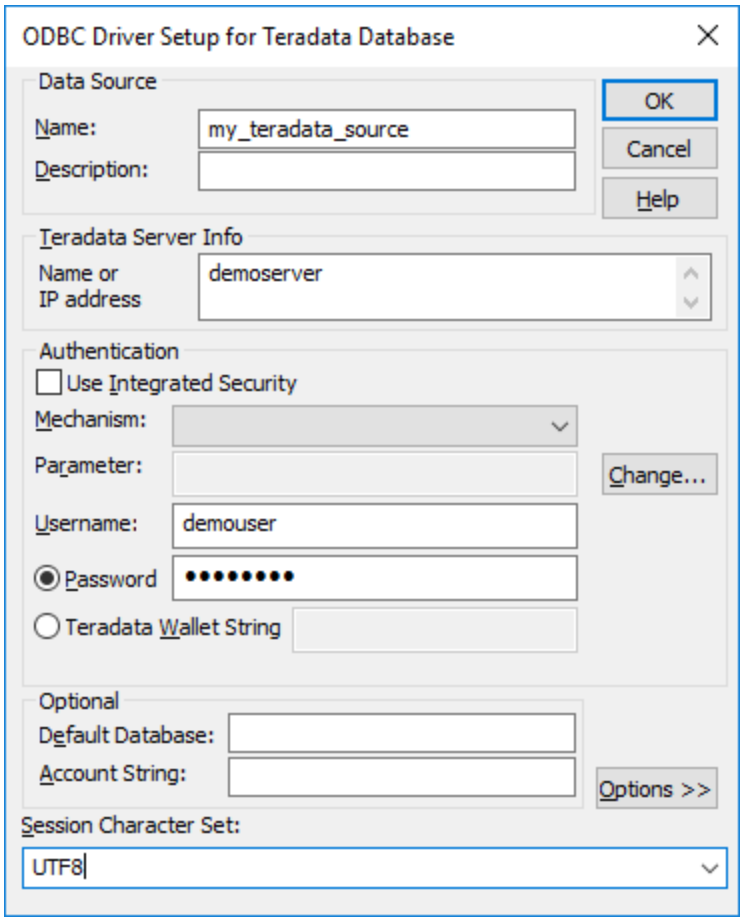


2. Klicken Sie auf das Register **System DSN** und anschließend auf **Hinzufügen**.

© 2016-2022 Altova GmbH

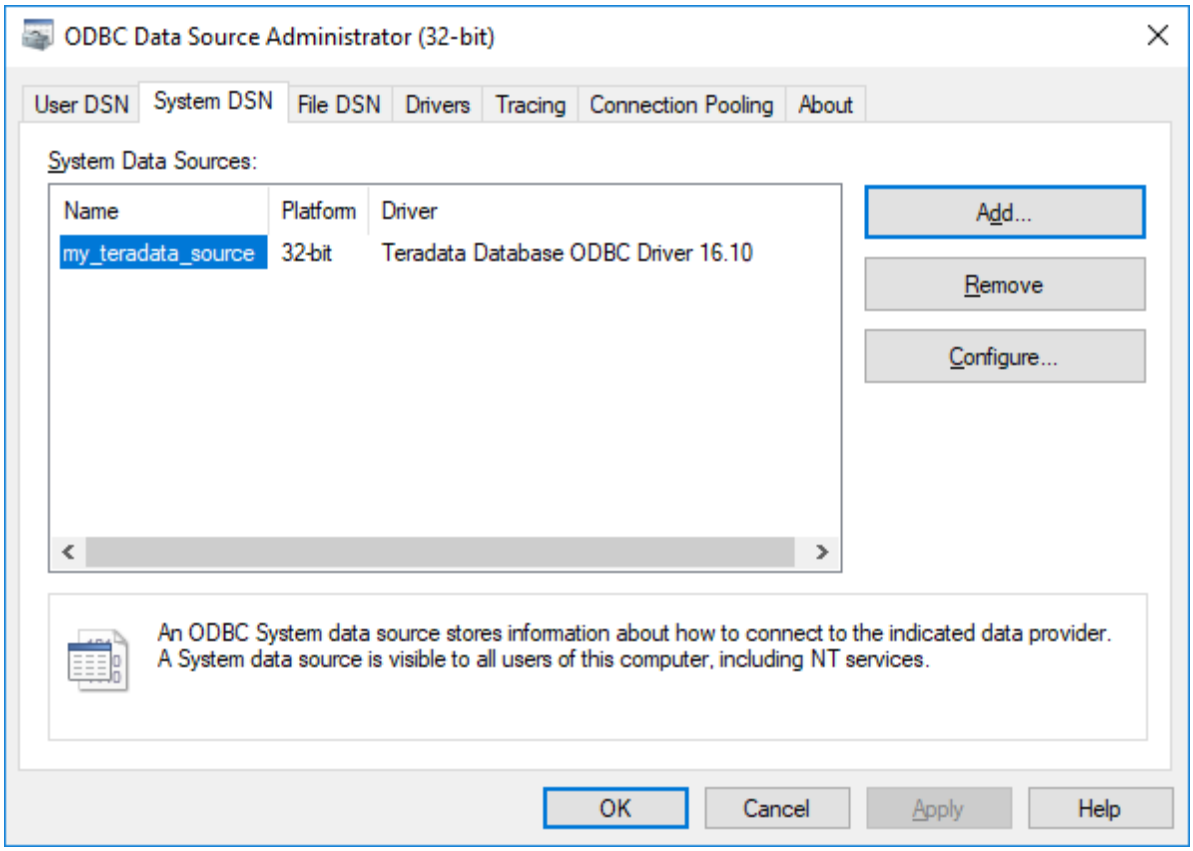


3. Wählen Sie **Teradata Database ODBC Driver** aus und klicken Sie auf **Fertig stellen**.



4. Geben Sie einen Namen und optional eine Beschreibung für diese ODBC-Datenquelle ein. Geben Sie außerdem die Anmeldeinformationen für die Datenbankverbindung ein (Datenbankserver, Benutzer, Passwort) und wählen Sie optional eine Datenbank aus.
5. Klicken Sie auf **OK**. Die Datenquelle wird nun in der Liste angezeigt.

© 2016-2022 Altova GmbH



6. Starten Sie XMLSpy und anschließend den Datenbankverbindungsassistenten[970].
7. Klicken Sie auf **ODBC-Verbindungen**.



8. Klicken Sie auf **System DSN,** wählen Sie die zuvor erstellte Datenquelle aus und klicken Sie auf **Verbinden**.

Anmerkung: Wenn Sie die folgende Fehlermeldung erhalten: "Der Treiber hat eine ungültige SQL_DRIVER_ODBC_VER: 03.80 zurückgegeben (oder ...konnte nicht zurückgegeben werden)", überprüfen Sie, ob der Pfad zum ODBC-Client (z.B. **C:\Programme\Teradata\Client\16.10\bin**, falls unter diesem Pfad installiert) in der PATH-Umgebungsvariablen Ihres Systems vorhanden ist. Fügen Sie den Pfad manuell hinzu, falls der Pfad fehlt.

© 2016-2022 Altova GmbH

19.2 Unterstützte Datenbanken

Die nachstehende Tabelle enthält eine Liste aller unterstützten Datenbanken. Altova ist bestrebt, auch Unterstützung für andere Datenbanken zu gewährleisten, doch wurde die erfolgreiche Herstellung der Verbindung und Verarbeitung der Daten nur für die aufgelisteten Datenbanken getestet. Wenn es sich bei Ihrer Altova-Applikation um eine 64-Bit-Version handelt, stellen Sie sicher, dass Sie Zugriff auf die 64-Bit-Datenbanktreiber haben, die für die entsprechenden Datenbank benötigt werden.

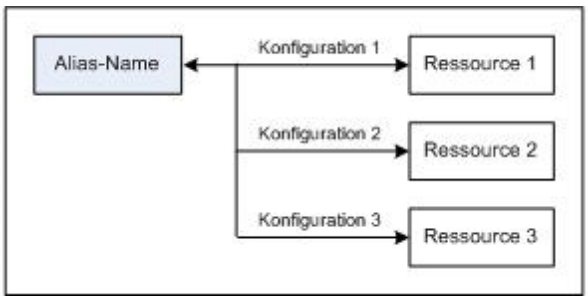
Datenbank	Anmerkungen
Firebird 2.x, 3.x	
IBM DB2 8.x, 9.x, 10.x, 11.x	
IBM Db2 for i 6.x, 7.4	Logische Dateien werden unterstützt und als Ansichten angezeigt.
IBM Informix 11.70 und höher	
MariaDB 10 und höher	
Microsoft Access 2003 und höher	Zum Zeitpunkt der Verfassung dieser Dokumentation (Anfang September 2019) gibt es kein Microsoft Access Runtime für Access 2019. Sie können nur dann eine Verbindung von Altova-Produkten zu einer Access 2019-Datenbank herstellen, wenn Microsoft Access 2016 Runtime installiert ist und der Datentyp "Large Number" in der Datenbank nicht verwendet wird.
Microsoft Azure SQL-Datenbank	SQL Server 2016 Codebase
Microsoft SQL Server 2005 und höher Microsoft SQL Server unter Linux	
MySQL 5 und höher	
Oracle 9i und höher	
PostgreSQL 8 und höher	PostgreSQL-Verbindungen werden sowohl als native Verbindungen als auch als treiberbasierte Verbindungen über Schnittstellen (Treiber) wie ODBC oder JDBC unterstützt. Für native Verbindungen werden keine Treiber benötigt.
Progress OpenEdge 11.6	
SQLite 3.x	SQLite-Verbindungen werden als native, direkte Verbindungen zur SQLite-Datenbankdatei unterstützt. Es sind keine separaten Treiber erforderlich. Daten, die aus einer SQLite-Datenbank stammen, können in der Authentic-Ansicht nicht bearbeitet werden. Wenn Sie versuchen, SQLite-Daten über die Authentic-Ansicht zu speichern, erscheint eine entsprechende Meldung dazu.

Datenbank	Anmerkungen
Sybase ASE15, 16	
Teradata 16	

© 2016-2022 Altova GmbH

20 Globale Altova-Ressourcen

Bei den globalen Altova-Ressourcen handelt es sich um eine Sammlung von Aliassen für Datei-, Ordner- und Datenbankressourcen. Jeder Alias kann mehrere Konfigurationen haben, wobei jede Konfiguration genau einer Ressource zugeordnet wird (*siehe Abbildung unten*). Bei Verwendung einer globalen Ressource als Input können Sie mittels Steuerelementen auf der Benutzeroberfläche jederzeit zwischen den verschiedenen Konfigurationen für die globale Ressource wechseln. So können z.B. bei Zuweisung eines XSLT Stylesheet zur Transformationen eines XML-Dokuments über eine globale Ressource (einen Alias) mehrere Konfigurationen für die globale Ressource definiert werden, von denen jede auf eine andere XSLT-Datei verweist. Nachdem Sie die globale Ressource auf diese Art definiert haben, wird beim Wechseln der Konfiguration die für die Transformationen verwendete XSLT-Datei gewechselt.



Eine globale Ressource kann nicht nur verwendet werden, um Ressourcen innerhalb einer Altova-Applikation zu wechseln, sondern auch, um Ressourcen in anderen Altova-Applikationen zu generieren und zu verwenden. So können z.B. Dateien on-the-fly in einer Altova-Applikation generiert werden, um in einer anderen Altova-Applikation verwendet zu werden. Damit lassen sich Entwicklungs- und Testabläufe enorm vereinfachen und beschleunigen. So kann z.B. in XMLSpy ein XSLT Stylesheet verwendet werden, um eine XML-Datei zu transformieren, die on-the-fly mit einem Altova MapForce Mapping generiert wurde.

Um globale Altova-Ressourcen verwenden zu können, sind zwei Schritte erforderlich:

- Definieren globaler Ressourcen 1062: Ressourcen werden definiert und die Definitionen werden in einer XML-Datei gespeichert. Diese Ressourcen können mehreren Altova-Applikationen zur Verfügung gestellt werden.
- Verwendung von globalen Ressourcen 1074: In XMLSpy können Dateien über eine globale Ressource anstatt über einen Dateipfad zugewiesen werden. Der Vorteil ist, dass die verwendete Ressource durch Wechseln der Konfiguration in XMLSpy sofort gewechselt werden kann.

Globale Ressourcen in anderen Altova-Produkten

Derzeit können globale Ressourcen in den folgenden Altova-Produkten definiert und verwendet werden: XMLSpy, StyleVision, MapForce,Authentic Desktop, <%MTD%> und DatabaseSpy..

20.1 Definieren globaler Ressourcen

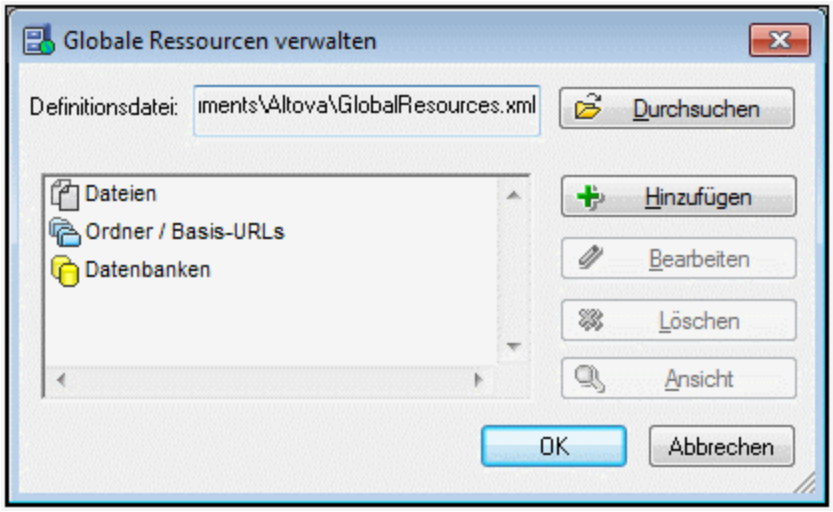
Globale Altova-Ressourcen werden im Dialogfeld **Globale Ressourcen verwalten** definiert, welches auf zwei Arten aufgerufen werden kann:

- Wählen Sie den Menübefehl **Extras | Globale Ressourcen.**
- Klicken Sie in der Symbolleiste "Globale Ressourcen" auf die Schaltfläche **Globale Ressourcen verwalten** , um das gleichnamige Dialogfeld aufzurufen. (*siehe Abbildung unten*)



Die Definitionsdatei für globale Ressourcen

Die Informationen über globale Ressourcen, werden in einer XML-Datei, der Definitionsdatei für globale Ressourcen, gespeichert. Diese Datei wird erstellt, sobald die erste globale Ressource im Dialogfeld "Globale Ressourcen verwalten" (*Abbildung unten*) definiert und gespeichert wird.



Wenn Sie das Dialogfeld "Globale Ressourcen verwalten" zum ersten Mal öffnen, wird der Standardpfad und -name der Definitionsdatei für globale Ressourcen im Textfeld *Definitionsdatei* (*siehe Abbildung oben)* definiert:

```
C:\Users\<username>\My Documents\Altova\GlobalResources.xml
```

Diese Datei ist bei allen Altova-Applikationen als Standard-Definitionsdatei für globale Ressourcen definiert. Infolgedessen kann eine globale Ressource von einer beliebigen Altova-Applikation aus in dieser Datei gespeichert werden und steht dann allen anderen Altova-Applikationen als globale Ressource zur Verfügung. Um eine globale Ressource in der Definitionsdatei für globale Ressourcen zu definieren und zu speichern, fügen Sie die globale Ressource im Dialogfeld "Globale Ressourcen verwalten" hinzu und klicken Sie auf **OK** um sie zu speichern.

Um eine bereits vorhandene Definitionsdatei für globale Ressourcen als aktive Definitionsdatei für eine bestimmte Altova Applikation auszuwählen, suchen Sie im Dialogfeld "Globale Ressourcen verwalten" (*Abbildung unten*) im Feld *Definitionsdatei* danach und wählen Sie sie aus.

© 2016-2022 Altova GmbH

Anmerkung: Sie können der Definitionsdatei für globale Ressourcen jeden beliebigen Namen geben und sie unter jedem beliebigen Pfad, auf den Sie von Ihren Altova Applikationen aus Zugriff haben, speichern. Sie müssen diese Datei in Ihrer Applikation nur (im Textfeld *Definitionsdatei*) als die Definitionsdatei für globale Ressourcen für die jeweilige Applikation definieren. Die Ressourcen lassen sich in allen Altova-Produkten als globale Ressourcen verwenden, wenn Sie in allen Altova-Produkten eine einzige Definitionsdatei verwenden.

Anmerkung: Sie können auch mehrere Definitionsdateien für globale Ressourcen erstellen. Es kann aber immer nur eine davon in einer Altova-Applikation aktiv sein und nur die in dieser Datei enthaltenen Definitionen stehen in der Applikation zur Verfügung. Sie können dadurch je nach Bedarf festlegen, welche Ressourcen nur eingeschränkt und welche in mehreren Produkten zur Verfügung stehen sollen.

Verwalten globaler Ressourcen: Hinzufügen, Bearbeiten, Löschen, Speichern

Im Dialogfeld "Globale Ressourcen verwalten" (*Abbildung oben*) können Sie eine globale Ressource zur ausgewählten Definitionsdatei für globale Ressourcen hinzufügen oder eine ausgewählte globale Ressource löschen oder bearbeiten. In der Definitionsdatei für globale Ressourcen werden die von Ihnen hinzugefügten globalen Ressourcen in Gruppen aufgelistet: Dateien, Ordner und Datenbanken (*siehe Abbildung oben*).

Um eine ***globale Ressource hinzuzufügen***, klicken Sie auf die Schaltfläche **Hinzufügen** und definieren Sie die globale Ressource im entsprechenden **Globale Ressourcen**-Dialogfeld, das angezeigt wird (*Beschreibung zu Dateien[1064], Ordnern[1069] und Datenbanken[1062] in den Unterabschnitten dieses Abschnitts*). Nachdem Sie eine globale Ressource definiert und (durch Klicken auf die Schaltfläche **OK** im Dialogfeld "Globale Ressourcen verwalten") gespeichert haben, wird die globale Ressource zur Bibliothek der globalen Definitionen in der ausgewählten Definitionsdatei für globale Ressourcen hinzugefügt. Die globale Ressource wird durch ein Alias identifiziert.

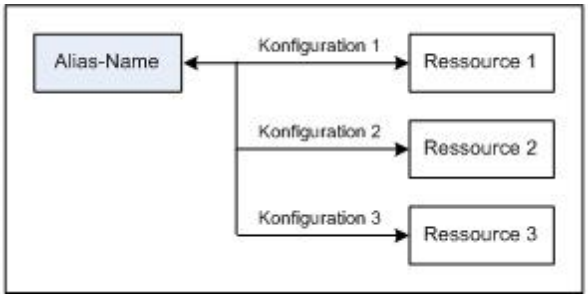
Um eine ***globale Ressource zu bearbeiten***, wählen Sie sie aus und klicken Sie auf **Bearbeiten**. Daraufhin wird das entsprechende Dialogfeld **Globale Ressource** angezeigt, in dem Sie die nötigen Änderungen vornehmen können (*siehe dazu die Beschreibungen in den Unterabschnitten Dateien[1064], Ordner[1069] in den Unterabschnitten dieses Abschnitts*).

Um eine ***globale Ressource zu löschen***, wählen Sie sie aus und klicken Sie auf **Löschen**.

Nachdem Sie mit dem Hinzufügen, Bearbeiten oder Löschen globaler Ressourcen fertig sind, klicken Sie im Dialogfeld **Globale Ressourcen verwalten** auf **OK**, um Ihre Änderungen in der Definitionsdatei für globale Ressourcen ***zu speichern***.

Verknüpfen von globalen Ressourcen mit Aliasnamen über Konfigurationen

Beim Erstellen einer globalen Ressource wird ein Aliasname auf eine Ressource (Datei, Ordner oder Datenbank) gemappt. Jedes Mapping wird als Konfiguration bezeichnet. Ein einziger Aliasname kann daher mittels unterschiedlicher Konfigurationen mit mehreren Ressourcen verknüpft werden (*Abbildung unten*).



Sie können anschließend in einer Altova-Applikation Aliasnamen anstelle von Dateien zuweisen. Für jeden Alias können Sie zwischen den auf diesen Alias gemappten Ressourcen wechseln, indem Sie einfach die aktive Konfiguration für die globalen Ressourcen (aktive Konfiguration) wechseln. Wenn Sie z.B. in Altova XMLSpy am XML-Dokument `MyXML.xml` eine XSLT-Transformation durchführen möchten, können Sie der globalen Ressource den Alias `MyXSLT` zuweisen, damit diese als globale Ressource für XSLT-Transformationen verwendet wird. Sie können anschließend in XMLSpy die aktive Konfiguration wechseln, um andere XSLT-Dateien zu verwenden. Wenn in `Configuration-1` die Datei `First.xslt` auf `MyXSLT` gemappt ist und `Configuration-1` als aktive Konfiguration ausgewählt wird, so wird für die Transformation `First.xslt` verwendet. Auf diese Art können mehrere Konfigurationen verwendet werden, um mehrere Ressourcen über einen einzigen Alias aufzurufen. Diese Methode ist beim Testen und Vergleichen von Ressourcen nützlich. Da globale Ressourcen außerdem in mehreren Altova Produkten verwendet werden können, können die Ressourcen in mehreren Altova-Produkten getestet und verglichen werden.

20.1.1 Dateien

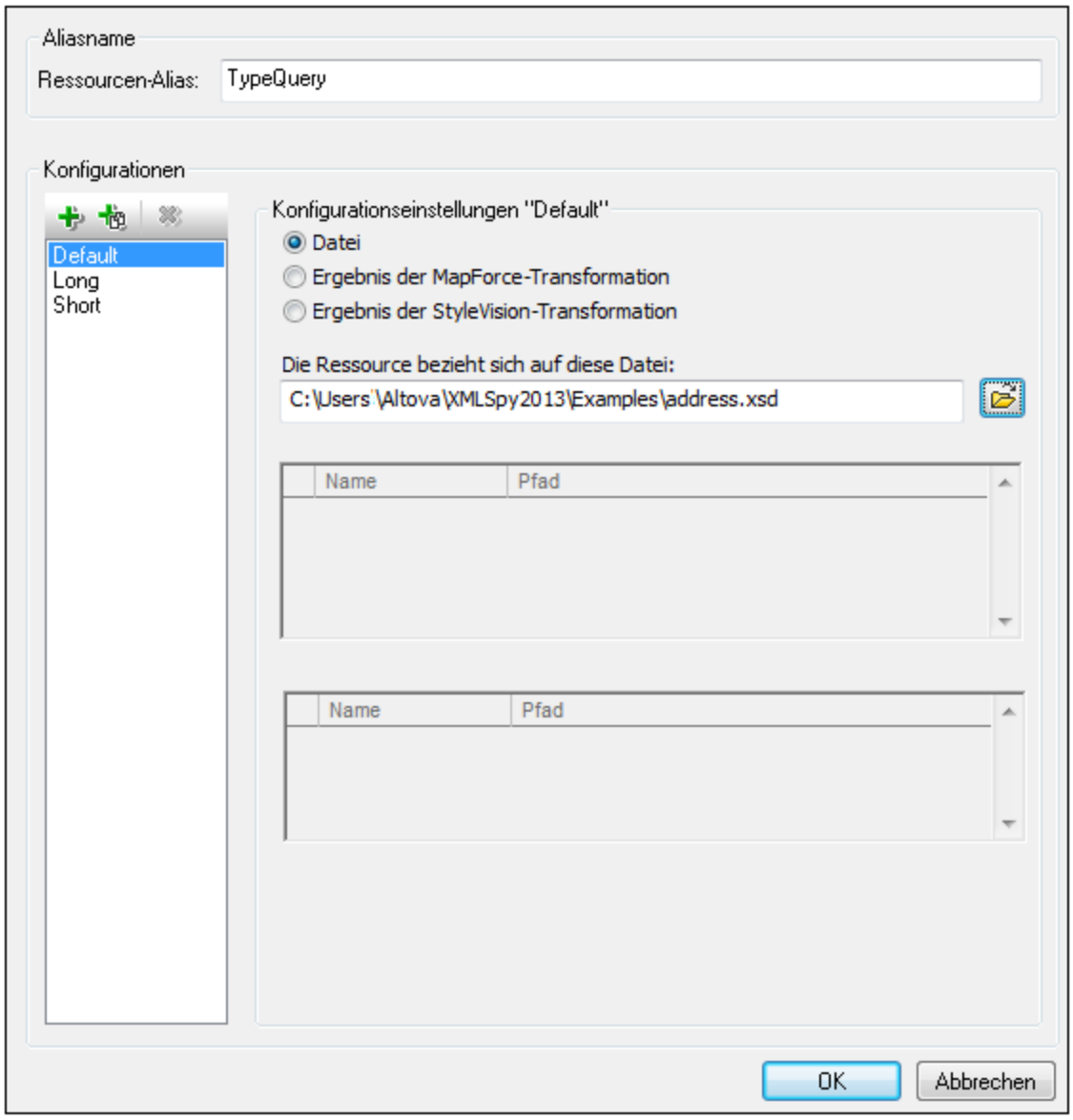
Das Dialogfeld **Globale Ressource** für Dateien (*Abbildung unten*) wird über den Befehl **Hinzufügen | Datei** im Dialogfeld "Globale Ressourcen verwalten" 1062 aufgerufen. In diesem Dialogfeld können Sie die Konfigurationen des im Textfeld *Ressourcen-Alias* angegebenen Alias definieren. Nachdem Sie die Eigenschaften der Konfiguration wie unten beschrieben definiert haben, speichern Sie die Aliasdefinition durch Klicken auf **OK**.

Nachdem Sie eine Aliasdefinition gespeichert haben, können Sie einen weiteren Alias hinzufügen, indem Sie die oben beschriebenen Schritte (Beginnend durch Aufrufen des Befehls **Hinzufügen | Datei** im Dialogfeld "Globale Ressourcen verwalten") 1062 wiederholen.

Dialogfeld "Globale Ressource"

Ein Alias wird im Dialogfeld "Globale Ressource" (*Abbildung unten*) definiert.

© 2016-2022 Altova GmbH



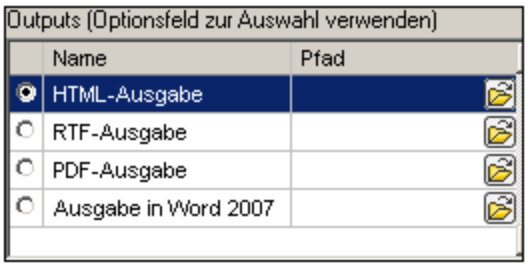
Schaltflächen im Dialogfeld "Globale Ressource"

- *Konfiguration hinzufügen:* Ruft das Dialogfeld "Konfiguration hinzufügen" auf, in das Sie den Namen der hinzuzufügenden Konfiguration eingeben.
- *Konfiguration als Kopie hinzufügen:* Ruft das Dialogfeld "Konfiguration hinzufügen" auf, in das Sie den Namen der Konfiguration eingeben, die als Kopie der ausgewählten Konfiguration erstellt werden soll.
- *Löschen:* Löscht die ausgewählte Konfiguration.
- *Öffnen:* Damit können Sie zur Datei navigieren, die als globale Ressource erstellt werden soll.

Definieren des Alias

So definieren Sie den Alias (seinen Namen und seine Konfigurationen):

1. *Geben Sie dem Alias einen Namen*: Geben Sie den Aliasnamen in das Textfeld *Ressourcen-Alias* ein.
2. *Fügen Sie Konfigurationen hinzu*: Im Fenster "Konfigurationen" finden Sie als Standardeinstellung eine Konfiguration mit dem Namen "`Default`" (*Abbildung oben*). Diese Standardkonfiguration kann weder gelöscht noch umbenannt werden. Sie können beliebig viele zusätzliche Konfigurationen hinzufügen. (i) Klicken Sie dazu auf die Schaltfläche **Konfiguration hinzufügen** oder auf **Konfiguration als Kopie hinzufügen** und (ii) geben Sie im daraufhin angezeigten Dialogfeld einen Konfigurationsnamen ein. Alle hinzugefügten Konfigurationen werden im Fenster "Konfigurationen" aufgelistet. In der Abbildung oben wurden zwei zusätzliche Konfigurationen namens `Long` und `Short` zur Konfigurationsliste hinzugefügt. Mit Hilfe der Schaltfläche "Konfiguration als Kopie hinzufügen" können Sie die ausgewählte Konfiguration kopieren und anschließend bearbeiten.
3. *Wählen Sie für jede Konfiguration einen Ressourcentyp aus*: Wählen Sie eine Konfiguration in der Konfigurationsliste aus und definieren Sie im Bereich *Konfigurationseinstellungen* eine Ressource für die Konfiguration: (i) Datei, (ii) Ergebnis der MapForce Transformation oder (iii) Ergebnis der StyleVision Transformation. Aktivieren Sie das entsprechende Optionsfeld. Bei Auswahl einer MapForce bzw. StyleVision Transformation wird anhand der entsprechenden `.mfd` oder `.sps`-Datei und der entsprechenden Input-Datei eine Transformation in MapForce oder StyleVision durchgeführt. Die Ressource wird mit der Transformation erzeugt.
4. *Wählen Sie eine Datei für den Ressourcentyp aus:* Wenn es sich bei der Ressource um eine direkt ausgewählte Datei handelt, navigieren Sie im Textfeld *Die Ressource bezieht sich auf diese Datei*: zur gewünschten Datei. Wenn die Ressource das Ergebnis einer Transformation ist, navigieren Sie im Textfeld *Die Ressource bezieht sich auf diese Datei* zur `.mfd` Datei (bei MapForce-Transformationen) oder der `.sps` Datei (bei StyleVision-Transformationen). In Fällen, in denen mehrere Inputs oder Outputs für die Transformation möglich sind, wird eine Auswahl der Optionen angezeigt. So hängen z.B. die Ausgabeoptionen einer StyleVision-Transformationen von der installierten StyleVision Edition ab (*in der Abbildung unten sehen Sie die Output-Optionen der Enterprise Edition*).



   Aktivieren Sie das Optionsfeld für die gewünschte Option (in der Abbildung oben wurde 'HTML-Ausgabe' ausgewählt). Wenn die Ressource das Ergebnis einer Transformation ist, kann die Ausgabe als Datei oder selbst als globale Ressource gespeichert werden. Klicken Sie auf die Schaltfläche und wählen Sie entweder "Globale Ressource" aus (um die Ausgabe als globale Ressource zu speichern) oder "Durchsuchen" (um die Ausgabe als Datei zu speichern). Wenn keine dieser beiden Speicheroptionen ausgewählt ist, wird das Transformationsergebnis als temporäre Datei geladen, wenn die globale Ressource aufgerufen wird.

5. *Definieren Sie, falls nötig, mehrere Konfigurationen:* Sie können mehrere Konfigurationen hinzufügen und für jede davon eine Ressource definieren. Wiederholen Sie dazu für jede Konfiguration die Schritte 3 und 4. Sie können jederzeit eine neue Konfiguration zur Aliasdefinition hinzufügen.

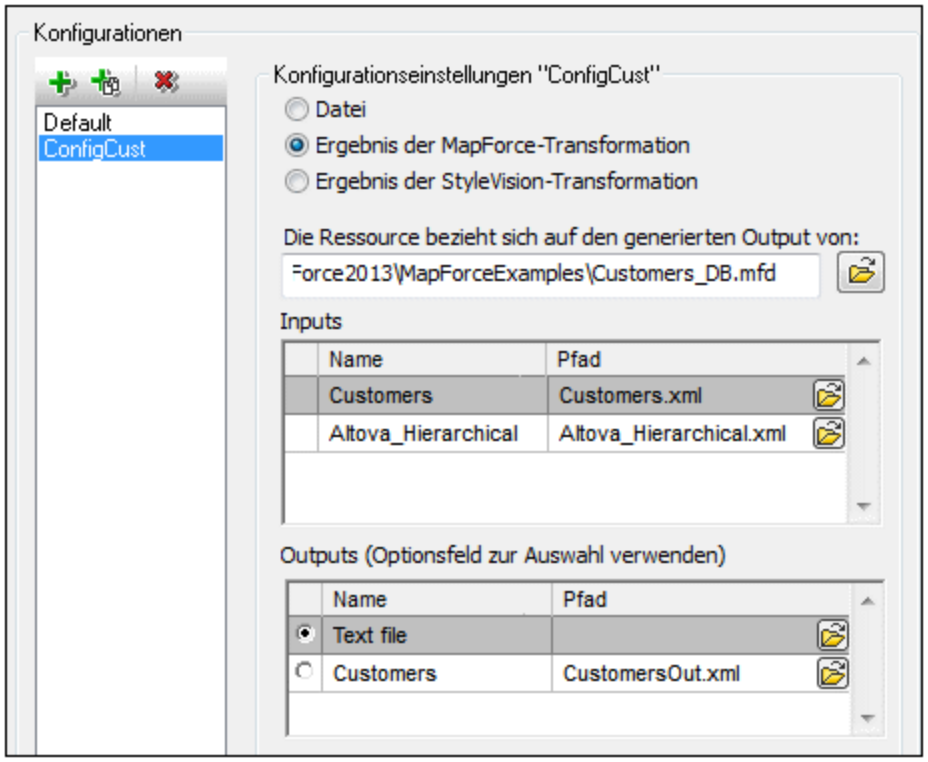
© 2016-2022 Altova GmbH

6. *Speichern Sie die Aliasdefinition:* Klicken Sie auf **OK,** um den Alias und alle seine Konfigurationen als globale Ressource zu speichern. Die globale Ressource wird im Dialogfeld "Globale Ressourcen verwalten" 1062 unter "Dateien" aufgelistet.

Ergebnis einer MapForce-Transformation

Altova MapForce mappt ein oder mehrere (bestehende) Input-Dokument-Schemas auf ein oder mehrere (neue) Output-Dokument-Schemas. Dieses Mapping, das von einem Benutzer in MapForce erstellt wird, wird als MapForce Design (MFD) bezeichnet. Als Datenquellen können XML-Dateien, Textdateien, Datenbanken usw., die den Input-Schemas entsprechen, verwendet werden. MapForce generiert XML-Ausgabedateien, die dem Output-Dokument-Schema entsprechen. Dieses generierte Ausgabedokument ist die Datei, die unter *Ergebnis der MapForce-Transformation*, als globale Ressource verwendet wird.

Wenn Sie eine von MapForce-generierte Datendatei als globale Ressource definieren möchten, müssen Sie im Dialogfeld "Globale Ressource" folgende Einstellungen definieren (*siehe Abbildung unten*):



- ***eine `.mfd` (MapForce Design) Datei.*** Sie müssen diese Datei im Textfeld *Die Ressource bezieht sich auf den generierten Output von:* (*siehe Abbildung oben*) definieren.
- ***eine oder mehrere Input-Datendateien.*** Nachdem die MFD-Datei definiert wurde, wird sie analysiert und im Bereich *Inputs (siehe Abbildung oben*) werden Standarddatendateien angezeigt. Sie können die Auswahl der Standarddatei für jedes Input-Schema ändern, indem Sie eine andere Datei definieren.
- ***eine Output-Datei.*** Wenn das MFD-Dokument mehrere Output-Schemas hat, werden diese alle im Bereich *Outputs* (*siehe Abbildung oben*) aufgelistet und Sie müssen eine davon auswählen. Wenn der Pfad der Ausgabedatei eines einzelnen Output-Schemas im MFD-Dokument definiert ist, so wird dieser Dateipfad für dieses Output-Schema im Bereich *Outputs* angezeigt. In der Abbildung oben sehen wir, dass im MFD-Dokument festgelegt ist, dass das Output-Schema `Customers` eine XML-

Standard-Datendatei hat (`CustomersOut.xml`), während das Output-Schema `Text file` in der MFD-Datei keine Dateiverknüpfung hat. Sie können den Standard-Dateipfad im Bereich "Outputs" verwenden oder selbst einen Pfad definieren. Das Ergebnis der MapForce Transformation wird unter dem Dateipfad des ausgewählten Output-Schemas gespeichert. Dies ist die Datei, die als globale Ressource verwendet wird.

Anmerkung: Der Vorteil dieser Option (Ergebnis der MapForce-Transformation) ist, dass die Transformation zum Zeitpunkt des Aufrufs der globalen Ressource durchgeführt wird. Dadurch enthält die globale Ressource die aktuellsten Daten (aus der/den Input-Datei(en).

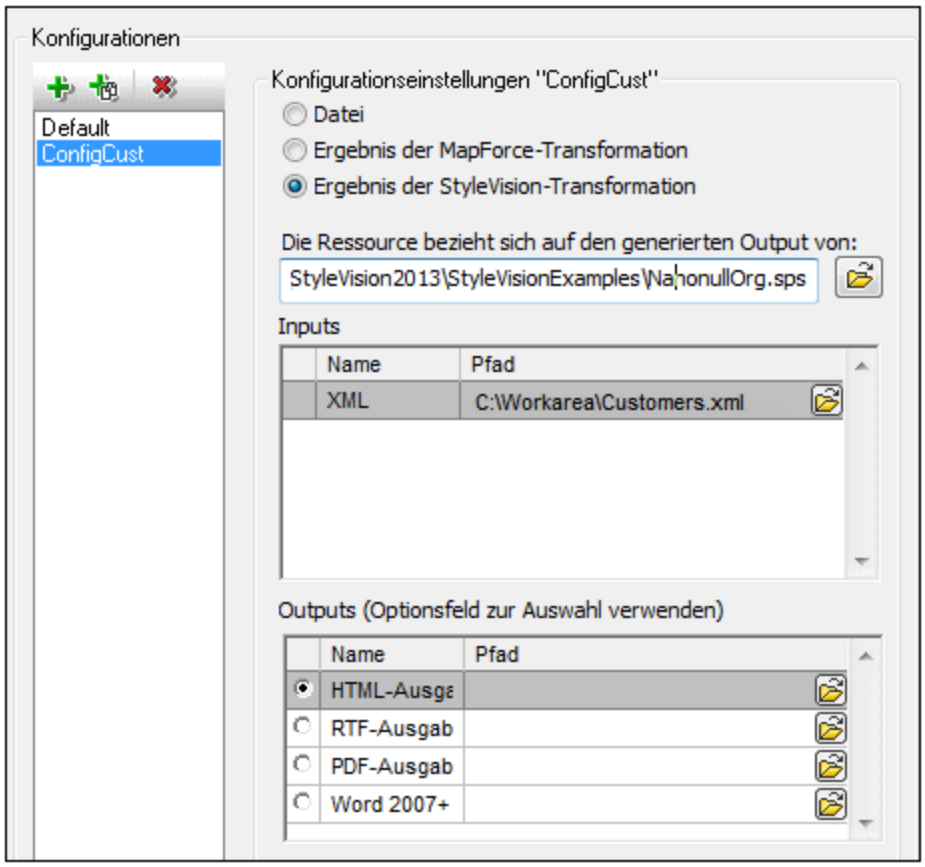
Anmerkung: Da zur Ausführung der Transformation MapForce benötigt wird, muss Altova MapForce installiert sein, damit diese Funktion zur Verfügung steht.

Ergebnis der StyleVision Transformation

Mit Altova StyleVision werden StyleVision Power Stylesheet (SPS)-Dateien erzeugt. Mit Hilfe dieser SPS-Dateien werden XSLT-Stylesheets generiert, mit Hilfe derer XML-Dokumente in Ausgabedokumente in verschiedenen Formaten (HTML, PDF, RTF, Word 2007+, usw) transformiert werden. Wenn Sie die Option *Ergebnis der StyleVision Transformation* auswählen, wird das von StyleVision erstellte Ausgabedokument als die globale Ressource verwendet, die mit der ausgewählten Konfiguration verknüpft ist.

Für die Option *Ergebnis der StyleVision Transformation* im Dialogfeld "Globale Ressource" (*siehe Abbildung unten*) müssen die folgenden Dateien definiert werden.

© 2016-2022 Altova GmbH



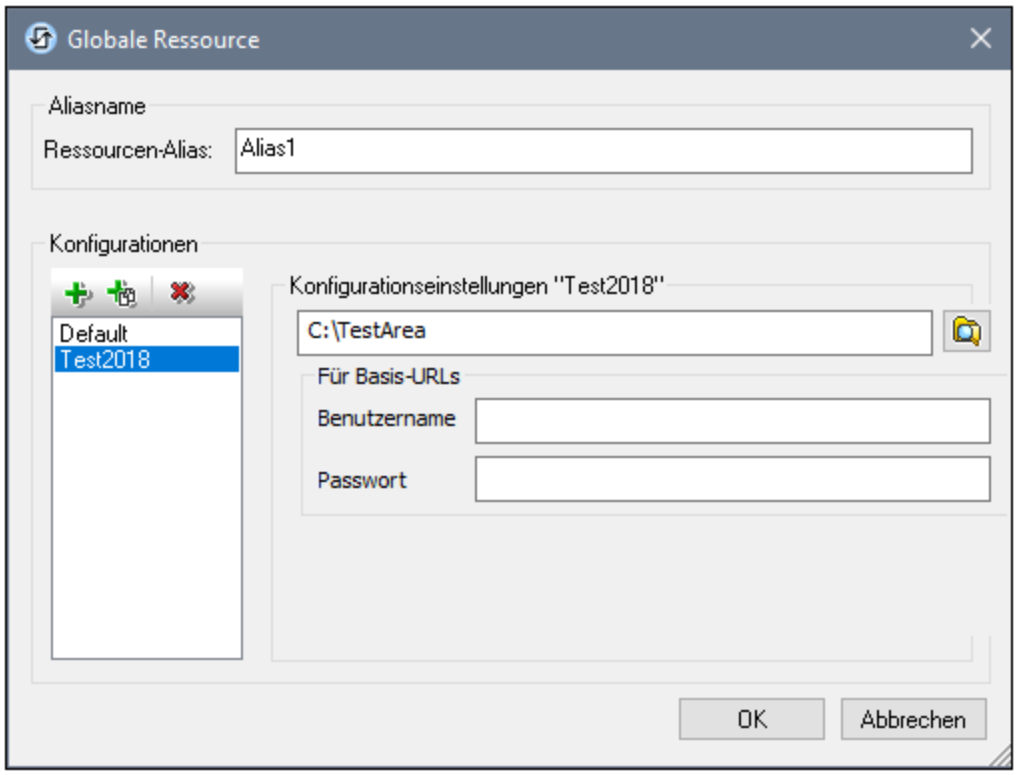
- ***eine*** ***.sps*** ***(SPS) Datei.*** Diese Datei muss im Textfeld *Die Ressource bezieht sich auf den generierten Output von* (*siehe Abbildung oben*) definiert werden.
- ***Input-Datei(en).*** Eventuell wurde die Input-Datei bereits in der SPS-Datei definiert. In diesem Fall wird sie automatisch im Bereich *Inputs* angezeigt, sobald die SPS-Datei ausgewählt wurde. Sie können diesen Eintrag ändern. Falls kein Input definiert ist, müssen Sie eine Input-Datei hinzufügen.
- ***Output-Datei(en).*** Wählen Sie im Bereich *Outputs* das Ausgabeformat aus und definieren Sie einen Ausgabedateipfad für dieses Format.

Anmerkung: Der Vorteil dieser Option (Ergebnis der StyleVision-Transformation) ist, dass die Transformation zum Zeitpunkt des Aufrufs der globalen Ressource durchgeführt wird. Dadurch enthält die globale Ressource die aktuellsten Daten (aus der/den Input-Datei(en).

Anmerkung: Da zur Ausführung der Transformation StyleVision benötigt wird, muss Altova StyleVision installiert sein, damit diese Funktion zur Verfügung steht.

20.1.2 Ordner

Im Dialogfeld **Globale Ressource** für Ordner (*Abbildung unten*) können Sie eine Ordnerressource auf folgende Art hinzufügen:



Schaltflächen im Dialogfeld "Globale Ressource"

Konfiguration hinzufügen: Ruft das Dialogfeld "Konfiguration hinzufügen" auf, in das Sie den Namen der hinzuzufügenden Konfiguration eingeben.

Konfiguration als Kopie hinzufügen: Ruft das Dialogfeld "Konfiguration hinzufügen" auf, in das Sie den Namen der Konfiguration eingeben, die als Kopie der ausgewählten Konfiguration erstellt werden soll.

Löschen: Löscht die ausgewählte Konfiguration.

Öffnen: Damit können Sie zum Ordner navigieren, der als globale Ressource erstellt werden soll.

Definieren des Alias

So definieren Sie den Alias (seinen Namen und seine Konfigurationen):

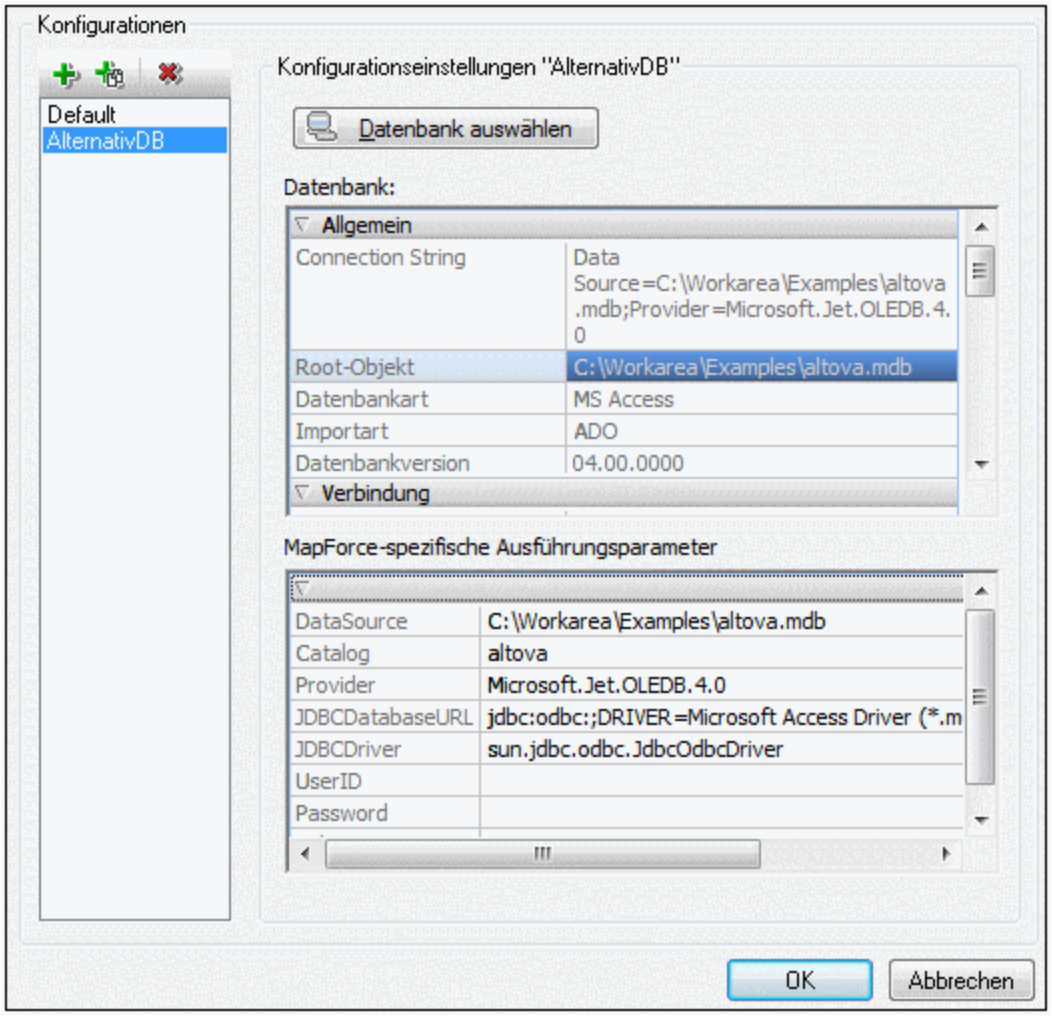
1. *Geben Sie dem Alias einen Namen*: Geben Sie den Aliasnamen in das Textfeld *Ressourcen-Alias* ein.
2. *Fügen Sie Konfigurationen hinzu*: Im Fenster "Konfigurationen" finden Sie eine Konfiguration mit dem Namen "Default" (*Abbildung oben*). Diese Standardkonfiguration kann nicht gelöscht werden und der Name kann nicht geändert werden. Sie können beliebig viele zusätzliche Konfigurationen für den ausgewählten Alias hinzufügen. Klicken Sie dazu auf die Schaltfläche **Konfiguration hinzufügen** oder auf **Konfiguration als Kopie hinzufügen.** Geben Sie im angezeigten Dialogfeld den Konfigurationsnamen ein. Klicken Sie auf **OK**. Die neue Konfiguration wird im Bereich "Konfigurationen" aufgelistet. Wiederholen Sie diesen Schritt um so viele Konfigurationen wie erforderlich hinzuzufügen.

© 2016-2022 Altova GmbH

3. *Wählen Sie einen Ordner als Ressource einer Konfiguration aus*: Wählen Sie im Bereich "Konfigurationen" eine der Konfigurationen aus und navigieren Sie zum Ordner, den Sie als globale Ressource erstellen möchten. Wenn für den Zugriff auf einen Ordner Anmeldeinformationen erforderlich sind, definieren Sie diese in den Feldern *Benutzername* und *Passwort*.
4. *Definieren Sie, falls nötig, mehrere Konfigurationen*: Definieren Sie eine Ordnerressource für jede Konfiguration, die Sie erstellt haben (d.h. wiederholen Sie Schritt 3 für die verschiedenen von Ihnen erstellten Konfigurationen). Sie können jederzeit eine neue Konfiguration zur Aliasdefinition hinzufügen.
5. *Speichern Sie die Aliasdefinition*: Klicken Sie im Dialogfeld **Globale Ressource** auf **OK**, um den Alias und alle seine Konfigurationen als globale Ressource zu speichern. Die globale Ressource wird im Dialogfeld Dialogfeld "Globale Ressourcen verwalten" 1062 unter "Ordner" aufgelistet.

20.1.3 Datenbanken

Im Dialogfeld **Globale Ressource** für Datenbanken (Abbildung unten) können Sie auf folgende Weise eine Datenbankressource hinzufügen:



Schaltflächen im Dialogfeld "Globale Ressource"



Konfiguration hinzufügen: Ruft das Dialogfeld "Konfiguration hinzufügen" auf, in das Sie den Namen der hinzuzufügenden Konfiguration eingeben.



Konfiguration als Kopie hinzufügen: Ruft das Dialogfeld "Konfiguration hinzufügen" auf, in das Sie den Namen der Konfiguration eingeben, die als Kopie der ausgewählten Konfiguration erstellt werden soll.



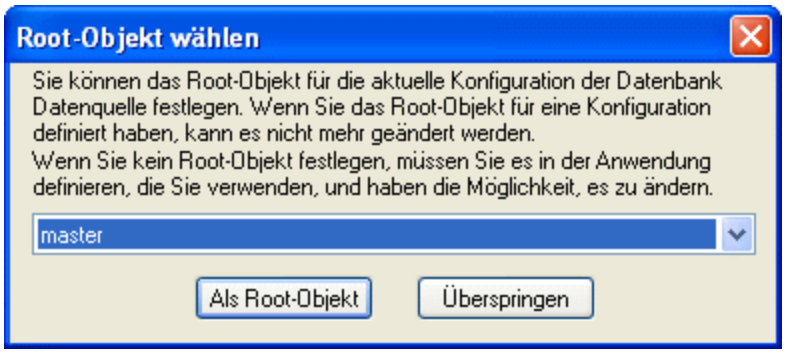
Löschen: Löscht die ausgewählte Konfiguration.

Definieren des Aliasnamens

So definieren Sie den Alias (seinen Namen und seine Konfigurationen):

1. *Geben Sie dem Alias einen Namen*: Geben Sie den Aliasnamen in das Textfeld *Ressourcen-Alias* ein.
2. *Fügen Sie Konfigurationen hinzu*: Im Fenster "Konfigurationen" finden Sie eine Konfiguration mit dem Namen "Default" (*Abbildung oben*). Diese Standardkonfiguration kann nicht gelöscht werden und der Name kann nicht geändert werden. Sie können beliebig viele zusätzliche Konfigurationen für den ausgewählten Alias hinzufügen. Klicken Sie dazu auf die Schaltfläche **Konfiguration hinzufügen** oder auf **Konfiguration als Kopie hinzufügen.** Geben Sie im angezeigten Dialogfeld den Konfigurationsnamen ein. Klicken Sie auf **OK**. Die neue Konfiguration wird im Bereich "Konfigurationen" aufgelistet. Wiederholen Sie diesen Schritt, um so viele Konfigurationen wie erforderlich hinzuzufügen.
3. *Beginnen Sie mit der Auswahl einer Datenbank als Ressource einer Konfiguration*: Wählen Sie im Bereich "Konfigurationen" eine der Konfigurationen aus und klicken Sie auf die Schaltfläche **Datenbank wählen**. Daraufhin wird das Dialogfeld **Verbindung zu Globaler Ressource erstellen** angezeigt.
4. *Stellen Sie die Datenbankverbindung her*: Wählen Sie aus, ob Sie die Verbindung zur Datenbank mit dem Verbindungsassistenten herstellen, eine bestehende Verbindung verwenden möchten oder eine ADO-Verbindung, eine ODBC- oder eine JDBC-Verbindung herstellen möchten. Stellen Sie die Definition der Verbindungsmethode fertig, wie im Abschnitt Herstellen einer Verbindung zu einer Datenbank[969] beschrieben. Sie können den Verbindungsassistenten[970], ADO-Verbindungen[975] oder ODBC-Verbindungen[989] verwenden.. Wenn von XMLSpy aus bereits eine Verbindung zu einer Datenbank hergestellt wurde, können Sie auf die Schaltfläche "Bestehende Verbindungen" klicken und die Datenbank aus der angezeigten Liste der Verbindungen auswählen.
5. *Wählen Sie das Root-Objekt aus*: Wenn Sie eine Verbindung zu einem Datenbankserver herstellen, bei dem ein Root-Objekt ausgewählt werden kann, werden Sie im Dialogfeld **Root-Objekt wählen** (*Abbildung unten*) aufgefordert, ein Root-Objekt auf dem Server auszuwählen. Wählen Sie das Root-Objekt aus und klicken Sie auf **Als Root-Objekt.** Das ausgewählte Root-Objekt ist das Root-Objekt, das bei Verwendung der Konfiguration geladen wird.

© 2016-2022 Altova GmbH



Klicken Sie auf die Schaltfläche **Überspringen**, wenn alle Root-Objekte in den Anwendungen verfügbar sein sollen. Im letzteren Fall muss der Benutzer das passende Root-Objekt in der entsprechenden Altova Anwendung auswählen, wenn die globale Ressource verwendet wird.

6. *Definieren Sie, falls nötig, mehrere Konfigurationen*: Geben Sie für jede Konfiguration eine Datenbankressource an (d.h. wiederholen Sie Schritt 3 und 4 für die verschiedenen von Ihnen erstellten Konfigurationen). Sie können jederzeit eine neue Konfiguration zur Aliasdefinition hinzufügen.
7. *Speichern Sie die Aliasdefinition*: Klicken Sie im Dialogfeld **Globale Ressource** verwalten" auf **OK**, um den Alias und alle seine Konfigurationen als globale Ressource zu speichern. Die globale Ressource wird im Dialogfeld **Globale Ressourcen verwalten** unter "Datenbanken" aufgelistet.

20.2 Verwendung globaler Ressourcen

Es gibt mehrere Arten von globalen Ressourcen (Datei, Ordner und Datenbank). Einige Szenarien, in denen Sie globale Ressourcen in XMLSpy verwenden können, sind hier aufgelistet: Dateien und Ordner[1074] und Datenbanken[1077].

Auswahloptionen, die festlegen, welche Ressource verwendet wird

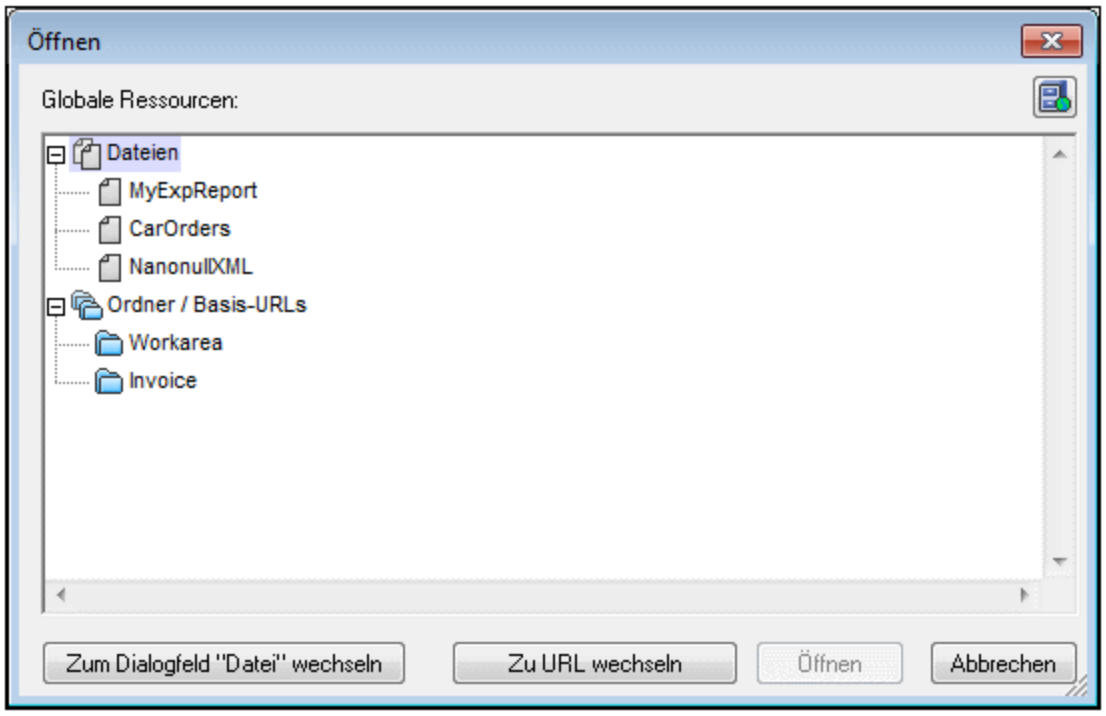
Es gibt zwei applikationsweite Auswahlmöglichkeiten, um festzulegen, welche globalen Ressourcen verwendet werden können und welche globalen Ressourcen zu einem bestimmten Zeitpunkt tatsächlich verwendet werden:

- *Die aktive XML-Datei für globale Ressourcen* wird im Dialogfeld "Globale Ressource"[1062] ausgewählt. Die globalen Ressource-Definitionen, die in der aktiven XML-Datei für globale Ressourcen enthalten sind, stehen allen in der Applikation geöffneten Dateien zur Verfügung. Nur die Definitionen in der aktiven XML-Datei für globale Ressourcen sind verfügbar. Sie können jederzeit eine andere XML-Datei für globale Ressourcen verwenden. In diesem Fall werden die globalen Ressourcen der vorher aktiven Datei sofort durch die der neuen aktiven Datei ersetzt. Die aktive XML-Datei für globale Ressourcen legt also Folgendes fest: (i) welche globale Ressourcen zugewiesen werden können und (ii) welche globalen Ressourcen angezeigt werden können (Wenn z.B. eine globale Ressource in einer XML-Datei für globale Ressourcen zugewiesen ist, es aber in der aktiven XML-Datei für globale Ressourcen keine globale Ressource dieses Namens gibt, dann kann die zugewiesene globale Ressource (der Alias) nicht angezeigt werden).
- *Die aktive Konfiguration* wird über den Menübefehl **Extras | Aktive Konfiguration**[1579] oder über die Symbolleiste "Globale Ressourcen" ausgewählt. Bei Auswahl dieses Befehls wird eine Liste der Konfigurationen für alle Aliasse angezeigt. Bei Auswahl einer Konfiguration wird diese in der gesamten Applikation aktiv. D.h. immer wenn eine globale Ressource (oder ein Alias) verwendet wird, wird die Ressource, die der aktiven Konfiguration der einzelnen verwendeten Aliasse entspricht, geladen. Die aktive Konfiguration wird auf alle verwendeten Aliasse angewendet. Wenn ein Alias keine Konfiguration mit dem Namen der aktiven Konfiguration hat, wird die Standardkonfiguration dieses Aliasses verwendet. Die aktive Konfiguration spielt beim Zuweisen von Ressourcen keine Rolle; Sie ist nur dann von Bedeutung, wenn die Ressourcen tatsächlich verwendet werden.

20.2.1 Zuweisen von Dateien und Ordnern

Die Zuweisung von globalen Ressourcen vom Typ Datei und Ordner erfolgt auf unterschiedliche Weise. In jedem der unten beschriebenen Szenarios erscheint bei Auswahl des Befehls **Globale Ressourcen** das Dialogfeld "Globale Ressource öffnen" (*Abbildung unten*).

© 2016-2022 Altova GmbH



Globale Ressourcen verwalten: Ruft das Dialogfeld Globale Ressourcen verwalten[1062] auf.

Bei Auswahl einer *globalen Ressource vom Typ Datei* wird die Datei zugewiesen. Bei Auswahl einer *globalen Ressource vom Typ Ordner* wird ein Öffnen-Dialogfeld angezeigt, in dem Sie nach der erforderlichen Datei suchen können. Der Pfad zur ausgewählten Datei wird relativ zur Ordnerressource eingegeben. Wenn also eine globale Ressource vom Typ Ordner zwei Konfigurationen hat, von denen jede auf einen anderen Ordner verweist, so könnten Dateien mit demselben Namen, die jedoch in unterschiedlichen Ordnern liegen, über diese zwei Konfigurationen aktiviert werden. Dies kann vor allem für Testzwecke nützlich sein.

Sie können durch Klicken auf die entsprechende Schaltfläche am unteren Rand des Dialogfelds zum Datei- bzw. URL-Dialogfeld wechseln. Mit der Schaltfläche **Globale Ressourcen verwalten** im rechten oberen Eck erscheint das Dialogfeld Globale Ressourcen verwalten[1062].

Verwendungsszenarien

Globale Ressourcen vom Typ Datei und Ordner können in den folgenden Szenarios verwendet werden:

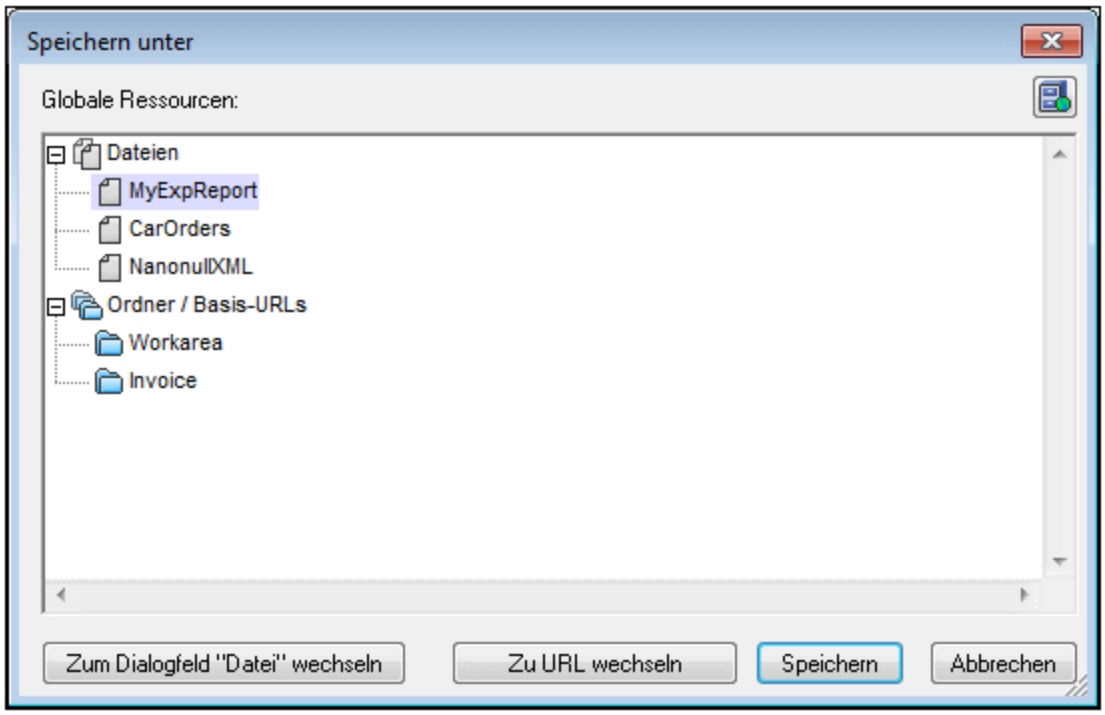
- Öffnen globaler Ressourcen[1076]
- Speichern als globale Ressource[1076]
- Zuweisen von Dateien zu XSLT-Transformationen[1077]
- XSLT-Transformation[1077]
- XQuery-Ausführungen[1077]
- Zuweisen eines SPS[1077]

Öffnen globaler Ressourcen

Eine globale Ressource kann in XMLSpy mit dem Befehl **Datei | Öffnen (Zu globaler Ressource wechseln)** 1280 geöffnet und bearbeitet werden. Im Fall einer globalen Ressource vom Typ Datei wird die Datei direkt geöffnet. Im Fall einer globalen Ressource vom Typ Ordner wird ein Öffnen-Dialogfeld aufgerufen, in dem der verknüpfte Ordner ausgewählt ist. Sie können nun in den untergeordneten Ordnern nach der gewünschten Datei suchen. Ein Vorteil der Definition von Dateien mittels globaler Ressourcen für die Bearbeitung ist, dass miteinander in Zusammenhang stehende Dateien unter unterschiedlichen Konfigurationen einer einzigen globalen Ressource gespeichert und nur durch Wechseln der Konfigurationen aufgerufen werden können. Alle Bearbeitungen müssten vor dem Ändern der Konfiguration gespeichert werden.

Speichern als globale Ressource

Eine neu erstellte Datei kann als globale Ressource gespeichert werden. Sie können auch eine vorhandene Datei öffnen und anschließend als globale Ressource speichern. Wenn Sie auf den Befehl **Datei | Speichern** oder **Datei | Speichern unter** klicken, wird das Dialogfeld "Speichern" aufgerufen. Klicken Sie auf die Schaltfläche **Globale Ressource**, um die verfügbaren globalen Ressourcen (*Abbildung unten*) aufzurufen. Dies sind die Aliasnamen, die in der aktuellen XML-Datei für globale Ressourcen gespeichert sind.



Wählen Sie einen Alias aus und klicken Sie auf **Speichern**. Wenn es sich beim Alias um einen Dateialias 1064 handelt, wird die Datei direkt gespeichert. Wenn es sich um einen Ordneralias 1069 handelt, wird ein Dialogfeld angezeigt, in dem Sie den Namen angeben müssen, unter dem die Datei gespeichert werden soll. In beiden Fällen wird die Datei unter dem Pfad gespeichert, der für die gerade aktive Konfiguration 1078 definiert wurde.

Anmerkung: Jede Konfiguration verweist auf einen bestimmten Dateipfad, der in der Definition dieser Konfiguration festgelegt ist. Wenn die Datei, die Sie als globale Ressource speichern, nicht dieselbe Dateityperweiterung wie die Datei unter dem aktuellen Pfad der Konfiguration hat, kann es beim Öffnen

© 2016-2022 Altova GmbH

der globalen Ressource in XMLSpy zu Bearbeitungs- und Validierungsfehlern kommen. Der Grund dafür ist, dass XMLSpy die Datei unter der Annahme öffnet, dass die Datei den in der Konfigurationsdefinition angegebenen Dateityp aufweist.

Zuweisen von Dateien zu XSLT-Transformationen

XSLT-Dateien können XML-Dokumenten und XML-Dateien können XSLT-Dokumenten über globale Ressourcen zugewiesen werden. Bei Auswahl der Befehle zum Zuweisen von XSLT-Dateien (**XSL/XQuery | XSL zuweisen**[1420] und **XSL/XQuery | XSL:FO zuweisen**[1421]) und XML-Dateien (**XSL/XQuery | XML-Beispieldatei zuweisen**[1421]) wird das Zuweisungsdialogfeld aufgerufen. Wenn Sie auf die Schaltfläche **Durchsuchen** klicken, wird das Dialogfeld "Öffnen" angezeigt, in dem Sie zum Dialogfeld "Globale Ressource öffnen" wechseln können. Ein großer Vorteil bei Verwendung einer globalen Ressource zur Definition einer XSLT-Transformation ist, dass die XSLT-Datei (oder die XML-Datei) für eine Transformation einfach durch Wechseln der aktiven Konfiguration in XMLSpy geändert werden kann; Sie müssen nicht jedes Mal, wenn Sie eine Transformation mit einer anderen Datei durchführen wollen, eine neue Dateizuweisung vornehmen. Wenn eine XSLT-Transformation gestartet wird, wird/werden dafür die Datei/die Dateien verwendet, die mit der aktiven Konfiguration verknüpft sind.

XSLT-Transformationen und XQuery-Ausführungen

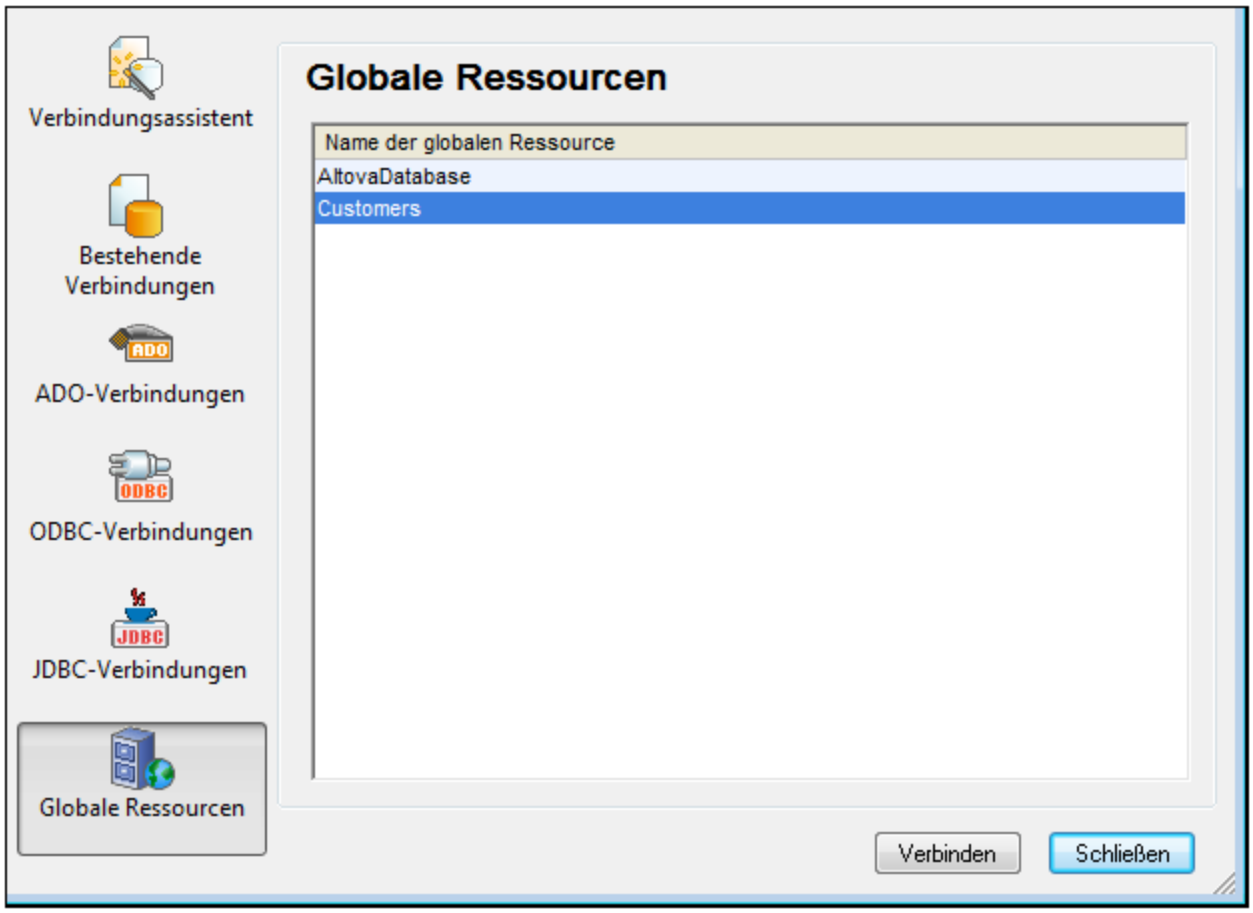
Bei Auswahl des Befehls **XSL/XQuery | XSL-Transformation**[1411] oder **XSL/XQuery | XSL:FO-Transformation**[1412] oder **XSL/XQuery | XQuery Update-Ausführung**[1417] wird ein Dialogfeld angezeigt, in dem Sie nach der gewünschten XSLT-, XQuery- oder XML-Datei suchen können. Klicken Sie auf die Schaltfläche **Durchsuchen** und anschließend auf **Globale Ressource** um das Dialogfeld "Globale Ressource öffnen" aufzurufen (*Abbildung ganz oben im Abschnitt*[1074]). Für die Transformation wird die Datei verwendet, die mit der aktuelle aktiven Konfiguration der ausgewählten globalen Ressource verknüpft ist.

Zuweisen eines SPS

Bei der Zuweisung eines StyleVision Stylesheet zu einer XML-Datei (**Authentic | StyleVision Stylesheet zuweisen**) können Sie eine globale Ressource als Pfad für das Stylesheet auswählen. Klicken Sie auf die Schaltfläche **Durchsuchen** und anschließend **Globale Ressource**, um das Dialogfeld "Globale Ressource öffnen" aufzurufen (*Abbildung ganz oben im Abschnitt*[1074]). Wenn für die Zuweisung eine globale Ressource ausgewählt ist, kann die Authentic-Ansicht des XML-Dokuments einfach durch Wechseln der aktiven Konfiguration in XMLSpy geändert werden.

20.2.2 Zuweisen von Datenbanken

Bei Ausführung eines Befehls, der Daten oder eine Datenstruktur (als ein XML-Schema) aus einer Datenbank in XMLSpy importiert (z.B. mit dem Befehl **Konvertieren | Datenbankdaten importieren**) können Sie die Option zur Verwendung der globalen Ressource auswählen (*Abbildung unten*). Weitere Befehle, bei denen eine globale Ressource vom Typ Datenbank verwendet werden kann, sind datenbankbezogene Befehle im Menü.



Im Verbindungsdialogfeld (*Abbildung oben)* werden alle globalen Ressourcen vom Typ Datenbank, die in der derzeit aktiven XML-Datei für globale Ressourcen[1062] definiert wurden, angezeigt. Wählen Sie die gewünschte globale Ressource aus und klicken Sie auf **Verbinden**. Wenn die ausgewählte globale Ressource mehr als eine Konfiguration hat, wird die Datenbankressource für die gerade aktive Konfiguration (überprüfen Sie diese unter **Extras | Aktive Konfiguration** oder über die Symbolleiste "Globale Ressourcen") verwendet und die Verbindung wird hergestellt. Sie müssen nun auswählen, welche Datenstrukturen und Daten verwendet werden sollen (Beschreibung siehe Erstellen eines XML-Schemas anhand einer Datenbank[1484] und Importieren von Datenbankdaten)[1473].

20.2.3 Wechseln der aktiven Konfiguration

Es kann immer nur eine Konfiguration einer globalen Ressource aktiv sein. Diese Konfiguration wird als die aktive Konfiguration bezeichnet. Sie ist applikationsweit aktiv. D.h. die aktive Konfiguration ist für alle globalen Ressourcenaliasse in allen derzeit offenen Dateien und Datenquellenverbindungen aktiv. Wenn ein Alias keine Konfiguration mit dem Namen der aktiven Konfiguration hat, wird die Standardkonfiguration für diesen Alias verwendet. Betrachten wir als Beispiel, wie man eine Konfiguration wechselt, einen Fall, in dem eine Datei über eine globale Ressource mit mehreren Konfigurationen zugewiesen wurde. Jeder Konfiguration ist eine andere Datei zugewiesen. Welche Datei ausgewählt ist, hängt daher davon ab, welche Konfiguration als die aktive Konfiguration der Applikation ausgewählt ist.

© 2016-2022 Altova GmbH

Sie können die aktive Konfiguration auf folgende Arten wechseln:

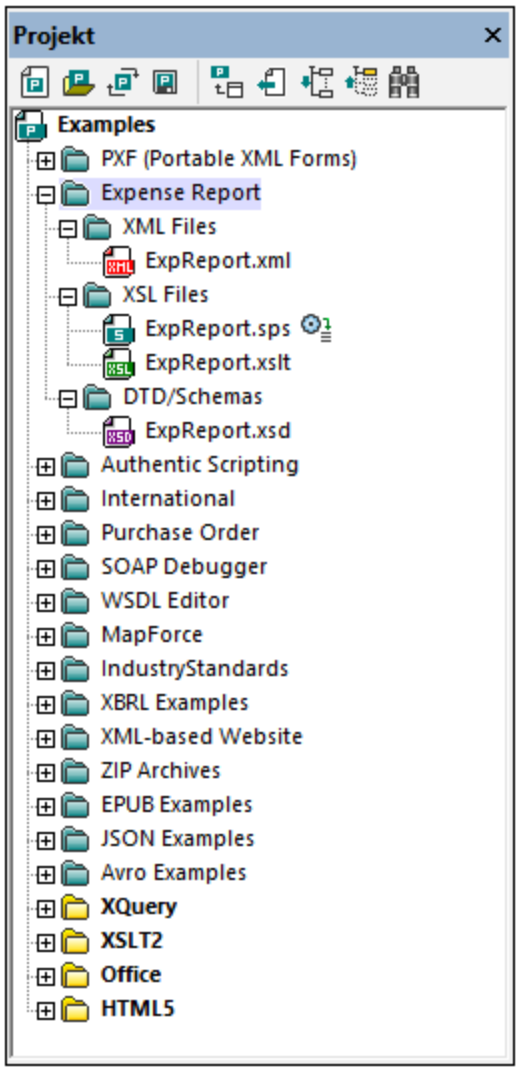
- Über den Menübefehl **Extras | Aktive Konfiguration.** Wählen Sie die gewünschte Konfiguration aus dem Untermenü des Befehls aus.
- Wählen Sie in der Auswahlliste der Symbolleiste "Globale Ressourcen" (*Abbildung unten*) die gewünschte Konfiguration aus.



Auf diese Art können Sie durch Ändern der aktiven Konfiguration Quelldateien auswechseln, die über eine globale Ressource zugewiesen sind.

21 Projekte

Ein Projekt ist eine Sammlung von Dateien, die auf eine von Ihnen festgelegten Art miteinander in Zusammenhang stehen. In der Abbildung unten enthält z.B. ein Projekt namens `Examples` die Dateien für verschiedene Beispiele in verschiedenen Beispielordnern, von denen jeder wiederum Unterordner enthalten kann. So enthält der Ordner `OrgChart` im Projekt `Examples` z.B. weitere Unterordner für XML-, XSL- und Schema-Dateien.



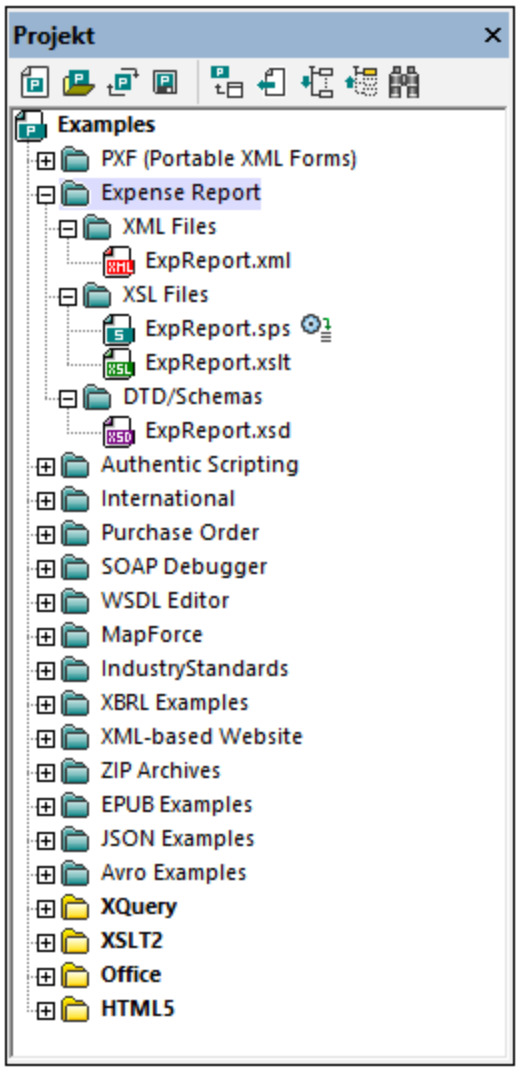
Sie können also in Projekten Dateien ablegen, die gemeinsam verwendet werden, um diese schneller aufrufen zu können. Außerdem können Sie Schemas und XSLT-Dateien für einzelne Ordner definieren und so die Batch-Verarbeitung von Dateien in einem Ordner ermöglichen.

In diesem Abschnitt wird beschrieben, wie man Projekte erstellt und bearbeitet[1081] und wie man Projekte verwendet.[1086]

© 2016-2022 Altova GmbH

21.1 Erstellen eines Projekts

Projekte werden über das Projektfenster[118] (*Abbildung unten*) und das Projektmenü[1316] verwaltet. Es kann immer nur ein Projekt gleichzeitig in der Applikation geöffnet sein. Das offene Projekt wird im Projektfenster[118] angezeigt.



Erstellen neuer Projekte, Öffnen bestehender Projekte

Ein neues Projekt wird mit dem Menübefehl **Projekt | Neues Projekt** erstellt. Ein bestehendes Projekt wird mit dem Menübefehl **Projekt | Projekt öffnen** geöffnet. Das neu geöffnete Projekt (ob nun ein neues oder bestehendes Projekt) tritt im Projektfenster an Stelle des geöffneten Projekts. Wenn das zuvor geöffnete Projekt nicht gespeicherte Änderungen enthält (gekennzeichnet durch ein Sternchen neben dem Ordnernamen; *siehe Abbildung unten*), werden Sie gefragt, ob Sie diese Änderungen speichern möchten.

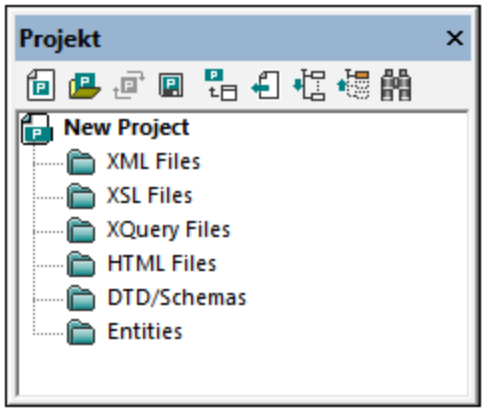
Benennen und Speichern von Projekten

Ein neues Projekt erhält seinen Namen, wenn Sie es speichern. Ein Projekt wird mit dem Befehl **Projekt | Projekt speichern** gespeichert und erhält die Dateierweiterung `.spp`. Nachdem ein Projekt geändert wurde, muss es gespeichert werden, damit die Änderungen erhalten bleiben. Beachten Sie, dass ein Projekt (der Name des Projekts ist der des obersten Ordners im Projektfenster) nur durch Ändern des Namens im Windows Datei Explorer umbenannt werden kann; der Name kann nicht über die Benutzeroberfläche der Applikation selbst geändert werden. (Die Namen von Unterordnern können jedoch über die Benutzeroberfläche der Applikation geändert werden.)

Projektstruktur

Ein Projekt hat eine Baumstruktur bestehend aus Ordnern und Dateien. Ordner und Dateien können in jeder hierarchischen Ebene und mit unbegrenzt vielen Unterebenen angelegt werden. Wählen Sie dazu einen Ordner im Projektfenster aus und verwenden Sie die Befehle im Menü **Projekt** oder im Kontextmenü, um Ordner, Dateien oder Ressourcen hinzuzufügen. Ordner, Dateien und Ressourcen, die zum Projekt hinzugefügt werden, können gelöscht oder an eine andere Stelle in der Projektstruktur verschoben werden.

Wenn ein neues Projekt erstellt wird, wird die Projektstruktur standardmäßig nach Dateityp (XML, XSL, usw.) angelegt (*siehe Abbildung unten*).



Dateityperweiterungen werden mit einem Ordner über die Eigenschaftsdefinitionen für diesen Ordner verknüpft. Wenn eine Datei zu einem Ordner hinzugefügt wird, wird sie automatisch ihrer Erweiterung entsprechend zum entsprechenden untergeordneten Ordner hinzugefügt. Sie können für jeden Ordner festlegen, welche Dateityperweiterungen diesem zugeordnet werden sollen.

Was kann zu einem Projekt hinzugefügt werden

Ordner, Dateien und andere Ressourcen können entweder zum obersten Projektordner oder zu einem beliebigen anderen Ordner in der Projekthierarchie hinzugefügt werden. Es gibt drei Arten von Ordnern: (i) Projektordner; (ii) externe Ordner; (iii) externe Webordner.

Um einen Eintrag hinzuzufügen, wählen Sie den entsprechenden Ordner aus und anschließend den gewünschten Befehl aus dem Menü **Projekt** oder dem Kontextmenü des ausgewählten Ordners. Zu einem Projektordner können die folgenden Einträge hinzugefügt werden:

© 2016-2022 Altova GmbH

- *Projektordner* (grün) sind Ordner, die Sie zu einem Projekt hinzufügen, um den Inhalt des Projekts zu gliedern. Sie können definieren, welche Dateierweiterungen (in den Eigenschaften dieses Ordners) mit einem Projektordner verknüpft werden sollen. Wenn Dateien zu einem Ordner hinzugefügt werden, werden Sie automatisch zum ersten untergeordneten Ordner hinzugefügt, mit dem diese Dateierweiterung verknüpft wurde. Wenn daher mehrere Dateien zu einem Ordner hinzugefügt werden, werden Sie nach ihrer Dateierweiterung in die untergeordneten Ordner eingeordnet.
- *Externe Ordner* (gelb) sind Ordner in einem Dateisystem. Wenn ein externer Ordner zu einem Ordner hinzugefügt wird, werden der externe Ordner sowie alle seine Dateien und Unterordner in das Projekt inkludiert. Wenn Sie für einen externen Ordner Dateierweiterungen definieren, werden diese zum Filtern der im Projekt verfügbaren Dateien verwendet.
- *Externe Webordner* sind wie externe Ordner, mit dem Unterschied, dass sie sich auf einem Webserver befinden und sich der Benutzer für den Zugriff auf diese Ordner authentifizieren muss. Wenn Sie für einen externen Webordner Dateierweiterungen definieren, werden diese zum Filtern der im Projekt verfügbaren Dateien verwendet.
- *Dateien* können zu einem Ordner hinzugefügt werden, indem Sie den Ordner auswählen und anschließend einen von drei entsprechenden Befehlen auswählen: (i) **Dateien hinzufügen**, um die Datei/en über ein Öffnen-Dialogfeld auszuwählen; (ii) **Aktive Datei hinzufügen**, um die im Hauptfenster gerade aktive Datei zum Projekt hinzuzufügen; (iii) **Aktive und verwandte Dateien hinzufügen**, fügt zusätzlich Dateien hinzu, die mit der aktiven XML-Datei in Zusammenhang stehen, z.B. ein XML-Schema oder eine DTD. Beachten Sie, dass Dateien, die über eine Verarbeitungsanweisung (z.B. XLST-Dateien) hinzugefügt werden, nicht als "verwandte" Dateien behandelt werden.
- *Globale Ressourcen* sind Aliasse für Datei-, Ordner und Datenbankressourcen. Eine Beschreibung zur Definition und Verwendung von globalen Ressourcen finden Sie im Abschnitt Globale Ressourcen 1061.
- *URLs* identifizieren ein Ressourcenobjekt über eine URL.
- *Ein Altova Skripting-Projekt:* Dies ist eine `.asprj`-Datei, die einem XMLSpy-Projekt zugewiesen werden kann, wodurch Makros und andere Skripts für das Projekt zur Verfügung stehen. Eine Anleitung zur Erstellung eines Skripting-Projekts und zum Zuweisen eines Skripting-Projekts zu einem XMLSpy-Projekt finden Sie im Abschnitt Skripterstellung 1662.

Projekt- und Ordnereigenschaften

Eigenschaften (wie z.B. das Schema für die Validierung und die XSLT-Datei für die Transformation) können nicht nur für das gesamte Projekt, sondern auch für einzelne Ordner definiert werden. Anschließend können Sie am gesamten Projekt oder einzelnen Ordnern Aktionen wie Validierung und Transformation durchführen. Um eine Aktion durchzuführen, klicken Sie mit der rechten Maustaste auf das Projekt oder den Ordner und wählen Sie im Kontextmenü die gewünschte Aktion aus.

Die Eigenschaften eines Ordners werden im Dialogfeld "Eigenschaften" dieses Ordners gespeichert. Um dieses Dialogfeld aufzurufen, wählen Sie zuerst den Ordner aus und anschließend den Menübefehl **Projekt | Eigenschaften** (oder den Kontextmenübefehl "Eigenschaften" des Ordners. Im Dialogfeld "Eigenschaften" können die folgenden Ordnereigenschaften definiert und bearbeitet werden:

- *Ordnername:* Der Name des Projektordners auf der obersten Ebene kann nicht bearbeitet werden (statt dessen wird dafür ein Dateipfad angezeigt).
- *Dateierweiterungen*: können nicht für die oberste Ebene des Projekts bearbeitet werden. Es handelt sich dabei um eine Liste von durch Kommas getrennten Dateierweiterungen (z.B. **`xml;svg;wml`**). Anhand dieser Liste wird ermittelt, welche Dateien zu welchen Ordnern hinzugefügt werden, wenn Dateien zu einem Projekt hinzugefügt werden. Wenn z.B. wie oben aktive und verwandte Dateien zu einem Projekt hinzugefügt werden, wird anhand der *Dateierweiterungen* des Ordners ermittelt, zu welchem Ordner die Dateien hinzugefügt werden sollen.
- *Validierung*: definiert die DTD bzw. die XML-Schema.Datei, die zur Validierung der XML-Dateien in einem Ordner verwendet werden soll.

- *Transformationen*: definiert (i) die XSLT-Dateien, die zur Transformation der XML-Dateien im Ordner verwendet werden soll und (ii) die XML-Dateien, die mit den XSLT-Dateien im Ordner transformiert werden sollen.
- *Zieldateien*: gibt für die Ausgabe der Transformationen die Dateierweiterung und den Ordner an, in dem die Ausgabedateien gespeichert werden sollen.
- *SPS-Dateien für Authentic-Ansicht*: definiert die SPS-Dateien, die verwendet werden müssen, damit XML-Dateien in einem Ordner in der Authentic-Ansicht angezeigt und bearbeitet werden können.

Beachten Sie die folgenden Punkte:

- Eine für einen Ordner definierte Eigenschaft setzt dieselbe Eigenschaft des Projekts außer Kraft.
- Wenn eine Eigenschaft für ein Projekt definiert wurde, wird sie auf alle Ordner, für die diese Eigenschaft noch nicht definiert wurde, angewendet.
- Wenn eine Aktion an einem Projekt durchgeführt wird, wird sie auf alle anwendbaren Dateitypen in allen Ordnern des Projekts angewendet. Wenn z.B eine Validierung an einem Projekt durchgeführt wird, wird sie an allen XML-Dateien in allen Ordnern des Projekts durchgeführt. In diesem Fall wird das für das Projekt definierte Schema für alle Validierungen mit Ausnahme der Dateien, die sich in Ordnern befinden, für deren Schemavalidierungseigenschaft ein anderes Schema definiert wurde, verwendet.

Nähere Informationen dazu finden Sie in der Beschreibung zu**Projekt | Eigenschaften**[1344]m Befehl .

Versionskontrolle in Projekten

Es werden in Projekten Versionskontrollsysteme, die mit Microsoft Visual Source-Safe kompatibel sind, unterstützt. Eine Anleitung zur Verwendung dieser Funktion finden Sie im Abschnitt Menübefehle[1320] dieses Handbuchs.

Speichern von Projekten

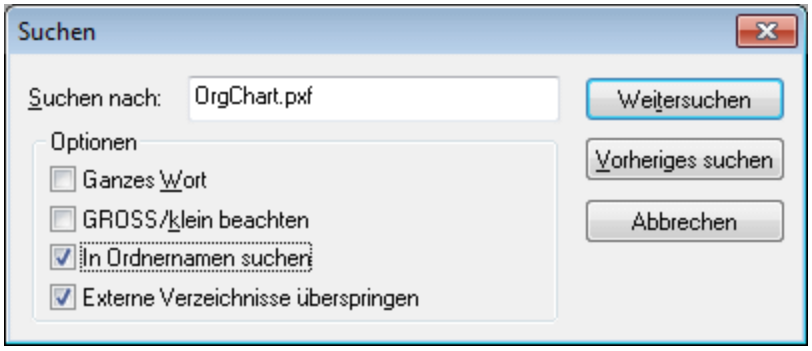
Alle an einem Projekt vorgenommenen Änderungen wie z.B. das Hinzufügen oder Löschen einer Datei oder die Änderung einer Projekteigenschaft müssen mit dem Befehl **Projekt speichern** gespeichert werden.

In Projekt suchen

Sie können Projektdateien und -ordner nach ihrem Namen bzw. einem Teil ihres Namens suchen. Wurden Treffer gefunden, so werden die gefundenen Dateien bzw. Ordner der Reihe nach markiert.

Um eine Suche zu starten, aktivieren Sie das Fenster "Projekt", indem Sie es anklicken (oder hinein klicken), und wählen Sie den Befehl **Bearbeiten | Suchen** (oder das Tastenkürzel **Strg+F**). Geben Sie nun im Dialogfeld "Suchen" (*Abbildung unten*) den gewünschten Such-String ein und aktivieren bzw. deaktivieren Sie die Suchoptionen nach Bedarf (*Erklärung weiter unten*)

© 2016-2022 Altova GmbH



Es stehen die folgenden Suchoptionen zur Verfügung:

- Ganzes Wort: Hierbei ist die Suche stärker eingeschränkt, da der Suchstring mit einem ganzen Wort im Datei- bzw. Ordnernamen übereinstimmen muss. Die Teile vor und nach dem Punkt in den Dateinamen werden (ohne Berücksichtigung des Punkts) jeweils als Wort behandelt.
- Sie können festlegen, dass die Groß- und Kleinschreibung im Suchstring genau mit dem Textstring im Datei- oder Ordnernamen übereinstimmen muss.
- Ordnernamen können in die Suche einbezogen werden. Andernfalls werden nur Dateinamen durchsucht.
- Externe Ordner[1337] können in die Suche einbezogen oder davon ausgeschlossen werden. Bei externen Ordnern handelt es sich um tatsächliche Ordner auf Ihrem Rechner bzw. Netzwerk und nicht um Projektordner, die innerhalb des Projekts und nicht auf dem Rechner angelegt wurden.

Wurden Treffer gefunden, wird der erste Treffer in der Seitenleiste "Projekt" markiert. Sie können anschließend im Dialogfeld "Suchen" mit Hilfe von **Weitersuchen** und **Vorh. suchen** der Reihe nach zu allen gefundenen Suchtreffern springen.

Aktualisieren von Projekten

Wenn eine Änderung an einem externen Ordner vorgenommen wird, wird diese Änderung erst im Projektfenster angezeigt, wenn das Projekt aktualisiert wurde.

21.2 Verwenden von Projekten

Projekte eignen sich gut dazu, Ihre Arbeit zu strukturieren, Einstellungen auf mehrere Dateien anzuwenden und Batch-Befehle einzurichten und auszuführen. Mit Hilfe von Projekten können Sie Ihre Arbeit daher vereinfachen und rationalisieren. Informationen zur Verwaltung von Projekten finden Sie unter Erstellen und Bearbeiten von Projekten[1081] und der Beschreibung des Projektfensters[118].

Vorteile der Verwendung von Projekten

In der folgenden Liste werden die Vorteile der Verwendung von Projekten aufgezählt.

- Dateien und Ordner können nach Dateityp oder anderen Kriterien in Ordnern abgelegt werden.
- Schemas und XSLT-Dateien können einem Ordner zugewiesen werden. Dies ist nützlich, wenn Sie eine einzelne XML-Datei schnell unter Verwendung unterschiedlicher Schemas oder XSLT-Dateien validieren oder transformieren möchten. Fügen Sie die XML-Datei zu verschiedenen Ordnern hinzu und definieren Sie Schemas und XSLT-Dateien für verschiedene Ordner.
- Auf einzelne Ordner kann eine Batch-Verarbeitung angewendet werden. Die für die Batch-Verarbeitung verfügbaren Befehle sind unten aufgelistet.
- Für Transformationen können Ausgabeordner definiert werden.

Organisieren von Ressourcen für den schnellen Zugriff

Ordner- und Dateiressourcen können in einer hierarchischen Struktur organisiert werden, sodass Sie einen klaren Überblick über die verschiedenen Ordner und Dateien in Ihren Projekt haben und Sie jederzeit schnell auf eine oder alle Dateien in einem Projekt zugreifen können. Doppelklicken Sie einfach im Projektfenster auf eine Datei um sie zu öffnen. Sie können Dateien und Ordner bei Bedarf schnell zu einem Projekt hinzufügen oder nicht benötigte Dateien und Ordner löschen. Wenn Sie mit einem anderen Projekt arbeiten möchten, schließen Sie das aktuell im Projektfenster geöffnete Projekt und öffnen Sie das gewünschte Projekt.

Batch-Verarbeitung

Die Befehle für die Batch-Verarbeitung von Dateien in einem Ordner - egal ob für den obersten Ordner im Projekt oder einen untergeordneten Ordner - stehen im **Kontextmenü dieses Ordners zur Verfügung** (das Kontextmenü wird durch Rechtsklick auf den Ordner geöffnet). Die Schritte bei der Batch-Verarbeitung sind die folgenden:

1. Definieren Sie im Dialogfeld "Eigenschaften" dieses Ordners die Dateien, die für die Validierung oder Transformation verwendet werden sollen.
2. Definieren Sie den Ordner, in dem die Ausgabe der Transformationen gespeichert werden sollen. Wenn für einen Ordner kein Ausgabeordner definiert wurde, wird der Ausgabeordner des nächsten übergeordneten Ordners in der Projektstruktur verwendet.
3. Verwenden Sie zur Batch-Verarbeitung die Befehle im Kontextmenü. Wenn Sie die entsprechenden Befehle in den Menüs **XML, DTD/Schema**, oder **XSL/XQuery** verwenden, wird der Befehl nur am aktiven Dokument im Hauptfenster ausgeführt und nicht an einem Projektordner im Projektfenster.

Im Kontextmenü eines Projektordners (oberste Ebene und darunter) stehen die folgenden Befehle für die Batch-Verarbeitung zur Verfügung:

- *Wohlgeformtheit prüfen*: Wenn während der Batch-Verarbeitung Fehler gefunden werden, werden diese im Fenster "Meldungen" ausgegeben.

© 2016-2022 Altova GmbH

- *Validierung*: Wenn während der Batch-Verarbeitung Fehler gefunden werden, werden diese im Fenster "Meldungen" ausgegeben
- *Transformationen*: Transformationsausgaben werden in dem Ordner gespeichert, der im Dialogfeld "Eigenschaften" dieses Ordners als Ausgabeordner angegeben ist. Wenn für einen Ordner kein Ausgabeordner definiert wurde, wird der Ausgabeordner des nächsten übergeordneten Ordners in der Projektstruktur verwendet. Wenn es keinen übergeordneten Projektordner gibt, für den ein Ausgabeordner definiert wurde, wird ein Dokumentfenster geöffnet, in dem der Reihe nach die Ergebnisse der einzelnen Transformationen angezeigt werden. Mit einer XSL-FO-Transformation wird ein XML-Dokument oder ein FO-Dokument in ein PDF-Dokument transformiert.
- *DTD / XML Schema generieren*: Bevor die Schemas generiert werden, werden Sie aufgefordert einen Ausgabeordner anzugeben. Die generierten Schema-Dateien werden in diesem Ordner gespeichert und auf der Benutzeroberfläche in separaten Fenstern angezeigt.

Anmerkung: Verwenden Sie zur Ausführung von Batch-Befehlen das Kontextmenü des entsprechenden Ordners im Projektfenster. Verwenden Sie nicht die Befehle in den Menüs XML, DTD/Schema, oder XSL/XQuery. Diese Befehle werden an dem im Hauptfenster aktiven Dokument ausgeführt.

Validierung und XSLT/XQuery mit RaptorXML Server

Mit Hilfe der Kontextmenübefehle im Projektordner können Sie die XML-Validierung und XSLT/XQuery-Transformationen mit RaptorXML Server mit hoher Geschwindigkeit ausführen. Nähere Informationen dazu finden Sie im Abschnitt RaptorXML Server[1088].

22 RaptorXML(+XBRL) Server

Wenn Altova RaptorXML(+XBRL) Server (in der Folge auch als RaptorXML Server, RaptorXML oder kurz Raptor bezeichnet) in Ihrem Netzwerk installiert und lizenziert ist und wenn Ihre XMLSpy-Installation Zugriff darauf hat, können Sie Ihre XML- und XBRL*-Dokumente mit Hilfe von RaptorXML Server validieren sowie XSLT- und XQuery-Transformationen[1109] daran ausführen. Sie können entweder das aktive Dokument oder alle Dokumente im XMLSpy-Projektordner validieren. Die Validierungsergebnisse werden im Fenster "Meldungen" der Benutzeroberfläche von XMLSpy angezeigt.

Sie können Dokumente in XMLSpy (i) validieren, (ii) daran eine XSLT/XQuery-Transformation ausführen oder (iii) XULE-Dokumente (oder Dokumentenmappen) an einer XBRL-Instanz ausführen. Einer der Hauptvorteile der Verwendung von Raptor ist, dass Sie einzelne Validierungseinstellungen oder Ausführungen mittels zahlreicher Validierungsoptionen konfigurieren können. Außerdem können Sie eine Gruppe von Raptor-Optionen in XMLSpy als "Konfiguration" speichern und anschließend eine der von Ihnen definierten Konfigurationen für eine bestimmte Raptor-Valdierung auswählen. Auch bei großen Datenmengen ist es von Vorteil, diese mit Raptor zu validieren.

Anmerkung: Die tatsächliche Leistung hängt von der Anzahl der PC-Prozessorkerne ab, die von RaptorXML Server für die Validierung verwendet werden: Je mehr Kerne verwendet werden, desto schneller ist die Verarbeitung.

***Anmerkung:** Es gibt zwei Editionen von Raptor: *RaptorXML Server* (für XML-Validierungen) und *RaptorXML+XBRL Server* (für XML- und XBRL-Validierungen). Zur Valdierung von XBRL-Dokumenten benötigen Sie RaptorXML+XBRL Server. Nähere Informationen zu RaptorXML (+XBRL) Server finden Sie auf der Altova Website und in den Benutzerhandbüchern: *RaptorXML Server* und *RaptorXML+XBRL Server*.

Anmerkung: RaptorXML Server kann nicht mit HTTP Proxies verwendet werden, da diese Websocket Upgrades nicht unterstützen. Falls dieses Problem bei Ihnen auftritt, fügen Sie den RaptorXML Server Host zur Proxy-ignorieren-Liste hinzu.

So validieren oder transformieren Sie Dokumente mit Hilfe von RaptorXML Server

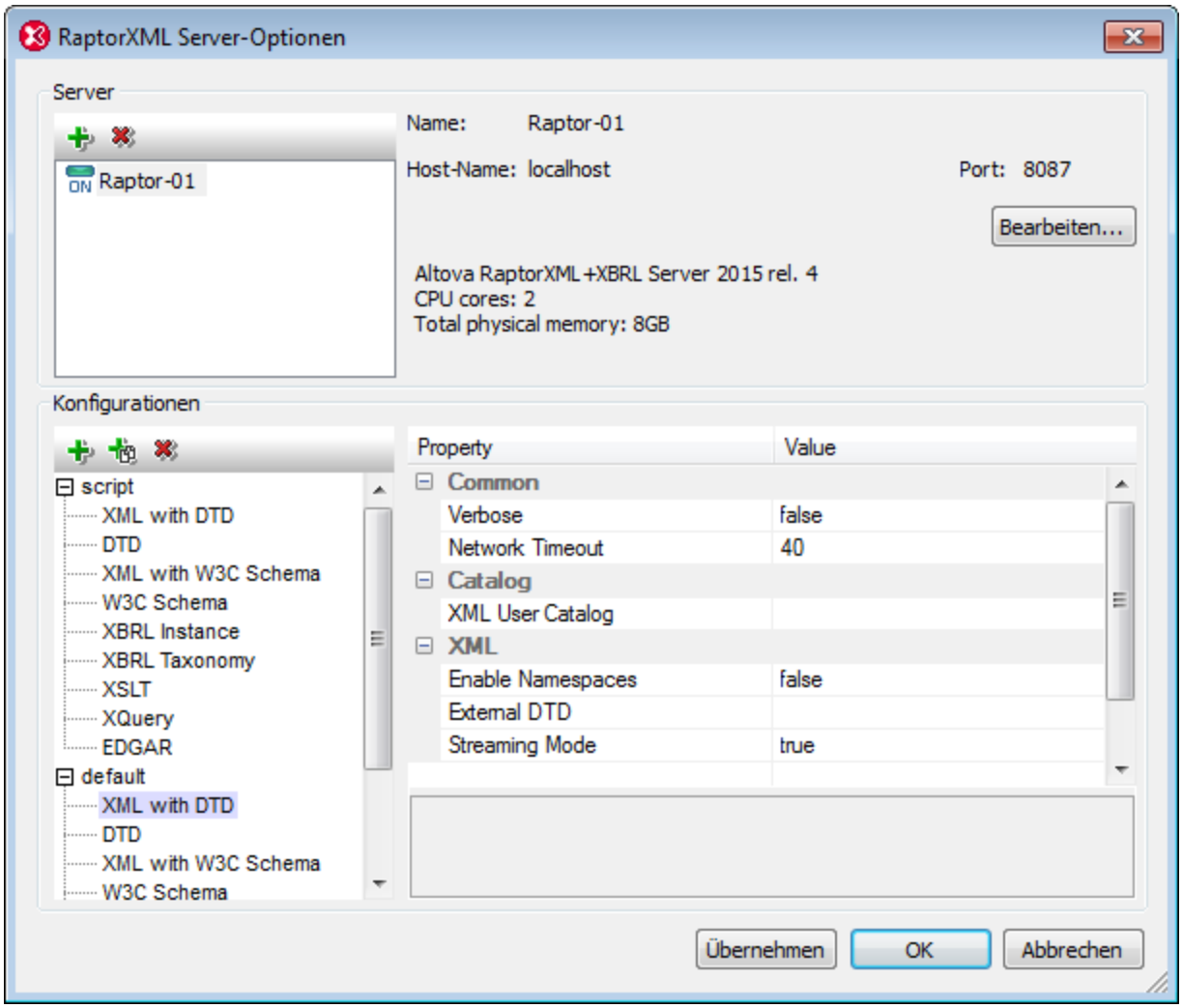
Um ein XML- oder XBRL-Dokument mit RaptorXML Server zu validieren oder eine XSLT- oder XQuery-Transformation auszuführen, muss in XMLSpy definiert werden, welcher RaptorXML Server verwendet werden soll, wie das Programm auf diesen Server zugreifen soll und welche Optionen für die Validierung an Raptor übergeben werden sollen. Dazu müssen Sie in XMLSpy die folgenden Informationen angeben:

1. *Hinzufügen eines Servers zum Pool der Raptor-Server*[1089]*:* In diesem Schritt werden RaptorXML Server zu einem Pool hinzugefügt. Die Zugriffsinformationen auf die einzelnen Server werden in XMLSpy gespeichert. Jeder Server wird anhand eines Namens identifiziert.
2. *Definition der Konfiguration für die einzelnen Server*[1090]*:* Eine Konfiguration ist eine Gruppe von Raptor-Validierungsoptionen. Jeder Server kann mehrere Konfigurationen haben. Um eine Validierung durchzuführen, müssen Sie eine der Konfigurationen auswählen. Diese wird zur aktiven Konfiguration.
3. *Auswahl einer Server-Konfiguration, mit der die Validierung durchgeführt werden soll*[1093]*:* Ein Server und eine seiner Konfigurationen wird als aktive Konfiguration ausgewählt. Diese aktive Konfiguration wird in der Folge für alle Validierungen, in denen Raptor verwendet wird, angewendet.
4. *Validierung*[1093] *oder Ausführung einer XSLT/XQuery-Transformation*[1109] *mit Raptor.*

© 2016-2022 Altova GmbH

22.1 Hinzufügen von Servern und Server-Konfigurationen

Im Dialogfeld "RaptorXML Server-Optionen" (*siehe Abbildung unten,* **Extras | Raptor-Server verwalten**[1579]) können Sie zum Pool der verfügbaren Server mehrere Raptor-Server hinzufügen[1089] und anschließend mehrere Konfigurationen[1090] für die einzelnen Server definieren. Die hinzugefügten Server werden zusammen mit den Konfigurationen, die Sie für jeden davon definiert haben, im Untermenü **Extras | Raptor-Server und Konfiguration**[1583] angezeigt. Sie können in diesem Untermenü die gewünschte Server-Konfiguration für eine Raptor-Validierung auswählen.



Hinzufügen eines Raptor-Servers

Klicken Sie im Bereich *Server* des Dialogfelds (*Abbildung unten*) auf das Symbol **Server hinzufügen** und geben Sie anschließend den Namen, anhand dessen der Raptor-Server identifiziert werden soll, den Netzwerk-Namen des Rechners, auf dem Raptor installiert ist (Host-Name) und den Port des Raptor-Servers ein. Klicken Sie auf **OK**, um die Einstellungen zu speichern.



- *Name:* Hier können Sie einen beliebigen String eingeben. Anhand des Namens wird in XMLSpy ein bestimmter RaptorXML-Server identifiziert.
- *Host-Name:* Der Name oder die IP-Adresse des Netzwerk-Rechners, auf dem der Raptor-Server installiert ist. Die Vearbeitung erfolgt schneller, wenn anstelle eines Host-Namens eine IP-Adresse verwendet wird. Die IP-Adresse für `localhost` (Ihren lokalen Rechner) ist **`127.0.0.1`**.
- *Port:* Der Port, über den der Zugriff auf den Raptor-Server erfolgt. Dieser Port wird in der Konfigurationsdatei von Raptor (`server_config.xml`) definiert. Der Port muss fix und bekannt sein, damit Requests korrekt an den Dienst adressiert werden können. Nähere Informationen zur Raptor-Konfigurationsdatei finden Sie in den Benutzerhandbüchern *RaptorXML Server* und *RaptorXML+XBRL Server*.

Klicken Sie nach Eingabe der Server-Informationen auf **OK**. Der Server-Name, den Sie eingegeben haben, wird (im linken Bereich) in der Liste der Server angezeigt. Neben dem Namen des Servers wird ein grünes Symbol angezeigt. Damit wird gekennzeichnet, dass der Raptor-Server gestartet wurde und ausgeführt wird. Die Details zum Server werden im Bereich angezeigt (*siehe Abbildung oben*). Ein rotes Symbol bedeutet, dass der Server offline ist. Wenn der Server nicht gefunden wurde, wird eine Fehlermeldung angezeigt.

Anmerkung: Der Raptor-Server muss ausgeführt werden, wenn der Server hinzugefügt wird, damit XMLSpy Informationen über den Server abrufen und speichern kann. Wenn der Server, nachdem er hinzugefügt wurde, offline ist oder nicht gefunden wird, so wird dies durch ein rotes Symbol bzw. eine Fehlermeldung angezeigt.

Um den Namen, Host-Namen oder Port eines Servers zu bearbeiten, klicken Sie auf die Schaltfläche **Bearbeiten** und bearbeiten Sie die entsprechenden Informationen im daraufhin angezeigten Dialogfeld nach Bedarf. Um einen Server aus dem Pool zu entfernen, wählen Sie den Server aus und klicken Sie auf das Symbol **Ausgewählten Server entfernen**.

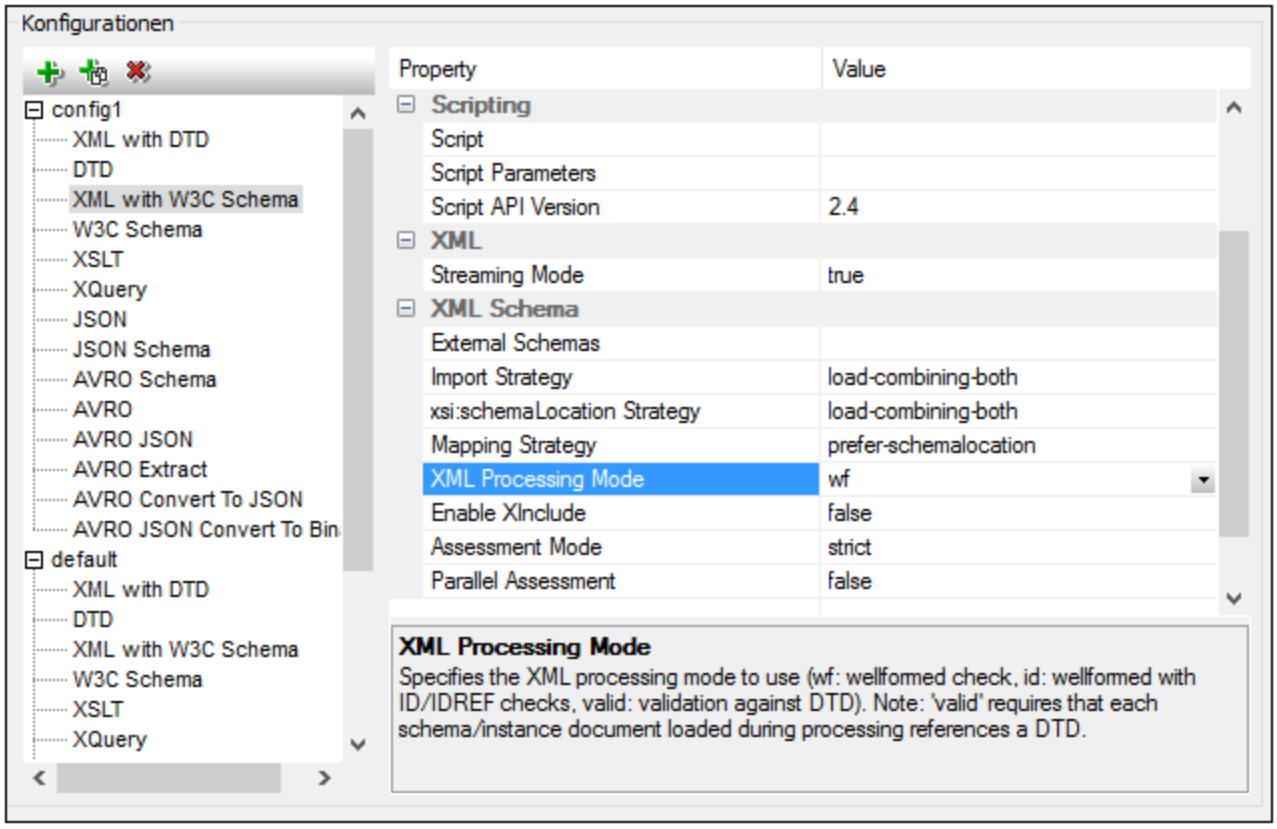
Server-Konfigurationen

Eine Konfiguration ist eine Gruppe von RaptorXML-Validierungsoptionen. Wenn ein Server hinzugefügt wird, erhält er eine Konfiguration namens `default`. Dabei handelt es sich um eine Gruppe von RaptorXML-Optionen, für die als Optionen die Standardeinstellungen definiert sind. Sie können neue Konfigurationen hinzufügen, die andere Optionswerte enthalten. Nachdem Sie mehrere Server-Konfigurationen definiert haben, können Sie eine dieser Konfigurationen als aktive Konfiguration auswählen. Dies ist die Konfiguration, die bei Aktivierung des Befehls **Am Server validieren** verwendet wird.

Der Bereich *Konfigurationen* besteht aus zwei Teilen: (i) einem linken Bereich, in dem die Konfigurationen jeweils mit einer Liste von validierbaren Dokumenttypen aufgelistet werden; (ii) einem rechte Bereich, in dem die Validierunsoptionen des im linken Bereich ausgewählten Dokumenttyps angezeigt werden; am unteren

 © 2016-2022 Altova GmbH

Rand des rechten Bereichs sehen Sie eine Beschreibung der ausgewählten Option (*siehe Abbildung unten*).



Hinzufügen einer Konfiguration

Klicken Sie im Bereich *Konfigurationen* des Dialogfelds "RaptorXML Server-Optionen" auf **Konfiguration hinzufügen** (*Abbildung oben*). Daraufhin wird eine neue Konfiguration mit den Standardoptionswerten hinzugefügt. Sie können auch auf **Ausgewählte Konfiguration kopieren** klicken, um eine neue Konfiguration zu erstellen. Auf diese Art erstellen Sie eine neue Konfiguration mit den Optionswerten der kopierten Konfiguration. Neue Konfigurationen werden mit Standardnamen vom Typ `config<X>` erstellt. Sie können den Namen einer Konfiguration bearbeiten, indem Sie auf den Namen doppelklicken und einen neuen Namen eingeben. Sie können jeden der Optionswerte der Konfiguration bearbeiten.

Bearbeiten der Optionswerte der Konfiguration

Wählen Sie zuerst den Dokumenttyp im linken Bereich aus. Daraufhin werden die Validierungsoptionen des ausgewählten Dokumenttyps im rechten Bereich angezeigt. Um den Wert einer Option zu bearbeiten, wählen Sie (abhängig vom Typ des Optionswerts) eine der folgenden Methoden:

- Wenn es sich beim Wert um einen aus einer Gruppe vordefinierter Werte handeln kann, wählen Sie den gewünschten Wert aus der Auswahlliste der Wertespalte dieser Option aus.
- Wenn es für den Wert keine Einschränkungen gibt, klicken Sie in das Wertefeld der Option und geben Sie den gewünschten Wert ein.
- Wenn es sich beim Wert um einen Dateipfad handelt, haben Sie zusätzlich zur Eingabe des Werts auch die Möglichkeit, über die Schaltfläche **Durchsuchen** in der Wertespalte der Option zur gewünschten Datei zu navigieren.

Wenn Sie eine Option auswählen, wird die Beschreibung dazu in einem Feld im rechten unteren Bereich angezeigt. Ausführlichere Beschreibungen zu den einzelnen Optionen finden Sie in den Benutzerhandbüchern zu RaptorXML Server und RaptorXML(+XBRL) Server in den Kapiteln zu den Befehlszeilenoptionen.

Entfernen einer Konfiguration
Wählen Sie die zu entfernende Konfiguration im linken Bereich aus und klicken Sie auf **Ausgewählte Konfiguration entfernen**.

XMLSpy in Visual Studio und Eclipse

Wenn XMLSpy in Visual Studio[1141] und Eclipse[1147] integriert wurde, so ist die aktive Konfiguration in diesen IDEs diejenige, die in der Standalone-Version von XMLSpy gerade als aktive Konfiguration definiert ist.

© 2016-2022 Altova GmbH

22.2 Validierung mit RaptorXML Server

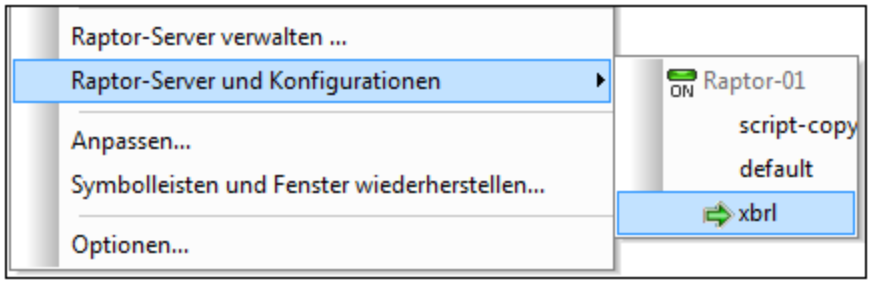
Sie können XML- und XBRL*-Dokumente mit RaptorXML Server validieren. Dies erfordert zwei Schritte:

- Auswählen des Servers und der Server-Konfiguration für die Validierung
- Ausführen der Validierung (mittels eines der **Am Server validieren**- Befehle; siehe unten)

***Anmerkung:** Es gibt zwei Editionen von Raptor: *RaptorXML Server* (für XML-Validierungen) und *RaptorXML+XBRL Server* (für XML- und XBRL-Validierungen). Zur Valdierung von XBRL-Dokumenten benötigen Sie RaptorXML+XBRL Server. Nähere Informationen zu RaptorXML (+XBRL) Server finden Sie auf der Altova Website und in den Benutzerhandbüchern: *RaptorXML Server* und *RaptorXML+XBRL Server*.

Auswählen der gewünschten Server-Konfiguration

Wenn Sie mehrere Konfigurationen auf mehreren Servern definiert haben, können Sie einen Server und eine seiner Konfigurationen als aktive Konfiguration auswählen. Die aktive Konfiguration wird in der Folge für Validierungen verwendet. Wenn Sie den Cursor über den Befehl **Extras | Raptor-Server und Konfigurationen** (*siehe Abbildung unten*) platzieren, wird ein Untermenü mit allen hinzugefügten Servern und deren Konfiguration angezeigt. Wählen Sie die Konfiguration, die Sie zur aktiven Konfiguration machen wollen, aus. In der Abbildung unten wurde die Konfiguration `xbrl` des Servers mit dem Namen `Raptor-01` als aktive Konfiguration ausgewählt (gekennzeichnet durch einen grünen Pfeil).



Validierung mit RaptorXML Server

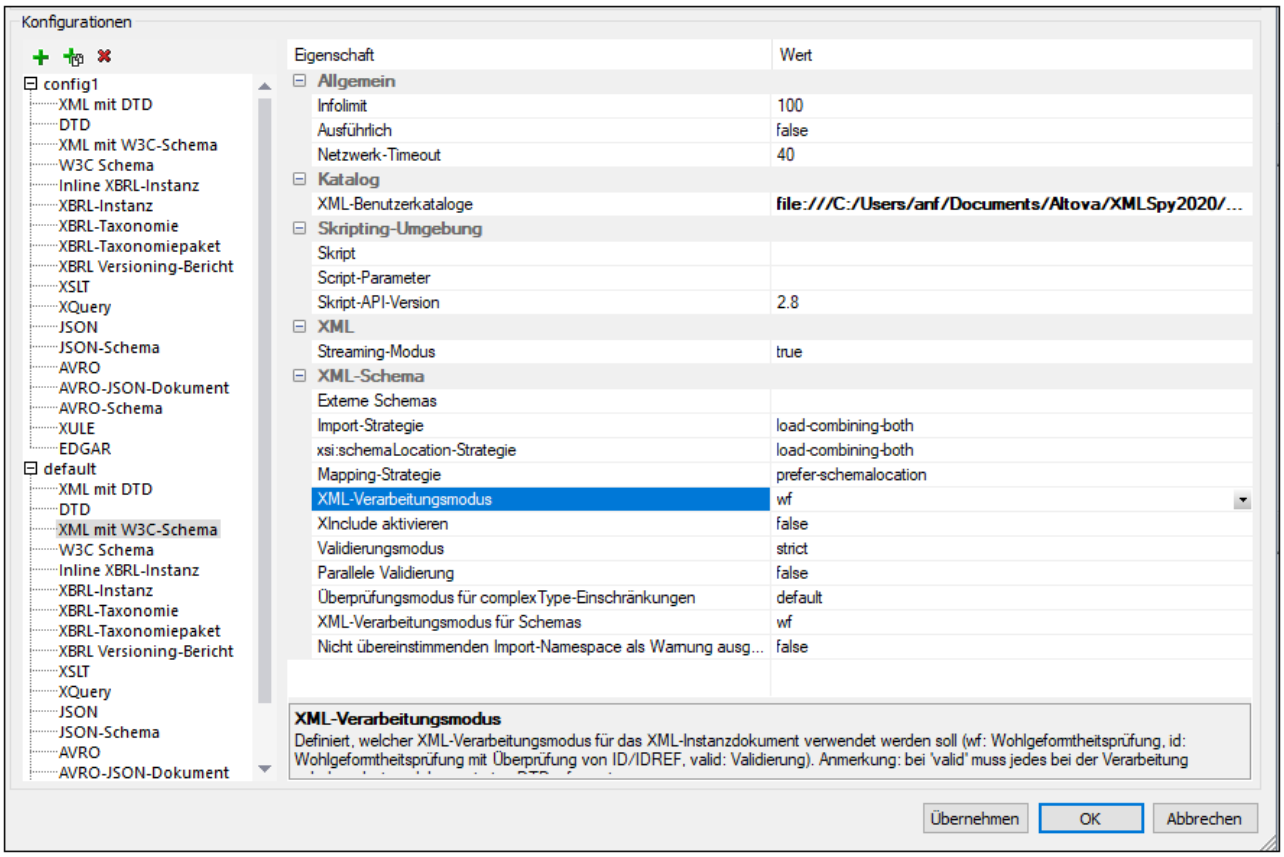
Sie können XML- und XBRL-Dokumente mit Hilfe der Validierungsprozessoren von XMLSpy oder mit Hilfe von RaptorXML Server validieren. Um RaptorXML Server zu validieren, wählen Sie eine der folgenden Methoden:

- Klicken Sie auf die Symbolleisten-Schaltfläche **Am Server validieren**
- Wählen Sie den Befehl **XML | XML am Server validieren (Hochleistungsvalidierung) (Strg+F8)**
- Klicken Sie im Fenster "Projekt" mit der rechten Maustaste auf das Projekt, einen Ordner oder eine Datei und wählen Sie den Befehl **XML am Server validieren**, um XML- oder XBRL-Daten im ausgewählten Objekt zu validieren.

Anmerkung: Die Raptor-Validierung steht in der Textansicht, der Schema-Ansicht, der XBRL-Ansicht und der Grid-Ansicht zur Verfügung.

22.3 Validierungsoptionen

Dieser Abschnitt ist nach dem Typ des zu validierenden Dokuments geordnet (siehe linker Bereich in der Abbildung unten). So wird z.B. mit *XML mit W3C Schema* ein XML-Dokument anhand eines W3C XML-Schemas validiert. Wenn ein Validierungstyp im linken Bereich ausgewählt ist, werden die RaptorXML-Validierungsoptionen zu dieser Validierungsart im rechten Bereich angezeigt. Diese Optionen sind in Gruppen geordnet wie z.B. *Scripting* und *XML Schema* (*siehe Abbildung unten*). Beachten Sie, dass in der Professional Edition nicht alle in der Abbildung gezeigten Gruppen (z.B. XBRL-Gruppen) zur Verfügung stehen.



Die Unterabschnitte dieses Abschnitts enthalten Links zur Beschreibung der jeweiligen RaptorXML Server-Validierungsoption.

22.3.1 Allgemeine Optionen

Optionen, die für alle Validierungstypen verwendet werden.

- Allgemeine
  - Info-Limit
  - Ausführlich
  - Netzwerk-Timeout

© 2016-2022 Altova GmbH

- Katalog

  XML-Benutzerkatalog

22.3.2 XML with DTD

Optionen für die Validierung von XML-Daten anhand einer DTD.

- Allgemeine

  Info-Limit
  Ausführlich
  Netzwerk-Timeout

- Katalog

  XML-Benutzerkatalog

- XML

  Namespaces aktivieren
  Externe DTD
  Streaming-Modus

22.3.3 DTD

Optionen für die Validierung von DTDs.

- Allgemeine

  Info-Limit
  Ausführlich
  Netzwerk-Timeout

- Katalog

  XML-Benutzerkatalog

- Skripting

  Skript
  Skript-Parameter
  API-Skriptversion

22.3.4 XML with W3C Schema

Optionen für die Validierung von XML-Daten anhand eines XML-Schemas.

- Allgemeine
  - Info-Limit
  - Ausführlich
  - Netzwerk-Timeout
- Katalog
  - XML-Benutzerkatalog
- Skripting
  - Skript
  - Skript-Parameter
  - API-Skriptversion
- XML
  - Streaming-Modus
- XML-Schema
  - Externe Schemas (xsd)
  - Import-Strategie (schema-imports)
  - xsi:schemaLocation-Strategie (schemalocation-hints)
  - Mapping-Strategie (schema-mapping)
  - XML-Verarbeitungsmodus (xml-mode)
  - XInclude aktivieren (xinclude)
  - Beurteilungsmodus
  - Parallele Beurteilung
  - Überprüfungsmodus für ComplexType-Einschränkungen
  - XML-Verarbeitungsmodus für Schemas
  - Nicht übereinstimmende importierte Namespaces als Warnung ausgeben

22.3.5 W3C Schema

Optionen für die Validierung von XML-Schemas.

- Allgemeine
  - Info-Limit
  - Ausführlich
  - Netzwerk-Timeout

© 2016-2022 Altova GmbH

- Katalog

  XML-Benutzerkatalog

- Skripting

  Skript
  Skript-Parameter
  API-Skriptversion

- XML-Schema

  Import-Strategie (schema-imports)
  xsi:schemaLocation-Strategie (schemalocation-hints)
  Mapping-Strategie (schema-mapping)
  XInclude aktivieren (xinclude)
  Überprüfungsmodus für ComplexType-Einschränkungen
  XML-Verarbeitungsmodus für Schemas
  Nicht übereinstimmende importierte Namespaces als Warnung ausgeben

22.3.6 Inline XBRL-Instanz

Optionen für die Validierung von Inline-XBRL-Instanzdokumenten.

- Allgemeine

  Info-Limit
  Ausführlich
  Netzwerk-Timeout

- Katalog

  XML-Benutzerkatalog

- Skripting

  Skript
  Skript-Parameter
  API-Skriptversion

- XML-Schema

  Import-Strategie (schema-imports)
  xsi:schemaLocation-Strategie (schemalocation-hints)
  Mapping-Strategie (schema-mapping)
  XInclude aktivieren (xinclude)
  Parallele Beurteilung
  Überprüfungsmodus für ComplexType-Einschränkungen
  Nicht übereinstimmende importierte Namespaces als Warnung ausgeben

- XBRL

  Dimensions-Erweiterung aktivieren (dimensions)
  Extensible Enumerations-Erweiterung aktivieren (extensible-enumerations)
  Unit Registry-Erweiterungen aktivieren
  XBRL-Spec-Schemas im Vorhinein laden (preload-xbrl-schemas)
  Taxonomiepakete
  Taxonomiepakete Konfig-Datei
  Nur referenziertes DTS validieren
  XBRL-Inkonsistenzen als Fehler behandeln (treat-inconsistencies-as-errors)
  UTR-Datei
  Unterstützter UTR-Status
  Zusätzlicher DTS-Eintrittspunkt
  URI-Transformationsstrategie (in Ausgabedokumenten)
  Summation-Item-Inkonsistenzen melden
  Essence-Alias-Inkonsistenzen melden
  Requires-Element-Inkonsistenzen melden
  Generic Preferred Label-Erweiterung aktivieren
  Generic Links-Erweiterung aktivieren
  De-duplizieren
  Duplikate melden
  Duplikate melden Schweregrad

- Inline XBRL

  Inline XBRL-Version (ixbrl-version)
  Inline XBRL-Transformations-Registry (transformation-registry)
  Argumente als Inline XBRL Document Set behandeln (document-set)
  Zieldokumentvalidierung aktivieren (validate-xbrl)
  Zieldokument-Ausgabedatei (xbrl-output)
  Normalisierung nicht numerischer Whitespaces
  Erweiterte Whitespace-Normalisierung

- XBRL Formula

  Formula-Erweiterung aktivieren (formula)
  Assertion Severity-Erweiterung (assertion-severity) aktivieren
  Formula-Spec-Schemas im Vorhinein laden (preload-formula-schemas)
  Nicht erfüllte Assertion-Auswertungen melden
  Sprache der Validierungsmeldungen (message-lang)
  Role der Validierungsmeldungen (message-role)
  Formulas-to-Ignore-Datei
  Formulas-to-Process-Datei
  Assertions-to-Ignore-Datei
  Assertions-to-Process-Datei
  Formulas-to-Ignore
  Assertions-to-Ignore
  Formula-Ausgabe validieren
  Formula-Optimierungen aktivieren

- XBRL Table

  Table-Erweiterung aktivieren (table)

© 2016-2022 Altova GmbH

Table-Spec-Schemas im Vorhinein laden (preload-table-schemas)
Table Linkbase Namespace
Table AspectNode-Reihenfolge

- XBRL XULE

XULE
XULE-Stapelgröße
XULE-Instanz-Namespace Bindings
Zu verarbeitende XULE-Regeln
Auswertungen der XULE-Regeln ausgeben
XULE-Ausgabedatei

22.3.7 XBRL-Instanz

Optionen für die Validierung von XBRL-Instanzdokumenten.

- Allgemeine

Info-Limit
Ausführlich
Netzwerk-Timeout

- Katalog

XML-Benutzerkatalog

- Skripting

Skript
Skript-Parameter
API-Skriptversion

- XML-Schema

Import-Strategie (schema-imports)
xsi:schemaLocation-Strategie (schemalocation-hints)
Mapping-Strategie (schema-mapping)
XInclude aktivieren (xinclude)
Parallele Beurteilung
Überprüfungsmodus für ComplexType-Einschränkungen
Nicht übereinstimmende importierte Namespaces als Warnung ausgeben

- XBRL

Dimensions-Erweiterung aktivieren (dimensions)
Extensible Enumerations-Erweiterung aktivieren (extensible-enumerations)
Unit Registry-Erweiterungen aktivieren

XBRL-Spec-Schemas im Vorhinein laden (preload-xbrl-schemas)
Taxonomiepakete
Taxonomiepakete Konfig-Datei
Nur referenziertes DTS validieren
XBRL-Inkonsistenzen als Fehler behandeln (treat-inconsistencies-as-errors)
UTR-Datei
Unterstützter UTR-Status
Zusätzlicher DTS-Eintrittspunkt
URI-Transformationsstrategie (in Ausgabedokumenten)
Summation-Item-Inkonsistenzen melden
Essence-Alias-Inkonsistenzen melden
Requires-Element-Inkonsistenzen melden
Generic Preferred Label-Erweiterung aktivieren
Generic Links-Erweiterung aktivieren
De-duplizieren
Duplikate melden
Duplikate melden Schweregrad

- XBRL Formula

Formula-Erweiterung aktivieren (formula)
Assertion Severity-Erweiterung (assertion-severity) aktivieren
Formula-Spec-Schemas im Vorhinein laden (preload-formula-schemas)
Nicht erfüllte Assertion-Auswertungen melden
Sprache der Validierungsmeldungen (message-lang)
Role der Validierungsmeldungen (message-role)
Formulas-to-Ignore-Datei
Formulas-to-Process-Datei
Assertions-to-Ignore-Datei
Assertions-to-Process-Datei
Formulas-to-Ignore
Assertions-to-Ignore
Formula-Ausgabe validieren
Formula-Optimierungen aktivieren

- XBRL Table

Table-Erweiterung aktivieren (table)
Table-Spec-Schemas im Vorhinein laden (preload-table-schemas)
Table Linkbase Namespace
Table AspectNode-Reihenfolge

- XBRL XULE

XULE
XULE-Stapelgröße
XULE-Instanz-Namespace Bindings
Zu verarbeitende XULE-Regeln
Auswertungen der XULE-Regeln ausgeben
XULE-Ausgabedatei

© 2016-2022 Altova GmbH

22.3.8 XBRL-Taxonomie

Optionen für die Validierung von XBRL-Taxonomien.

- Allgemeine
  - Info-Limit
  - Ausführlich
  - Netzwerk-Timeout
- Katalog
  - XML-Benutzerkatalog
- Skripting
  - Skript
  - Skript-Parameter
  - API-Skriptversion
- XML-Schema
  - Import-Strategie (schema-imports)
  - xsi:schemaLocation-Strategie (schemalocation-hints)
  - Mapping-Strategie (schema-mapping)
  - XInclude aktivieren (xinclude)
  - Überprüfungsmodus für ComplexType-Einschränkungen
  - Nicht übereinstimmende importierte Namespaces als Warnung ausgeben
- XBRL
  - Dimensions-Erweiterung aktivieren (dimensions)
  - Extensible Enumerations-Erweiterung aktivieren (extensible-enumerations)
  - XBRL-Spec-Schemas im Vorhinein laden (preload-xbrl-schemas)
  - Taxonomiepakete
  - Taxonomiepakete Konfig-Datei
  - XBRL-Inkonsistenzen als Fehler behandeln (treat-inconsistencies-as-errors)
  - Generic Preferred Label-Erweiterung aktivieren
  - Generic Links-Erweiterung aktivieren
- XBRL Formula
  - Formula-Erweiterung aktivieren (formula)
  - Assertion Severity-Erweiterung (assertion-severity) aktivieren
  - Formula-Spec-Schemas im Vorhinein laden (preload-formula-schemas)
- XBRL Table
  - Table-Erweiterung aktivieren (table)
  - Table-Spec-Schemas im Vorhinein laden (preload-table-schemas)
  - Table Linkbase Namespace

Table AspectNode-Reihenfolge

22.3.9 XBRL-Taxonomiepaket

Optionen für die Validierung von XBRL-Taxonomiepaketen:

- Allgemeine

  Info-Limit
  Ausführlich
  Netzwerk-Timeout

- Katalog

  XML-Benutzerkatalog

- Skripting

  Skript
  Skript-Parameter
  API-Skriptversion

- XML-Schema

  Import-Strategie (schema-imports)
  xsi:schemaLocation-Strategie (schemalocation-hints)
  Mapping-Strategie (schema-mapping)

22.3.10 XBRL Versioning Report

Optionen für die Validierung von XBRL Versioning.

- Allgemeine

  Info-Limit
  Ausführlich
  Netzwerk-Timeout

- Katalog

  XML-Benutzerkatalog

- XML-Schema

  Import-Strategie (schema-imports)
  xsi:schemaLocation-Strategie (schemalocation-hints)
  Mapping-Strategie (schema-mapping)

© 2016-2022 Altova GmbH

XInclude aktivieren (xinclude)

22.3.11 XSLT

Optionen für die Validierung von XSLT-Dokumenten.

- Allgemeine
  - Info-Limit
  - Ausführlich
  - Netzwerk-Timeout
- Katalog
  - XML-Benutzerkatalog
- XML-Schema
  - Import-Strategie (schema-imports)
  - xsi:schemaLocation-Strategie (schemalocation-hints)
  - Mapping-Strategie (schema-mapping)
  - XML-Verarbeitungsmodus (xml-mode)
  - XInclude aktivieren (xinclude)
- Java-Erweiterung
  - Java-Erweiterunge deaktivieren (javaext-disable)
  - Pfad zur Barcode-Erweiterung (javaext-barcode-location)
- Diagramm-Erweiterungen
  - Diagramm-Erweiterungen deaktivieren (chartext-disable)
- .NET-Erweiterungen
  - .NET-Erweiterungen deaktivieren (dotnetext-disable)
- Allgemeine Optionen XEngines
  - XML mit PSVI laden (load-xml-with-psvi)
- XSLT
  - XSLT-Prozessorversion (xslt-version)
  - Vorlagenmodus
  - Vorlagen-Eintrittspunkt

22.3.12 XQuery

Optionen für die Validierung von XQuery-Dokumenten.

- Allgemeine
  - Info-Limit
  - Ausführlich
  - Netzwerk-Timeout
- Katalog
  - XML-Benutzerkatalog
- XML-Schema
  - Import-Strategie (schema-imports)
  - xsi:schemaLocation-Strategie (schemalocation-hints)
  - Mapping-Strategie (schema-mapping)
  - XML-Verarbeitungsmodus (xml-mode)
  - XInclude aktivieren (xinclude)
- Java-Erweiterung
  - Java-Erweiterunge deaktivieren (javaext-disable)
  - Pfad zur Barcode-Erweiterung (javaext-barcode-location)
- Diagramm-Erweiterungen
  - Diagramm-Erweiterungen deaktivieren (chartext-disable)
- .NET-Erweiterungen
  - .NET-Erweiterungen deaktivieren (dotnetext-disable)
- Allgemeine Optionen XEngines
  - XML mit PSVI laden (load-xml-with-psvi)
- XQuery
  - XQuery-Prozessorversion (xquery-version)
  - XML-Deklaration weglassen

22.3.13 JSON

Optionen für die Validierung von JSON (Instanz)-Dokumenten.

- Allgemein

© 2016-2022 Altova GmbH

Info-Limit
Ausführlich
Netzwerk-Timeout

- Katalog

  XML-Benutzerkatalog

- JSON-Validierung

  Formatüberprüfungen deaktivieren
  JSON Lines
  JSON mit Kommentaren

22.3.14 JSON Schema

Optionen für die Validierung von JSON-Schema-Dokumenten.

- Allgemein

  Info-Limit
  Ausführlich
  Netzwerk-Timeout

- Katalog

  XML-Benutzerkatalog

- JSON-Validierung

  Formatüberprüfungen deaktivieren

22.3.15 AVRO

Optionen für die Validierung eines Datenblocks in einer oder mehreren Avro-Binärdateien anhand des jeweiligen Avro-Schemas in den einzelnen Binärdateien.

- Allgemein

  Info-Limit
  Ausführlich
  Netzwerk-Timeout

- Katalog

  XML-Benutzerkatalog

22.3.16 AVRO JSON

Optionen für die Validierung eines JSON-Dokuments anhand eines AVRO-Schemas.

- Allgemein

  Info-Limit
  Ausführlich
  Netzwerk-Timeout

- Katalog

  XML-Benutzerkatalog

22.3.17 AVRO Schema

Optionen für die Validierung eines oder mehrerer Avro-Schema-Dokumente anhand der Avro-Schema-Spezifikation.

- Allgemein

  Info-Limit
  Ausführlich
  Netzwerk-Timeout

- Katalog

  XML-Benutzerkatalog

22.3.18 EDGAR

EDGAR (Electronic Data Gathering, Analysis, and Retrieval) ist ein System, das von Unternehmen an die amerikanische SEC (Securities and Exchange Commission) gesendete Finanzdaten automatisch sammelt, validiert und indiziert. Wenn Sie eine Validierung über EDGAR ausführen, validiert Raptor das XBRL-Instanzdokument mit Hilfe eines internen EDGAR-Skripts. Sie können die folgenden Zusatzoptionen definieren.

- EDGAR-Skript-Parameter

  Mit dem EDGAR-Skript wird eine zusätzliche Überprüfung entsprechend dem Dokument EDGAR Filing Manual Volume II: EDGAR Filing durchgeführt. Über das Skript können die folgenden zusätzlichen Skript-Parameter definiert werden:

CIK	Der CIK des Registranten

© 2016-2022 Altova GmbH

`submissionType`	Der EDGAR-Einreichungstyp, z.B.: `'10-K'`
`cikList`	Eine Liste von kommagetrennten `','` CIKs
`cikNameList`	Eine Liste offizieller Registrantennamen für die einzelnen CIKs in CIK. Die Namen werden durch `'|Edgar|'` getrennt.
`forceUtrValidation`	Wird auf `true` gesetzt, um die UTR-Validierung zu erzwingen
`edbody-url`	Der Pfad zur Datei `edbody.dtd`, anhand der eingebettete HTML-Fragmente validiert werden
`edgar-taxonomies-url`	Der Pfad zur Datei `edgartaxonomies.xml`, die eine Liste von Taxonomiedateien enthält, die von der Unternehmenserweiterungstaxonomie referenziert werden dürfen.

- Allgemeine

  Info-Limit
  Ausführlich
  Netzwerk-Timeout

- Katalog

  XML-Benutzerkatalog

- XML-Schema

  Import-Strategie (schema-imports)
  xsi:schemaLocation-Strategie (schemalocation-hints)
  Mapping-Strategie (schema-mapping)
  XInclude aktivieren (xinclude)
  Parallele Beurteilung
  Überprüfungsmodus für ComplexType-Einschränkungen
  Nicht übereinstimmende importierte Namespaces als Warnung ausgeben

- XBRL

  Dimensions-Erweiterung aktivieren (dimensions)
  Extensible Enumerations-Erweiterung aktivieren (extensible-enumerations)
  XBRL-Spec-Schemas im Vorhinein laden (preload-xbrl-schemas)
  Taxonomiepakete
  Taxonomiepakete Konfig-Datei
  XBRL-Inkonsistenzen als Fehler behandeln (treat-inconsistencies-as-errors)
  UTR-Datei
  Unterstützter UTR-Status
  Zusätzlicher DTS-Eintrittspunkt
  URI-Transformationsstrategie (in Ausgabedokumenten)
  Summation-Item-Inkonsistenzen melden
  Essence-Alias-Inkonsistenzen melden
  Requires-Element-Inkonsistenzen melden
  Generic Preferred Label-Erweiterung aktivieren
  Generic Links-Erweiterung aktivieren
  De-duplizieren
  Duplikate melden

Duplikate melden Schweregrad

XBRL Formula

Formula-Erweiterung aktivieren (formula)
Assertion Severity-Erweiterung (assertion-severity) aktivieren
Formula-Spec-Schemas im Vorhinein laden (preload-formula-schemas)
Nicht erfüllte Assertion-Auswertungen melden
Sprache der Validierungsmeldungen (message-lang)
Role der Validierungsmeldungen (message-role)
Formulas-to-Ignore-Datei
Formulas-to-Process-Datei
Assertions-to-Ignore-Datei
Assertions-to-Process-Datei
Formulas-to-Ignore
Assertions-to-Ignore
Formula-Ausgabe validieren
Formula-Optimierungen aktivieren

XBRL Table

Table-Erweiterung aktivieren (table)
Table-Spec-Schemas im Vorhinein laden (preload-table-schemas)
Table Linkbase Namespace
Table AspectNode-Reihenfolge

XBRL XULE

XULE
XULE-Stapelgröße
XULE-Instanz-Namespace Bindings
Zu verarbeitende XULE-Regeln
Auswertungen der XULE-Regeln ausgeben
XULE-Ausgabedatei

© 2016-2022 Altova GmbH

22.4 XSLT und XQuery mit RaptorXML Server

Sie können (i) XSLT-Transformationen und (ii) XQuery Updates oder -Ausführungen an XML-Dokumenten mit Hilfe von RaptorXML Server ausführen. Diese Aktionen stehen nur über Projekte[1080] zur Verfügung und erfolgen in den folgenden drei Schritten:

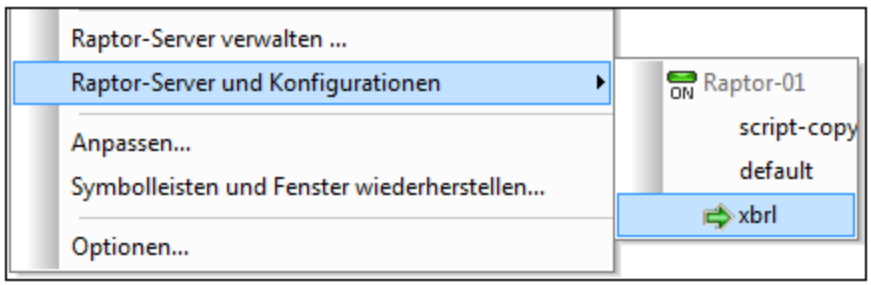
- Auswahl des Servers und der Serverkonfiguration für die Ausführung
- Einrichten des Projektordners[1081] und Definition der zu verwendenden XSLT/XQuery-Dateien (Definition erfolgt im Dialogfeld "Projekteigenschaften"[1081]). Die XSLT/XQuery-Dateien, die über das Dialogfeld "Projekteigenschaften"[1081] eines Ordners zugewiesen werden, werden für die XSLT- und XQuery-Transformation aller XML-Dateien in diesem Projektordner verwendet. Sie können XSLT/XQuery-Dateien in einem Projektordner nicht einzelnen XML-Dateien zuweisen; XSLT/XQuery-Dateien können nur einem ganzen Ordner zugewiesen werden.
- Ausführung der XSLT-Transformation oder des XQuery-Update/der XQuery-Ausführung

Anmerkung: Wenn im XSLT- oder XQuery-Dokument Java- oder .NET-Erweiterungsfunktionen verwendet werden, so werden die JAR (Java)-Dateien oder die externen (nicht registrierten) Assembly-Dateien (.NET) mit Hilfe von Dateipfaden lokalisiert. Wenn daher dasselbe XSLT/XQuery-Dokument für Transformationen/Ausführung mit XMLSpy sowie RaptorXML Server verwendet wird, so müssen die darin befindlichen Dateipfade zu JAR-Dateien und/oder Assembly-Dateien diese Dateien korrekt referenzieren.

Anmerkung: Wenn RaptorXML Server auf demselben Rechner wie XMLSpy installiert ist, sollten Sie für die Server-Einstellung **`server.unrestricted-filesystem-access`** den Wert `true` definieren, um die besten Ergebnisse zu erzielen. Nähere Informationen dazu finden Sie in der Dokumentation zur RaptorXML Server-Konfigurationsdatei.

Auswahl der gewünschten Serverkonfiguration

Wenn Sie mehrere Konfigurationen auf mehreren Servern definiert haben, können Sie einen Server und eine seiner Konfigurationen als aktive Konfiguration auswählen. Die aktive Konfiguration wird in der Folge für Validierungen verwendet. Wenn Sie den Cursor über den Befehl **Extras | Raptor-Server und Konfigurationen** (*siehe Abbildung unten*) platzieren, wird ein Untermenü mit allen hinzugefügten Servern und deren Konfiguration angezeigt. Wählen Sie die Konfiguration, die Sie zur aktiven Konfiguration machen wollen, aus. In der Abbildung unten wurde die Konfiguration `xbrl` des Servers mit dem Namen `Raptor-01` als aktive Konfiguration ausgewählt (gekennzeichnet durch einen grünen Pfeil).



Ausführen einer XSLT-Transformation

Sie können eine XSLT-Transformation mit Hilfe der XSLT-Prozessoren von XMLSpy oder mit Hilfe von RaptorXML Server ausführen. Um die XSLT-Transformation mit RaptorXML Server auszuführen, gehen Sie folgendermaßen vor:

- Klicken Sie mit der rechten Maustaste auf den Projektordner, in dem sich die zu transformierenden XML-Dateien befinden. Dabei kann es sich um den gesamten Projektordner oder einen einzelnen Ordner in der Projekthierarchie handeln.
- Wählen Sie im daraufhin angezeigten Menü den Befehl **XSL-Transformation am Server (Hochleistungstransformation)**

Anmerkung: Sie können XSLT/XQuery-Dateien in einem Projektordner nicht einzelnen XML-Dateien zuweisen; XSLT/XQuery-Dateien können nur einem ganzen Ordner zugewiesen werden. Siehe Anfang des Abschnitts 1109.

Nähere Informationen dazu finden Sie in den Abschnitten XSLT 498 und XSLT-Transformation 1411.

Ausführen eines XQuery Update/ eines XQuery-Dokuments

Sie können ein XQuery Update bzw. eine XQuery-Ausführung mit Hilfe der XQuery-Prozessoren von XMLSpy oder mit Hilfe von RaptorXML Server ausführen. Um das XQuery-Dokument bzw. das Update mit RaptorXML Server auszuführen, gehen Sie folgendermaßen vor:

- Klicken Sie mit der rechten Maustaste auf den Projektordner, in dem sich die zu aktualisierenden bzw. auszuführenden XQuery- oder XML-Dateien befinden. Dabei kann es sich um den gesamten Projektordner oder einen einzelnen Ordner in der Projekthierarchie handeln.
- Wählen Sie im daraufhin angezeigten Menü den Befehl **XQuery/Update-Ausführung am Server (Hochleistungsausführung)**

Anmerkung: Sie können XSLT/XQuery-Dateien in einem Projektordner nicht einzelnen XML-Dateien zuweisen; XSLT/XQuery-Dateien können nur einem ganzen Ordner zugewiesen werden. Siehe Anfang des Abschnitts 1109.

Nähere Informationen dazu finden Sie in den Abschnitten XQuery 514 und XQuery/Update-Ausführung 1417.

 © 2016-2022 Altova GmbH

23 Datei-/Verzeichnisvergleiche

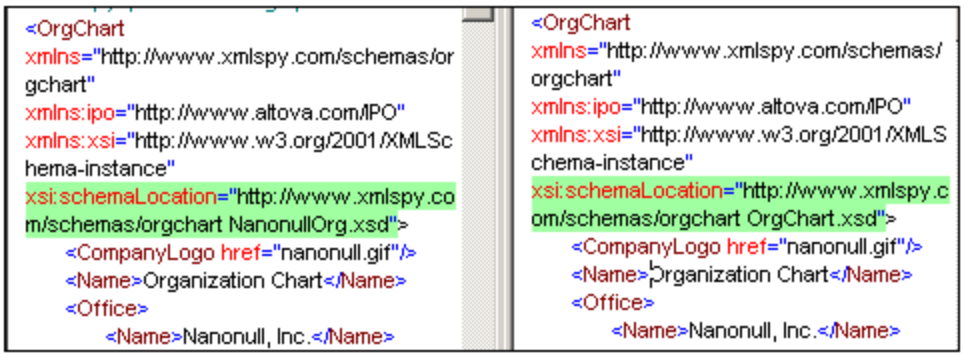
XMLSpy bietet eine Dateivergleichsfunktion sowie eine Verzeichnisvergleichsfunktion, die miteinander verknüpft sind. Datei- und Verzeichnisvergleiche werden mit dem Befehl **Geöffnete Datei vergleichen mit** bzw. **Verzeichnisse vergleichen** aus dem Menü **Extras** gestartet. Die Vergleichsoptionen für Dateivergleiche können im Dialogfeld "Einstellungen", das durch Auswahl des Befehls **Vergleichseinstellungen** aus dem Menü **Extras** aufgerufen wird, definiert werden.

Die einzelnen Befehle sind im Abschnitt Menübefehle[1567] näher beschrieben. In den Unterabschnitten dieses Abschnitts finden Sie eine Übersicht über die Funktionsweise von Dateivergleichen[1112] und Verzeichnisvergleichen[1113].

23.1 Dateivergleiche

Mit Hilfe der Dateivergleichsfunktion[1567] können Sie die aktive Datei mit einer anderen Datei, welche über ein "Datei öffnen" Dialogfeld oder eine globale Ressource[1061] ausgewählt wird, vergleichen. Im Folgenden finden Sie eine Anleitung zur Vorgehensweise. Nähere Informationen dazu finden Sie im Abschnitt Menübefehle.[1567]

- Die aktuellen Einstellungen im Dialogfeld Vergleichsoptionen[1575] sind die Einstellungen, die bei Start eines Dateivergleichs verwendet werden.
- Sie können auswählen, ob die Dateien als XML-Dateien (wobei auch die Dokumentstruktur berücksichtigt wird) oder als Textdateien verglichen werden sollen. Wählen Sie dazu im Dialogfeld "Einstellungen"[1575] entweder (i) für XML-Vergleiche die Grid- oder die Textansicht (wobei "Nur Textvergleich" deaktiviert sein muss) oder (ii) für Textvergleiche die Textansicht (wobei "Nur Textvergleich" aktiviert sein muss).
- Die beiden Dateien werden in der ausgewählten Ansicht (Text- oder Grid-Ansicht) in zwei Fenstern nebeneinander angezeigt, wobei die Unterschiede in beiden Dateien markiert erscheinen (*Abbildung unten*).



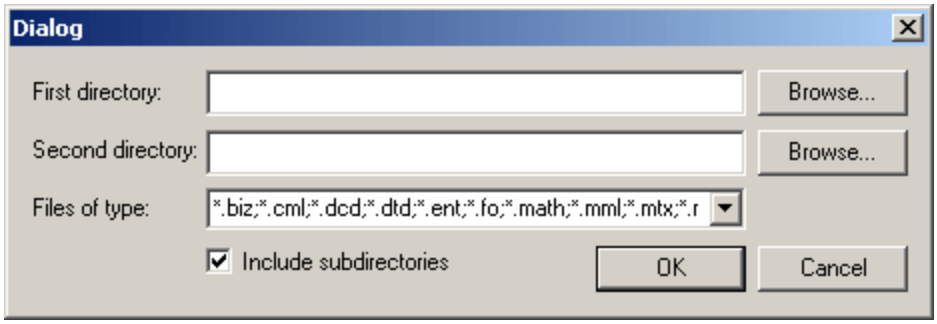
Außerdem wird ein Fenster mit Steuerelementen angezeigt, über das Sie durch die Unterschiede navigieren und diese zusammenführen können.

Das Dialogfeld "Einstellungen"[1575] enthält eine Reihe von Optionen, um zu definieren, welche Aspekte der XML-Dokumente beim Vergleich berücksichtigt und welche ignoriert werden sollen. Nähere Informationen dazu finden Sie in den Menübefehlen[1567] im Abschnitt Vergleichsoptionen[1575].

© 2016-2022 Altova GmbH

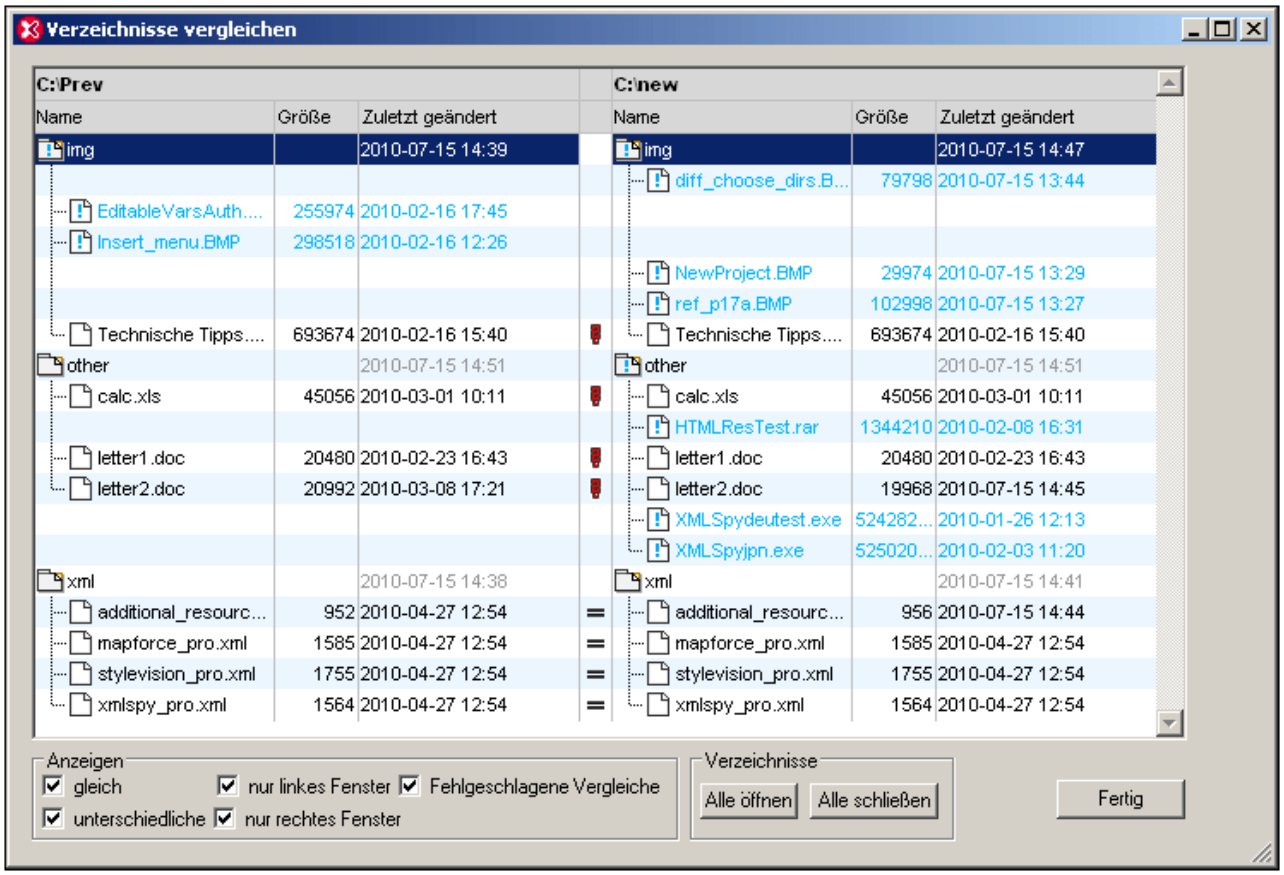
23.2 Verzeichnisvergleiche

Mit Hilfe der Verzeichnisvergleichsfunktion[1572] können Sie Verzeichnisse miteinander vergleichen. Jedes der Verzeichnisse wird über ein separates Dialogfeld zum Suchen nach Verzeichnissen ausgewählt. Außerdem können Sie festlegen, ob auch Unterverzeichnisse verglichen werden sollen und welche Dateiarten beim Verzeichnisvergleich berücksichtigt werden sollen.



Auch ZIP-Archive können in den Vergleich inkludiert werden, wenn Sie die Erweiterung für ZIP-Dateien in die Liste der zu vergleichenden Dateitypen inkludieren.

Beim Verzeichnisvergleich wird angezeigt, welche Dateien fehlen und ob Dateien desselben Namens sich unterscheiden oder nicht. Der Vergleich zwischen Dateien erfolgt auf Basis der Einstellungen im Dialogfeld "Einstellungen"[1575]. Die Ergebnisse des Verzeichnisvergleichs werden in einem separaten Fenster angezeigt (*Abbildung unten*).



Nähere Informationen zur Bedeutung der einzelnen Symbole und zum Verwalten der Ansicht im Fenster "Verzeichnisse vergleichen" finden Sie in der Menübefehle[1567] unter der Beschreibung des Befehls **Verzeichnisse vergleichen.** Sie können anschließend auf eine Dateizeile doppelklicken, um einen Dateivergleich sofort zu starten.

© 2016-2022 Altova GmbH

24 Versionskontrolle

Die Versionskontrollunterstützung in XMLSpy steht über die Microsoft Source Control Plug-in API (vormals bekannt als MSSCCI API), Version 1.1, 1.2 sowie 1.3 zur Verfügung. Dadurch können Sie Versionskontrollbefehle wie z.B. "Einchecken" oder "Auschecken" direkt über XMLSpy an praktisch jedem Versionskontrollsystem ausführen, das nativen Clients oder Drittanbieter-Clients über das Microsoft Source Control Plug-in API die Verbindung zu XMLSpy gestattet.

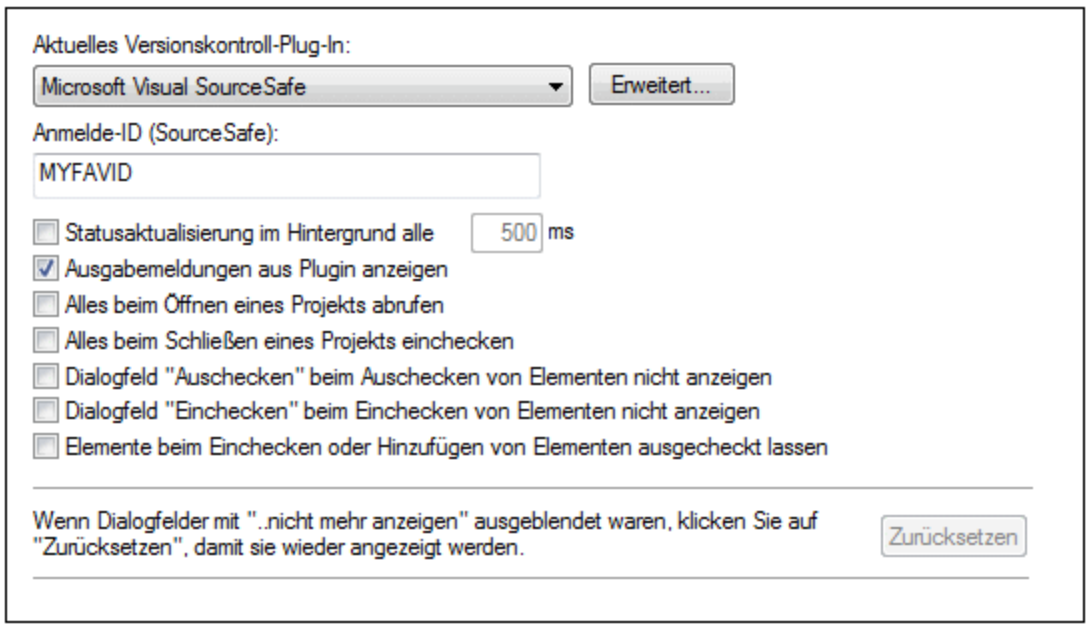
Als Versionskontrollsystemanbieter kann jedes kommerzielle oder nicht kommerzielle Plug-in, das die Microsoft Source Control Plug-in API unterstützt und mit einem kompatiblen Versionskontrollsystem verbunden werden kann, verwendet werden. Eine Liste von von Altova getesteten Versionskontrollsystemen und Plug-ins finden Sie unter Unterstützte Versionskontrollsysteme 1118.

Installieren und Konfigurieren des Versionskontrollanbieters

Um die auf Ihrem System verfügbaren Versionskontrollsysteme anzuzeigen, gehen Sie folgendermaßen vor:

1. Klicken Sie im Menü **Extras** auf **Optionen**.
2. Klicken Sie auf das Register **Versionskontrolle**.

Alle mit der Microsoft Source Code Control Plug-in API kompatiblen Versionskontroll-Plug-ins werden in der Dropdown-Liste **Aktuelles Versionskontroll-Plug-in** angezeigt.



Wenn auf Ihrem System kein kompatibles Plug-in gefunden wurde, wird die folgende Meldung angezeigt:

```
"Die Registrierung des installierten Versionskontrollproviders konnte nicht gefunden werden
oder ist unvollständig."
```

Bei einigen Versionskontrollsystemen wird das Versionskontroll-Plug-in nicht automatisch installiert. In diesem Fall müssen Sie es gesondert installieren. Eine genauere Anleitung dazu finden Sie in der Dokumentation zum

jeweiligen Versionskontrollsystem. Ein Plug-in (Anbieter), das mit der Microsoft Source Code Control Plug-in API kompatibel ist, sollte auf Ihrem Betriebssystem unter dem folgenden Registrierdateieintrag zu finden sein:

```
HKEY_LOCAL_MACHINE\SOFTWARE\SourceCodeControlProvider\InstalledSCCProviders
```

Wenn das Plug-in korrekt installiert wurde, steht es automatisch in der Liste der Plug-ins in XMLSpy zur Verfügung.

Aufrufen der Versionskontrollbefehle

Die Versionskontrollbefehle stehen über das Menü **Projekt | Versionskontrolle** zur Verfügung.

Probleme im Zusammenhang mit den Ressourcen / der Geschwindigkeit

Bei sehr großen Versionskontrolldatenbanken kann die automatische Durchführung von Statusaktualisierungen im Hintergrund etwas länger dauern.

Um schneller arbeiten zu können, können Sie versuchen, auf dem Register "Versionskontrolle" (Aufruf über **Extras | Optionen**) das Feld "**Statusaktualisierung im Hintergrund alle xxx Sek**." zu deaktivieren oder das Aktualisierungsintervall zu vergrößern.

Anmerkung: Die **64-Bit**-Version Ihrer Altova-Applikation unterstützt automatisch alle in dieser Dokumentation aufgelisteten 32-Bit-Versionskontrollsysteme. Bei Verwendung einer 64-Bit-Altova-Applikation zusammen mit einem 32-Bit-Versionskontrollsystem ist die Option **Statusaktualisierung im Hintergrund alle xxx Sek** automatisch deaktiviert und kann nicht ausgewählt werden.

Vergleich mit Altova DiffDog

Sie können viele Versionskontrollsysteme (einschließlich Git und TortoiseSVN) für die Verwendung mit Altova DiffDog als Vergleichstool konfigurieren. Nähere Informationen zu DiffDog finden Sie unter https://www.altova.com/de/diffdog.html. Die Dokumentation zu DiffDog finden Sie unter https://www.altova.com/de/documentation.html.

© 2016-2022 Altova GmbH

24.1 Einrichten der Versionskontrolle

So richten Sie eine Versionskontrolle ein und stellen Dateien in einem XMLSpy-Projekt unter eine Versionskontrolle:

1. Installieren Sie ein Versionskontrollprogramm (siehe Unterstützte Versionskontrollsysteme[1118]), falls noch keines installiert ist. Richten Sie die Versionskontrolldatenbank (das Repository), in der Sie Ihre Arbeit speichern möchten, ein.
2. Erstellen Sie einen Ordner für den lokalen Arbeitsbereich, der die Arbeitsdateien enthalten soll, die unter Versionskontrolle gestellt werden sollen. Der Ordner, der alle Ihre Arbeitsbereichordner und -Dateien enthält, wird als ihr lokaler Ordner und der Pfad zum lokalen Ordner als der lokale Pfad bezeichnet. Der Ordner wird an einen bestimmten Ordner im Repository gebunden.
3. Erstellen Sie in Ihrer Altova-Applikation einen Applikationsprojektordner, zu dem Sie die unter Versionskontrolle zu stellenden Dateien hinzufügen müssen. Diese Gliederung der Dateien in einem Applikationsprojekt ist abstrakt. Die Dateien in einem Projekt referenzieren physische lokal (vorzugsweise in einem einzigen Ordner, falls nötig mit Unterordnern) gespeicherte Dateien.
4. In der Datenbank des Versionskontrollsystems (die auch als Versionskontrolle oder Repository bezeichnet wird) wird ein Ordner erstellt, der an den lokalen Ordner gebunden ist. Dieser (als gebundener Ordner bezeichnete) Ordner repliziert die Struktur des lokalen Ordners, sodass sich alle unter Versionskontrolle zu stellenden Dateien an der richtigen Stelle im gebundenen Ordner befinden. Der gebundene Ordner wird normalerweise erstellt, wenn Sie zum ersten Mal eine Datei oder ein Applikationsprojekt zur Versionskontrolle hinzufügen. Nähere Informationen zur Ordnersturktur des Repository finden Sie im Abschnitt Applikationsprojekt[1121].
5. Projektdateien werden mit dem Befehl**Projekt | Versionskontrolle | Zu Versionskontrolle hinzufügen** zur Versionskontrolle hinzugefügt. Wenn Sie zum ersten Mal eine Datei oder ein Applikationsprojekt zur Versionskontrolle hinzufügen, werden die korrekten Bindings erstellt und die Ordnerstruktur wird im Repository erstellt.
6. Versionskontrollaktionen wie z.B. das Ein- und Auschecken von Dateien und das Entfernen von Dateien aus der Versionskontrolle können über Befehle im Untermenü **Projekt | Versionskontrolle** ausgeführt werden. Diese Befehle werden in der Benutzerreferenz in den Unterabschnitten des Menüs "Projekt"[1320] beschrieben.

Anmerkung: Sie können den aktuellen Versionskontrollanbieter auf zwei Arten wechseln: (i) über die Versionskontrolloptionen (**Extras | Optionen | Versionskontrolle**[1645]), oder (ii) im Dialogfeld "Versionskontrolle wechseln" (**Projekt | Versionskontrolle | Versionskontrolle wechseln**).

24.2 Unterstützte Versionskontrollsysteme

In der nachfolgenden Liste sind die von XMLSpy unterstützten Versionskontrollsysteme zusammen mit ihren entsprechenden Versionskontroll-Clients (SCCs) aufgelistet. Die Liste ist alphabetisch nach Versionskontrollsystem geordnet.

- Altova hat in XMLSpy die Microsoft Source Control Plug-in API (Version 1.1, 1.2 und 1.3) implementiert und die Unterstützung für die aufgelisteten Treiber und Versionskontrollsysteme getestet. Aller Voraussicht nach wird XMLSpy diese Produkte, falls diese aktualisiert werden, weiterhin unterstützen.
- Auch Versionskontroll-Clients, die nicht in der nachstehenden Liste erwähnt sind, die aber die Microsoft Source Control Plug-in API implementieren, sollten ebenfalls mit XMLSpy funktionieren.

Versionskontrollsystem	Versionskontroll-Clients
AccuRev 4.7.0 Windows	AccuBridge for Microsoft SCC 2008.2
Bazaar 1.9 Windows	Aigenta Unified SCC 1.0.6
Borland StarTeam 2008	Borland StarTeam Cross-Platform Client 2008 R2
Codice Software Plastic SCM Professional 2.7.127.10 (Server)	Codice Software Plastic SCM Professional 2.7.127.10 (SCC Plugin)
Collabnet Subversion 1.5.4	• Aigenta Unified SCC 1.0.6 • PushOK SVN SCC 1.5.1.1 • PushOK SVN SCC x64 Version 1.6.3.1 • TamTam SVN SCC 1.2.24
ComponentSoftware CS-RCS (PRO) 5.1	ComponentSoftware CS-RCS (PRO) 5.1
Dynamsoft SourceAnywhere for VSS 5.3.2 Standard/Professional Server	Dynamsoft SourceAnywhere für VSS 5.3.2 Client
Dynamsoft SourceAnywhere Hosted	Dynamsoft SourceAnywhere Hosted Client (22252)
Dynamsoft SourceAnywhere Standalone 2.2 Server	Dynamsoft SourceAnywhere Standalone 2.2 Client
Git	PushOK GIT SCC Plug-in (siehe Versionskontrolle mit Git 1136)
IBM Rational ClearCase 7.0.1 (LT)	IBM Rational ClearCase 7.0.1 (LT)
March-Hare CVSNT 2.5 (2.5.03.2382)	Aigenta Unified SCC 1.0.6
March-Hare CVS Suite 2008	• Jalindi Igloo 1.0.3 • March-Hare CVS Suite Client 2008 (3321) • PushOK CVS SCC NT 2.1.2.5 • PushOK CVS SCC x64 Version 2.2.0.4 • TamTam CVS SCC 1.2.40
Mercurial 1.0.2 für Windows	Sergey Antonov HgSCC 1.0.1

© 2016-2022 Altova GmbH

Versionskontrollsystem	Versionskontroll-Clients
Microsoft SourceSafe 2005 mit CTP	Microsoft SourceSafe 2005 mit CTP
Microsoft Visual Studio Team System 2008/2010 Team Foundation Server	Microsoft Team Foundation Server 2008/2010 MSSCCI Provider
Perforce 2008 P4S 2008.1	Perforce P4V 2008.1
PureCM Server 2008/3a	PureCM Client 2008/3a
QSC Team Coherence Server 7.2.1.35	QSC Team Coherence Client 7.2.1.35
Reliable Software Code Co-Op 5.1a	Reliable Software Code Co-Op 5.1a
Seapine Surround SCM Client/Server für Windows 2009.0.0	Seapine Surround SCM Client 2009.0.0
Serena Dimensions Express/CM 10.1.3 für Win32 Server	Serena Dimensions 10.1.3 for Win32 Client
Softimage Alienbrain Server 8.1.0.7300	Softimage Alienbrain Essentials/Advanced Client 8.1.0.7300
SourceGear Fortress 1.1.4 Server	SourceGear Fortress 1.1.4 Client
SourceGear SourceOffsite Server 4.2.0	SourceGear SourceOffsite Client 4.2.0 (Windows)
SourceGear Vault 4.1.4 Server	SourceGear Vault 4.1.4 Client
VisualSVN Server 1.6	• Aigenta Unified SCC 1.0.6 • PushOK SVN SCC 1.5.1.1 • PushOK SVN SCC x64 Version 1.6.3.1 • TamTam SVN SCC 1.2.24

24.3 Lokaler Arbeitsbereichordner

Die Dateien, mit denen Sie arbeiten werden, sollten innerhalb eines lokalen Arbeitsbereichordners in Form einer Hierarchie gespeichert werden (*siehe Diagramm unten*).

```
Lokaler Arbeitsbereichordner
|
|-- MyProject.spp
|-- QuickStart
|   |-- QuickStart.css
|   |-- QuickStart.xml
|   |-- QuickStart.xsd
|-- Grouping
|   | -- Persons
|   |    | -- Persons.xml
```

Die Applikationsprojektdatei (`.spp` Datei) befindet sich normalerweise direkt innerhalb des loalen Arbeitsbereichordners (*siehe obiges Diagramm*).

Wenn eine oder mehrere Dateien dieses (Arbeitsbereichs)ordners unter Versionskontrolle gestellt werden, wird die Struktur des lokalen Arbeitzsbereichsordners zum Teil oder zur Gänze im Repository nachvollzogen. Wenn z.B. die Datei `Persons.xml` aus dem oben gezeigten lokalen Ordner unter Versionskontrolle gestellt wird, so lautet der Pfad zu dieser Datei im Repository:

```
[RepositoryFolder]/MyProject/Grouping/Persons/Persons.xml
```

Der Ordner `MyProject` im Repository-Ordner ist an den lokalen Ordner gebunden. Normalerweise erhält er den Namens des Projekts, kann aber auch jeden anderen Namen erhalten.

Wenn das gesamte Applikationsprojekt (durch Auswahl des Projektnamens im Fenster "Projekt" und setzen des Projekts unter Versionskontrolle) unter Versionskontrolle gestellt wird, so wird die gesamte lokale Ordnerstruktur auch im Repository angelegt.

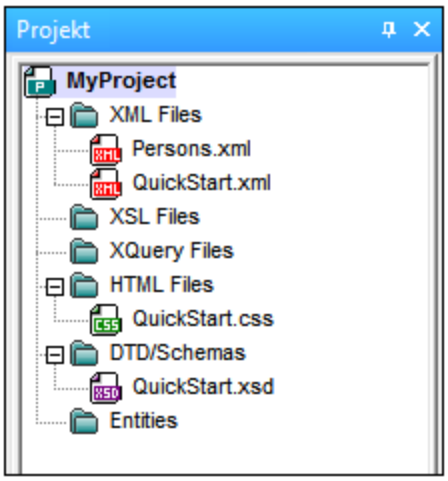
Anmerkung: Es können auch Dateien, die außerhalb des lokalen Arbeitsbereichsordners gespeichert sind, zum Applikationsprojekt hinzugefügt werden. Ob Sie eine solche Datei unter Versionskontrolle stellen können, hängt vom verwendeten Versionskontrollsystem ab und lässt sich nicht bei allen Versionskontrollsystemen bewerkstelligen. Es wird daher empfohlen, dass sich alle Projektdateien, die unter Versionskontrolle gestellt werden sollen, im lokalen Arbeitsbereichsordner befinden sollten.

© 2016-2022 Altova GmbH

24.4 Applikationsprojekt

Erstellen oder laden Sie das Altova Applikationsprojekt, das Sie unter Versionskontrolle stellen möchten. Wenn Sie eine einzige Datei unter Versionskontrolle stellen möchten, sollten Sie sie in ein Projekt inkludieren, da die Versionskontrolle nur über ein Projekt aufgerufen werden kann.

Betrachten wir z.B. ein Projekt in der Altova Applikation XMLSpy. Die Eigenschaften des Projekts sind in einer `.spp` Datei gespeichert. Das Projekt wird in der Applikation im Fenster "Projekt" (*siehe Abbildung unten*) angezeigt. Das Projekt in der Abbildung unten hat den Namen `MyProject` und die Eigenschaften des Projekts sind in der Datei `MyProject.spp` gespeichert.



Sie können das gesamte Projekt (alle Dateien im Projekt) oder nur einige Projektdateien unter Versionskontrolle stellen. **Es können nur Dateien, die sich im Projekt befinden, unter Versionskontrolle gestellt werden.** Sie müssen daher Dateien zum Projekt hinzufügen, bevor Sie sie unter Versionskontrolle stellen können. Die Projektdatei (`.spp` Datei) wird automatisch unter Versionskontrolle gestellt, sobald eine Datei aus dem Projekt unter Versionskontrolle gestellt wird.

Mit dem Befehl **Projekt | Versionskontrolle | Zu Versionskontrolle hinzufügen** (*siehe nächster Abschnitt weiter unten*) wird das gesamte Projekt oder eine oder mehrere Projektdateien unter Versionskontrolle gestellt.

Beachten Sie jedoch, dass die Ordnerstruktur des Repository nicht der Ordnerstruktur des Projekts (*Abbildung oben*), sondern der Struktur des lokalen Arbeitsbereichordners 1120 (*siehe Ordnerdiagramm unten*) entspricht. Beachten Sie, dass der Ordner `MyProject` im Diagramm unten im Repository eine Ordnerstruktur hat, die der des lokalen Arbeitsbereichordners entspricht. Beachten Sie außerdem, dass sich der gebundene Ordner innerhalb des Repository-Ordners befindet.

```
Lokaler Arbeitsbereichsordner          Repository
|                                      |
|-- MyProject.spp                      |-- MyProject (gebunden an lokalen
                                       Arbeitsbereich)
|-- QuickStart                         ||-- MyProject.spp
|   |-- QuickStart.css                 ||-- QuickStart
|   |-- QuickStart.xml                 ||   |-- QuickStart.css
```

```
|   |-- QuickStart.xsd
|-- Grouping
|   |-- Persons
|   |   |-- Persons.xml
```

```
||   |-- QuickStart.xml
||   |-- QuickStart.xsd
||-- Grouping
||   |-- Persons
||   |   |-- Persons.xml
```

Anmerkung: Ein Applikationsprojekt kann Projektordner (grün) und externe Ordner (gelb) enthalten. Nur Dateien in (grünen) Projektordnern können unter Versionskontrolle gestellt werden. Dateien in (gelben) externen Ordnern können nicht unter Versionskontrolle gestellt werden.

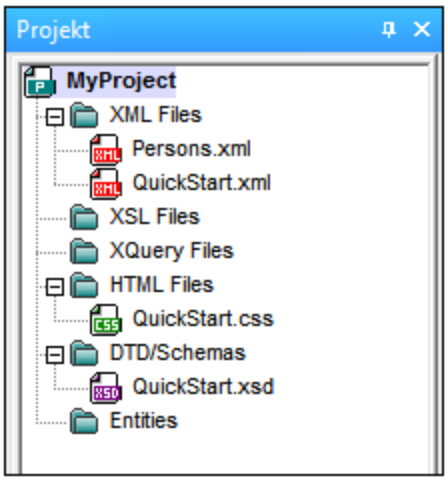
Anmerkung: Es können auch Dateien, die außerhalb des lokalen Arbeitsbereichsordners gespeichert sind, zum Applikationsprojekt hinzugefügt werden. Ob Sie eine solche Datei unter Versionskontrolle stellen können, hängt vom verwendeten Versionskontrollsystem ab und lässt sich nicht bei allen Versionskontrollsystemen bewerkstelligen. Es wird daher empfohlen, dass sich alle Projektdateien, die unter Versionskontrolle gestellt werden sollen, im lokalen Arbeitsbereichsordner befinden sollten.

© 2016-2022 Altova GmbH

24.5 Zu Versionskontrolle hinzufügen

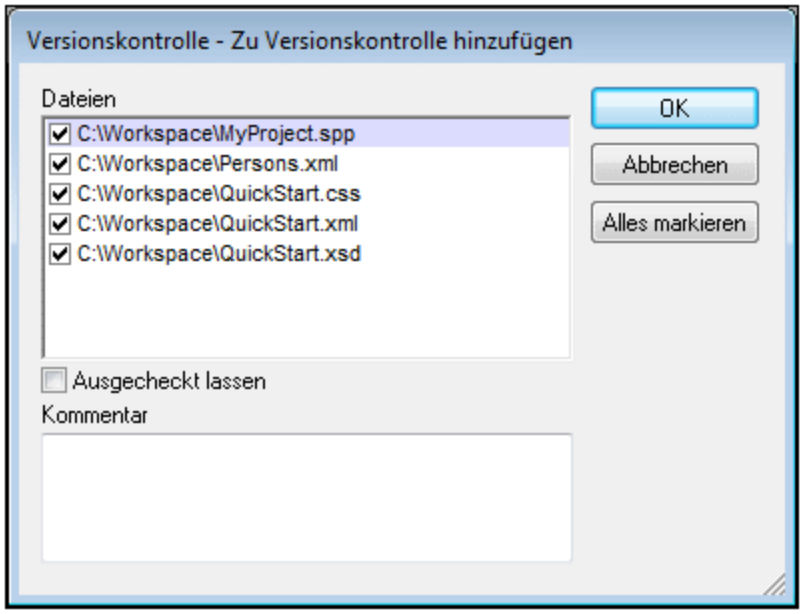
Wenn Sie das Projekt zur Versionskontrolle hinzufügen, werden automatisch die richtigen Bindings und die korrekte Repository-Struktur erstellt, bevor die Projektdatei (`.spp` Datei) oder einzelne Dateien zur Versionskontrolle hinzugefügt werden. Um ein Projekt zur Versionskontrolle hinzuzufügen, gehen Sie folgendermaßen vor.

Wählen Sie das Projekt im Fenster "Projekt" aus (in der Abbildung unten `MyProject`), so dass es markiert erscheint (*wie in der Abbildung unten*). Wählen Sie alternativ dazu eine einzige Datei aus oder halten Sie beim Anklicken der Dateien die **Strg**-Taste gedrückt, um mehrere auszuwählen. Wenn Sie eine einzige Datei zur Versionskontrolle hinzufügen, wird auch automatisch die Projektdatei (`.spp` Datei) zur Versionskontrolle hinzugefügt.



Wählen Sie als nächstes den Menübefehl **Projekt | Versionskontrolle | Zu Versionskontrolle hinzufügen**. Daraufhin werden die Dialogfelder zur Herstellung und Konfigurierung der Verbindung zum aktuell ausgewählten Versionskontrollsystem angezeigt. (Sie können das Versionskontrollsystem über das Dialogfeld "Versionskontrolle wechseln" (**Projekt | Versionskontrolle | Versionskontrolle wechseln**) wechseln).

Befolgen Sie die Anweisungen des Versionskontrollsystems um die Verbindung herzustellen und das System zu konfigurieren. Anschließend werden alle zum Hinzufügen ausgewählten Dateien plus die Projektdatei (`.spp` Datei) im Dialogfeld "Zu Versionskontrolle hinzufügen" (*Abbildung unten*) angezeigt. Wählen Sie die gewünschten Dateien aus und klicken Sie auf **OK**.



Die Dateien werden zum Repository hinzugefügt und je nachdem, ob das Kontrollkästchen *Ausgecheckt lassen* aktiviert wurde oder nicht, entweder ein- oder ausgecheckt[1127].

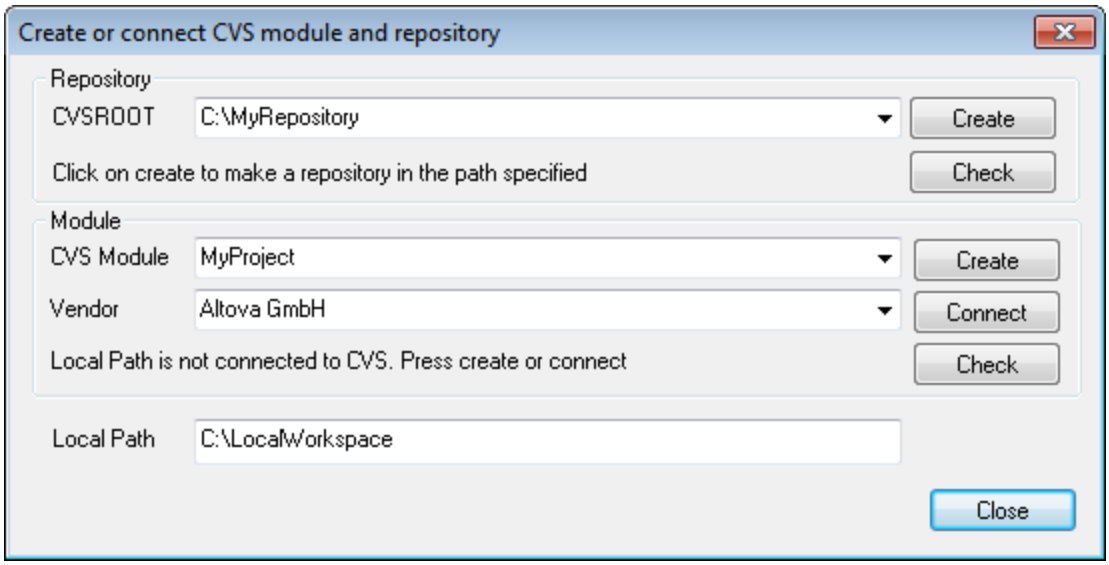
Anmerkungen zur Konfiguration

Eventuell werden Sie aufgefordert, im Repository einen Ordner für das Projekt zu erstellen, falls dies noch nicht geschehen ist. Nachdem Sie diesen erstellt haben, wird der lokale Arbeitsbereichordner[1120] an diesen im Repository erstellten Ordner gebunden (*siehe Diagramme unten*).

```
Lokaler Arbeitsbereichsordner        Repository
|                                    |
|-- MyProject.spp                    |-- MyProject (gebunden an lokalen
                                     Arbeitsbereich)
|-- QuickStart                       ||-- MyProject.spp
|   |-- QuickStart.css               ||-- QuickStart
|   |-- QuickStart.xml               ||   |-- QuickStart.css
|   |-- QuickStart.xsd               ||   |-- QuickStart.xml
|-- Grouping                         ||   |-- QuickStart.xsd
|   |-- Persons                      ||-- Grouping
|   |   |-- Persons.xml              ||   |-- Persons
                                     ||   |   |-- Persons.xml
```

Unten sehen Sie das Konfigurationsdialogfeld für Jalindi Igloo. Das Feld "CVSROOT" ist der Pfad zum Repository-Ordner.

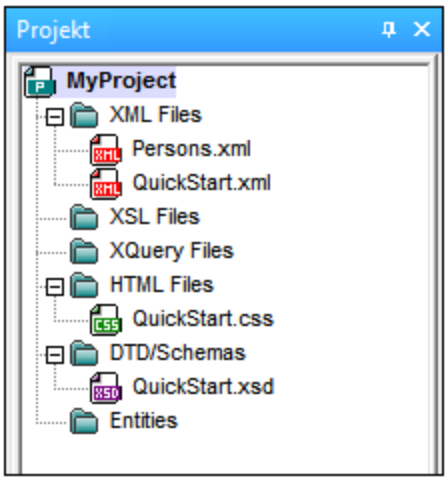
© 2016-2022 Altova GmbH



In der Abbildung oben entspricht der lokale Pfad dem lokalen Arbeitsbereichordner, der dem CVS-Modul `MyProject` entspricht und an diesen gebunden ist.

24.6 Arbeiten mit Versionskontrolle

Um mit der Versionskontrolle zu arbeiten, wählen Sie das Projekt, einen Projektordner oder eine Projektdatei im Fenster "Projekt" aus (*Abbildung unten*) und wählen Sie anschließend im Menü **Projekt | Versionskontrolle** den gewünschten Befehl aus. Die Befehle **Einchecken** und **Auschecken** stehen im Fenster "Projekt" als Kontextmenübefehle zur Verfügung.



In diesem Abschnitt beschreiben wir die wichtigsten Versionskontrollfunktionen im Detail:

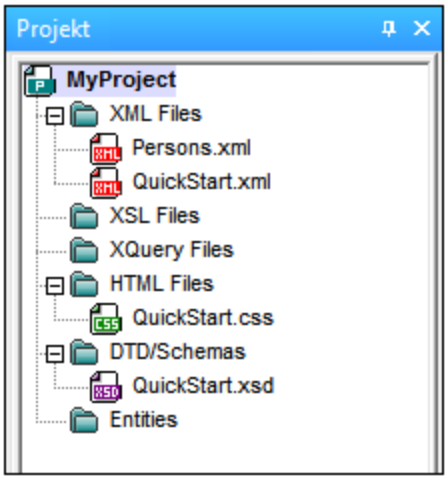
- Hinzufügen zur und Entfernen aus der Versionskontrolle 1126
- Einchecken, Auschecken 1127
- Dateien schreibgeschützt abrufen 1130
- Kopieren und Freigeben über die Versionskontrolle 1131
- Wechseln der Versionskontrolle 1134

Weitere Befehle im Menü **Projekt | Versionskontrolle** werden im Abschnitt Menübefehle 1320 dieses Handbuchs beschrieben. Informationen zu einem spezifischen Versionskontrollsystem finden Sie in der Dokumentation zu diesem System.

24.6.1 Hinzufügen zur, Entfernen aus der Versionskontrolle

Hinzufügen
Nachdem ein Projekt zu einer Versionskontrolle hinzugefügt wurde, können Sie Dateien entweder einzeln oder in Gruppen unter Versionskontrolle stellen. Dies wird auch als Hinzufügen der Dateien zur Versionskontrolle bezeichnet. Wählen Sie die Datei im Fenster "Projekt" aus und klicken Sie auf den Befehl **Projekt | Versionskontrolle | Zu Versionskontrolle hinzufügen**. Um mehrere Dateien auszuwählen, halten Sie beim Anklicken der Dateien die **Strg**-Taste gedrückt. Wenn Sie den Befehl an einem (grünen) Projektordner (*siehe Abbildung unten*) ausführen, werden alle Dateien im Ordner und seinen Unterordnern zur Versionskontrolle hinzugefügt.

© 2016-2022 Altova GmbH



Wenn Dateien zur Versionskontrolle hinzugefügt werden, wird die lokale Ordnerhierarchie im Repository repliziert[1121] (jedoch nicht die Projektordnerhierarchie). Wenn eine Datei sich also im lokalen Ordner in einem Unterordner mehrere Ebenen unterhalb des lokalen Ordners befindet, so werden im Repository automatisch auch der der Datei übergeordnete Ordner und alle darüber liegenden erstellt.

Wenn die erste Datei aus einem Projekt zur Versionskontrolle hinzugefügt wird, werden im Repository die korrekten Bindings erstellt und die Projektdatei (`.spp` Datei) wird automatisch hinzugefügt. Nähere Informationen dazu finden Sie im Abschnitt Zur Versionskontrolle hinzufügen[1123].

Versionskontrollsymbole

Zu Dateien und dem Projektordner können bestimmte Symbole angezeigt werden, deren Bedeutung unten erklärt ist.

	Eingecheckt. Für das Auschecken verfügbar.
	Von einem anderen Benutzer ausgecheckt. Kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Kann bearbeitet und eingecheckt werden.

Entfernen

Um eine Datei aus der Versionskontrolle zu entfernen, wählen Sie sie aus und klicken Sie auf den Befehl **Projekt | Versionskontrolle | Von Versionskontrolle ausgliedern**. Außerdem können Sie: (i) Dateien in einem Projektordner durch Ausführen des Befehls am Ordner entfernen und (ii) das gesamte Projekt durch Ausführen des Befehls am Projekt entfernen.

24.6.2 Auschecken, Einchecken

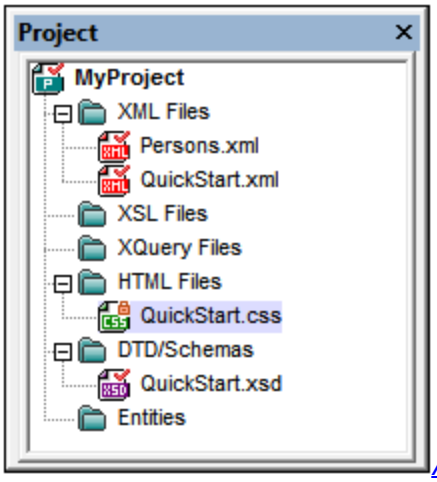
Nachdem Sie eine Projektdatei unter Versionskontrolle gestellt haben, kann sie durch Auswahl der Datei (im Projektfenster) und Klicken auf den entsprechenden Befehl im Menü **Projekt | Versionskontrolle** ein- und ausgecheckt werden.

Wenn eine Datei ausgecheckt wird, wird eine Kopie aus dem Repository in den lokalen Ordner platziert. Eine ausgecheckte Datei kann bearbeitet werden. Wenn eine Datei, die sich unter Versionskontrolle befindet, nicht ausgecheckt ist, kann sie nicht bearbeitet werden. Nachdem eine Datei bearbeitet wurde, können die Änderungen durch Einchecken der Datei im Repository gespeichert werden. Selbst wenn die Datei nicht in der Applikation gespeichert wurde, werden die Änderungen an der Datei durch Einchecken im Repository gespeichert. Der Ein- und Auscheckstatus einer Datei wird durch ein Häkchen bzw. ein Schloßsymbol im Symbol der jeweiligen Datei im Projektfenster angezeigt.

Zu Dateien und dem Projektordner können bestimmte Symbole angezeigt werden, deren Bedeutung unten erklärt ist.

	Eingecheckt. Für das Auschecken verfügbar.
	Von einem anderen Benutzer ausgecheckt. Kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Kann bearbeitet und eingecheckt werden.

Durch Auswahl des Projekts oder eines Ordners im Projekt werden alle Dateien im ausgewählten Objekt ausgewählt. Um mehrere Objekte (Dateien und Ordner) auszuwählen, halten Sie beim Anklicken der Objekte die Strg-Taste gedrückt. In der Abbildung unten sehen Sie ein ausgechecktes Projekt. Die Datei `QuickStart.css` wurde im Anschluss daran eingecheckt.



Abrufen von Dateien als schreibgeschützte Dateien 1130

Speichern und Ablehnen von Bearbeitungen

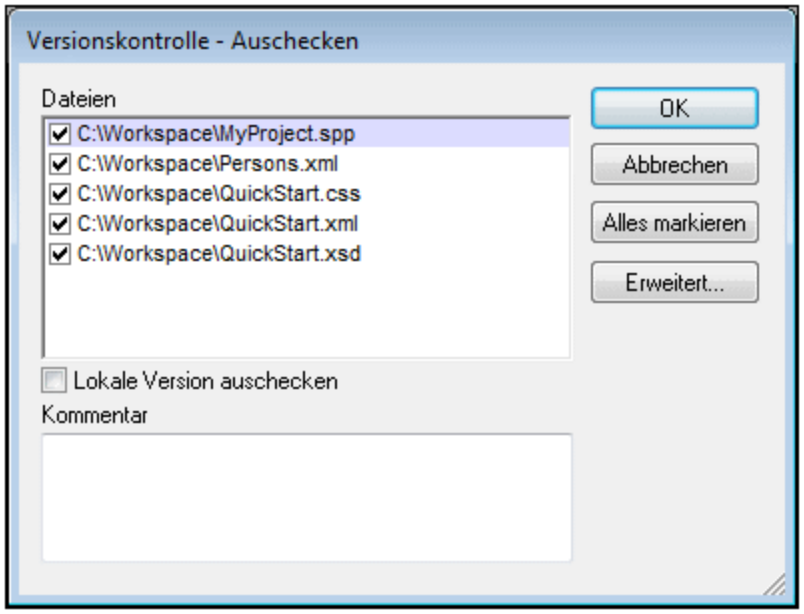
Beachten Sie, dass Sie eine Datei beim Einchecken ausgecheckt lassen können. Dabei werden die Änderungen im Repository gespeichert, während die Datei weiterhin ausgecheckt bleibt. Diese Option ist nützlich, wenn Sie Ihre Änderungen regelmäßig im Repository speichern und mit der Bearbeitung fortfahren möchten.

Wenn Sie eine Datei ausgecheckt haben, diese bearbeitet haben und die Änderungen anschließend aber nicht übernehmen möchten, können Sie die im Repository gespeicherte Dokumentversion in Ihrem lokalen Ordner wiederherstellen, indem Sie den Befehl **Projekt | Versionskontrolle | Auschecken rückgängig machen** auswählen.

© 2016-2022 Altova GmbH

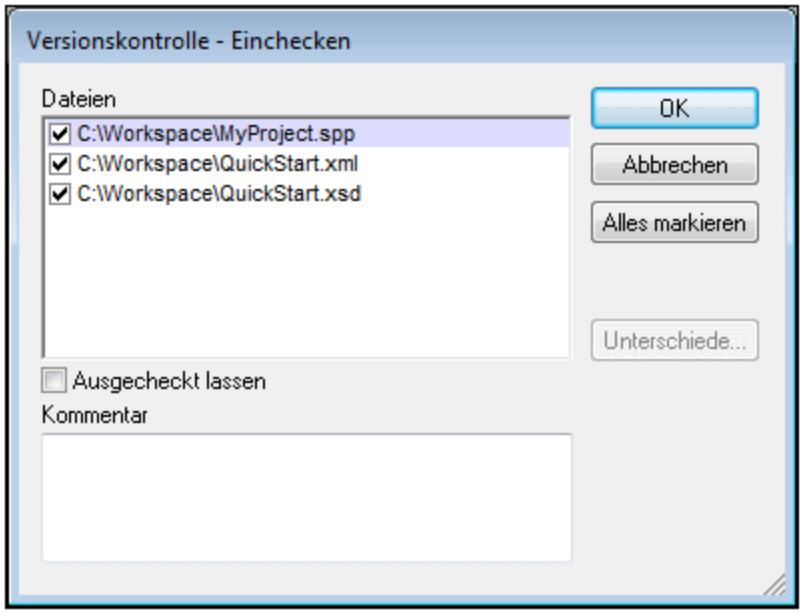
Auschecken

Im Dialogfeld "Auschecken" (*Abbildung unten*) können Sie (i) die auszucheckenden Dateien auswählen und (ii) wählen, ob die Repository-Version oder die lokale Version ausgecheckt werden soll.



Einchecken

Im Dialogfeld "Einchecken" (*Abbildung unten*) können Sie (i) die einzucheckenden Dateien auswählen und (ii) wählen, ob die Datei ausgecheckt bleiben soll.

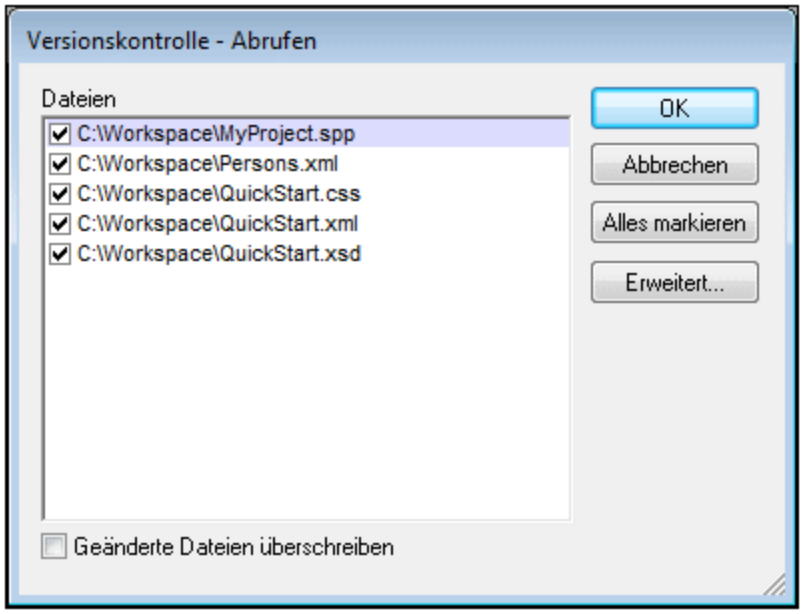


Anmerkung: In beiden Dialogfeldern (Einchecken und Auschecken) werden mehrere Dateien angezeigt, wenn das ausgewählte Objekt (Projekt oder Projektordner) mehrere Dateien enthält.

24.6.3 Dateien schreibgeschützt abrufen

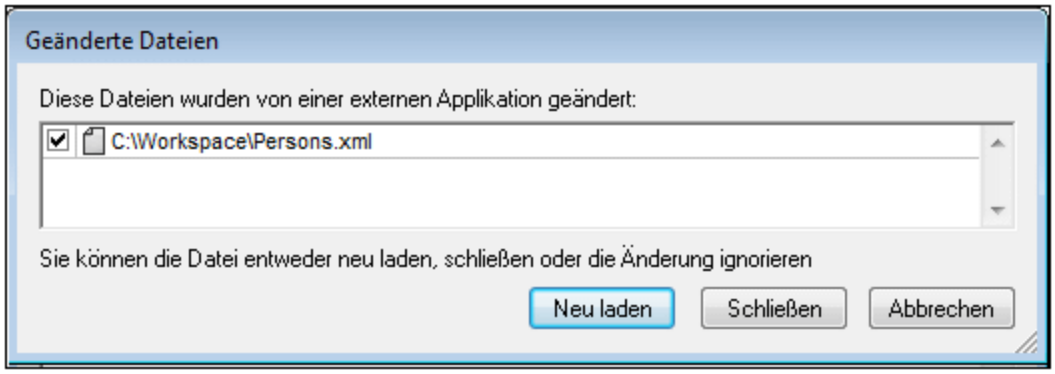
Mit dem Befehl **Abrufen** (im Menü **Projekt | Versionskontrolle**) werden Dateien aus dem Repository als schreibgeschützte Dateien abgerufen. (Um eine Datei bearbeiten zu können, müssen Sie sie auschecken[1127].) Im Dialogfeld "Abrufen" werden die Dateien im Objekt (Projekt oder Ordner), an denen der Befehl "Abrufen" ausgeführt wurde, aufgelistet (*siehe Abbildung unten*). Sie können die abzurufenden Dateien markieren, indem Sie sie im Dialogfeld "Abrufen" mit einem Häkchen versehen.

Anmerkung: Mit dem Befehl **Ordner abrufen** können Sie einzelne Unterordner im Repository auswählen, falls dies Ihr Versionskontrollsystem gestattet.



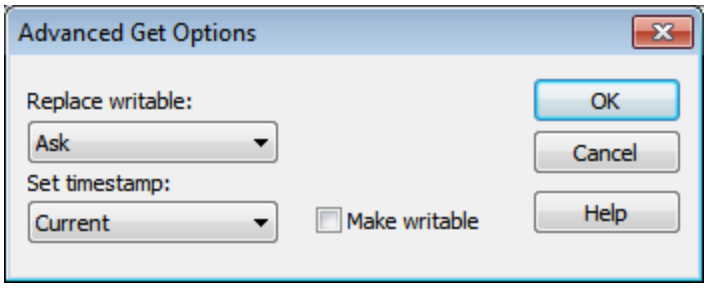
Sie können auswählen, ob geänderte ausgecheckte Dateien überschrieben werden sollen, indem Sie die entsprechende Option am unteren Rand des Dialogfelds "Abrufen" aktivieren. Wenn Sie auf **OK** klicken, werden die Dateien überschrieben. Wenn eine der überschriebenen Dateien gerade geöffnet ist, wird ein Dialogfeld angezeigt (*Abbildung unten*), in dem Sie gefragt werden, ob die Datei(en) neu geladen (Schaltfläche **Neu laden**), geschlossen (Schaltfläche **Schließen**) werden sollen oder ob die aktuelle Ansicht der Datei beibehalten werden soll (**Abbrechen**).

© 2016-2022 Altova GmbH



Erweiterte Abrufoptionen

Das Dialogfeld "Erweiterte Abrufoptionen" (*Abbildung unten*) wird über die Schaltfläche **Erweitert** im Dialogfeld "Abrufen" (*siehe erste Abbildung in diesem Abschnitt*) aufgerufen.



Hier können Sie Optionen zum (i) Ersetzen ausgecheckter beschreibbarer Dateien (ii) zum Zeitstempel und (iii) ob die Eigenschaft "Schreibgeschützt" der abgerufenen Datei geändert werden soll, damit die Datei beschreibbar wird, einstellen.

Aktuellste Version holen

Mit dem Befehl **Aktuellste Version holen** (im Menü **Projekt | Versionskontrolle**) wird die neueste Version der ausgewählten Datei(en) abgerufen und in das Arbeitsverzeichnis platziert. Die Dateien werden schreibgeschützt abgerufen und nicht ausgecheckt. Dieser Befehl funktioniert wie der Befehl **Abrufen** (*siehe oben*), doch wird das Dialogfeld "Abrufen" nicht angezeigt.

Wenn die ausgewählten Dateien derzeit ausgecheckt sind, hängt es von Ihrem Versionskontrollsystem ab, was geschieht. Normalerweise werden Sie gefragt, ob die ausgecheckte Datei ersetzt, zusammengeführt oder so belassen werden soll.

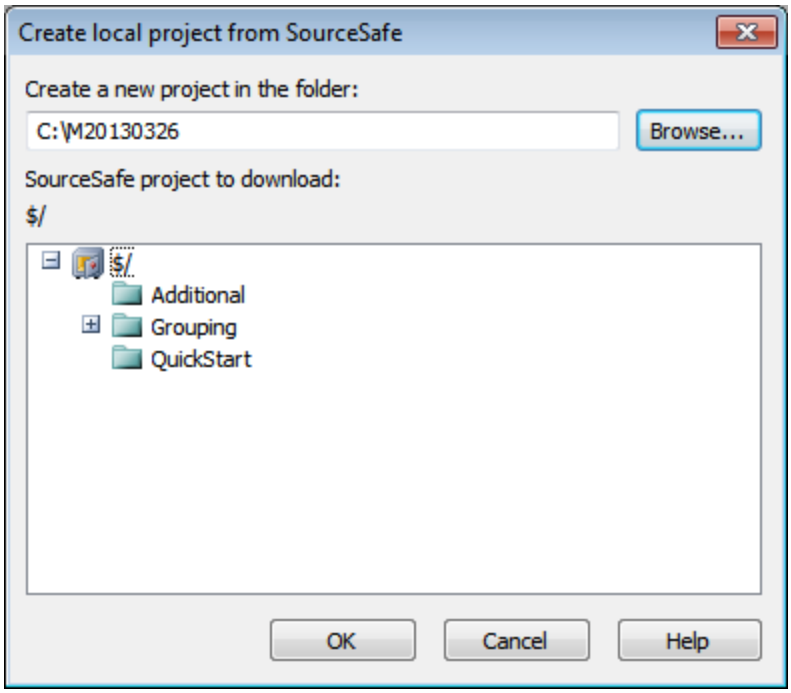
Anmerkung: Dieser Befehl ist bei Ausführung an einem Ordner rekursiv, d.h. er wird auf alle Dateien unterhalb des aktuellen Ordners in der Ordnerhierarchie angewendet.

24.6.4 Kopieren und Freigeben über die Versionskontrolle

Mit dem Befehl **Aus Versionskontrolle öffnen** erstellen Sie ein neues Applikationsprojekt von einem unter Versionskontrolle gestellten Projekt.

Um ein neues Projekt zu erstellen, gehen Sie folgendermaßen vor:

1. Je nach verwendetem Versionskontrollsystem müssen Sie vor dem Erstellen eines neuen Projekts über die Versionskontrolle sicherstellen, dass keine Datei aus dem unter Versionskontrolle stehenden Projekt ausgecheckt ist.
2. In der Applikation kann ein Projekt geöffnet sein, dies muss aber nicht der Fall sein.
3. Wählen Sie den Befehl **Projekt | Versionskontrolle | Aus Versionskontrolle öffnen**.
4. Das aktuell eingestellte Versionskontrollsystem wird mit den jeweiligen Überprüfungs- und Verbindungsdialogfeldern angezeigt. Stellen Sie eine Verbindung zum gewünschten Repository, d.h. zum gebundenen Ordner im Repository, der kopiert werden soll, 1121 her.
5. Navigieren Sie im Dialogfeld, das daraufhin angezeigt wird (*Abbildung unten*), zum lokalen Ordner, in den der Inhalt des gebundenen Ordners im Repository (zu dem Sie soeben eine Verbindung hergestellt haben) kopiert werden soll. In der Abbildung unten hat der gebundene Ordner den Namen `MyProject` und wird durch das Zeichen `$` repräsentiert; der lokale Ordner ist `C:\M20130326`.



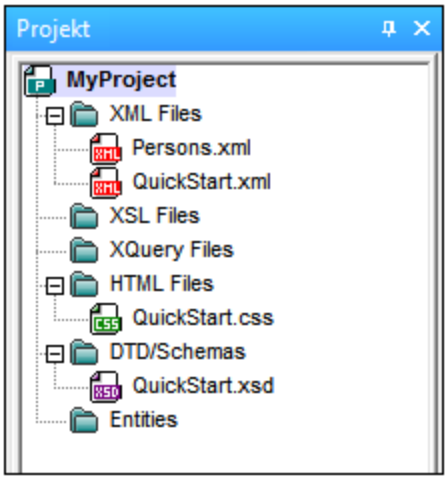
6. Klicken Sie auf **OK**. Der Inhalt des gebundenen Ordners (`MyProject`) wird in den lokalen Ordner `C:\M20130326`. kopiert und ein Dialogfeld wird aufgerufen, in dem Sie aufgefordert werden, die Projektdatei (`.spp` Datei) auszuwählen, die als das neue Projekt erstellt werden soll.
7. Wählen Sie die `.spp` Datei aus, die in den lokalen Ordner kopiert worden sein sollte. In unserem Beispiel handelt es sich um `MyProject.spp` im Ordner `C:\M20130326`. In der Applikation wird ein neues Projekt namens `MyProject` erstellt und im Fenster "Projekt" angezeigt. Die Dateien des Projekts werden im Ordner `C:\M20130326` gespeichert.

Freigeben über die Versionskontrolle

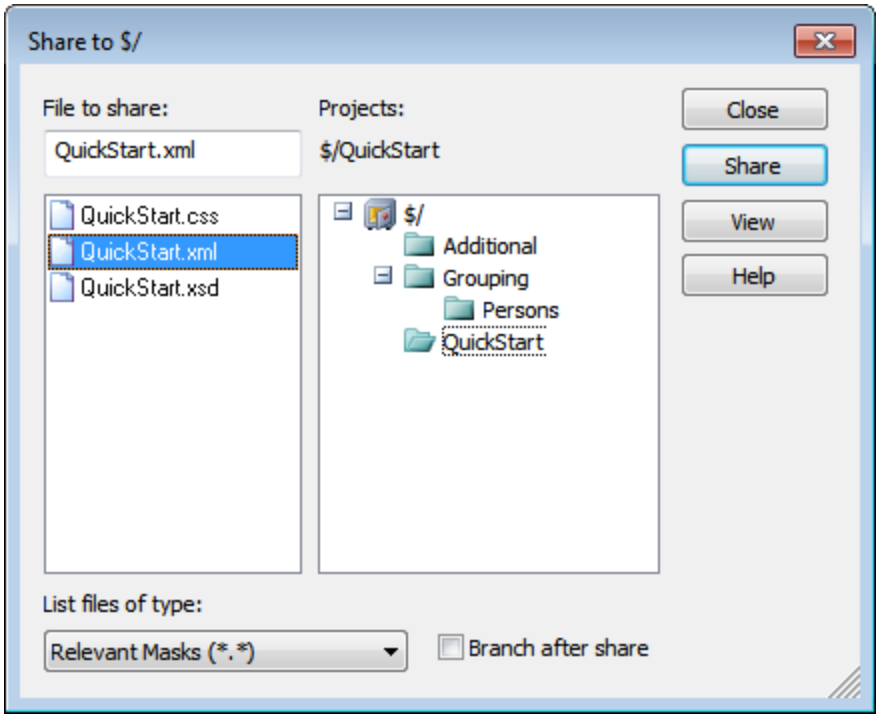
Der Befehl **Aus Versionskontrolle freigeben** wird unterstützt, wenn das verwendete Versionskontrollsystem die Freigabe von Dateien unterstützt. Sie können eine Datei freigeben, sodass sie auf mehreren lokalen

Rechnern verfügbar ist. Wenn an einer dieser Dateien eine Änderung vorgenommen wird, wird dies in allen anderen freigegebenen Versionen übernommen.

Wählen Sie das (*in der Abbildung unten markierte)* Projekt im Fenster "Projekt" der Applikation aus und klicken Sie anschließend auf den Befehl **Aus Versionskontrolle freigeben**.



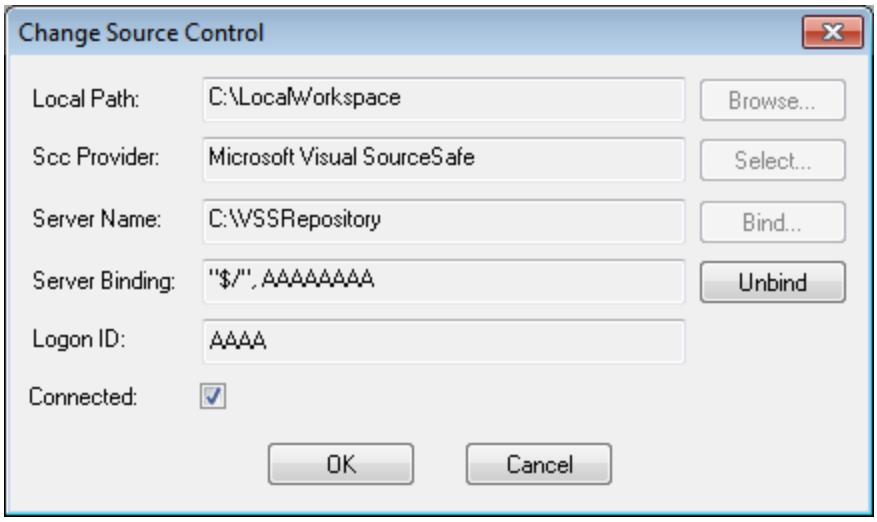
Daraufhin wird das Dialogfeld "Share To [Ordner]" (*Abbildung unten*) angezeigt.



Um die gewünschten Dateien auszuwählen, wählen Sie zuerst den Ordner, in dem sich die Dateien befinden, in der Projektstruktur auf der rechten Seite des Dialogfelds (*siehe Abbildung oben*) aus. Die Dateien im ausgewählten Ordner werden im Fenster auf der linken Seite angezeigt. Wählen Sie die freizugebende Datei aus (um mehrere Dateien auszuwählen, halten Sie beim Anklicken die **Strg**-Taste gedrückt). Die ausgewählten Datei(en) werden im Textfeld *Files to Share* (*links oben*) angezeigt. Die Dateien verschwinden aus dem Fenster

auf der linken Seite. Klicken Sie auf **Freigeben** und anschließend **Schließen,** um die ausgewählte(n) Datei(en) in den ausgewählten lokalen Ordner zu kopieren.

Der freigegebene Ordner wird im Namen des Dialogfelds "Share to [Ordner]" angegeben. In der Abbildung oben ist es der lokale Ordner (da sich das Zeichen `$` in dem Ordner im Repository, an den der lokale Ordner gebunden ist, befindet). Sie können den freigegebenen Ordner im Dialogfeld "Versionskontrolle wechseln" (*Abbildung unten,* **Versionskontrolle wechseln***)* sehen und den Ordner durch Ändern des lokalen Pfads und Binding einstellen.



Nähere Informationen zum Freigeben von Dateien über Ihr Versionskontrollsystem finden Sie in der Benutzerdokumentation zu Ihrem Versionskontrollsystem.

24.6.5 Wechseln der Versionskontrolle

Die Versionskontrolleinstellungen können über zwei Befehle im Menü **Projekt | Versionskontrolle** geändert werden:

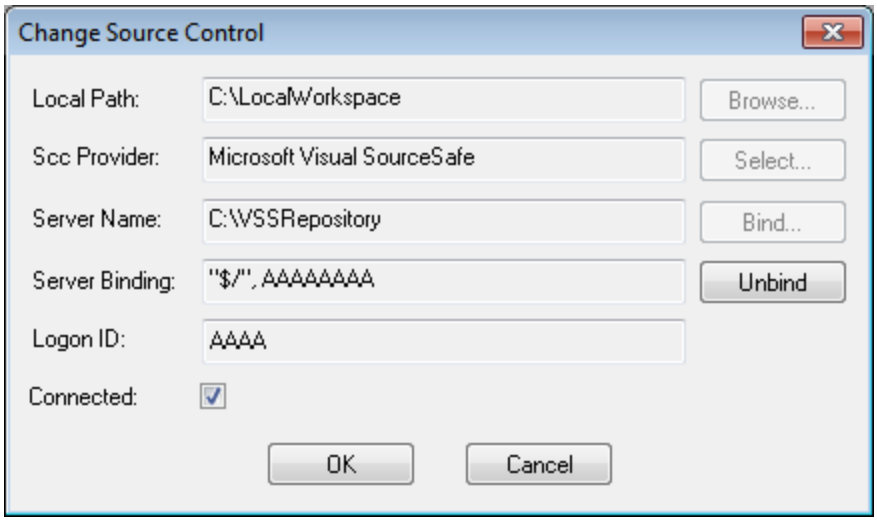
- **Versionskontrollmanager:** Öffnet die Versionskontrollsystemapplikation und ermöglicht Ihnen, Datenbanken einzurichten und Bindings zu konfigurieren.
- **Versionskontrolle wechseln:** Ruft das Dialogfeld "Versionskontrolle wechseln" auf, in dem Sie das von der Altova Applikation verwendete Versionskontrollsystem und das aktuelle Binding wechseln können. Dieses Dialogfeld ist im Folgenden beschrieben.

Über das aktuelle Binding stellt das aktive Applikationsprojekt eine Verbindung zur Versionskontrolldatenbank her. Das aktuelle Binding ist korrekt sein, wenn sich die Applikationsprojektdatei (`.spp` Datei) im lokalen Ordner und der gebundene Ordner in dem Repository befindet, in dem die Projektdateien gespeichert werden sind. Normalerweise entsprechen der gebundene Ordner und seine Substruktur dem lokalen Arbeitsbereichordner und dessen Substruktur.

Im Dialogfeld "Versionskontrolle wechseln" (*Abbildung unten*) können Sie das Versionskontrollsystem (*SCC Provider*), den lokalen Ordner (*Lokaler Pfad*) und das Repository-Binding (*Server-Name* und *Server Binding*) wechseln.

© 2016-2022 Altova GmbH

Die Einstellungen können erst geändert werden, wenn Sie die aktuelle Bindung aufgehoben haben. Verwenden Sie dazu die Schaltfläche **Bindung aufheben**. Alle Einstellungen können nun bearbeitet werden.



Ändern Sie die Versionskontrolleinstellungen folgendermaßen:

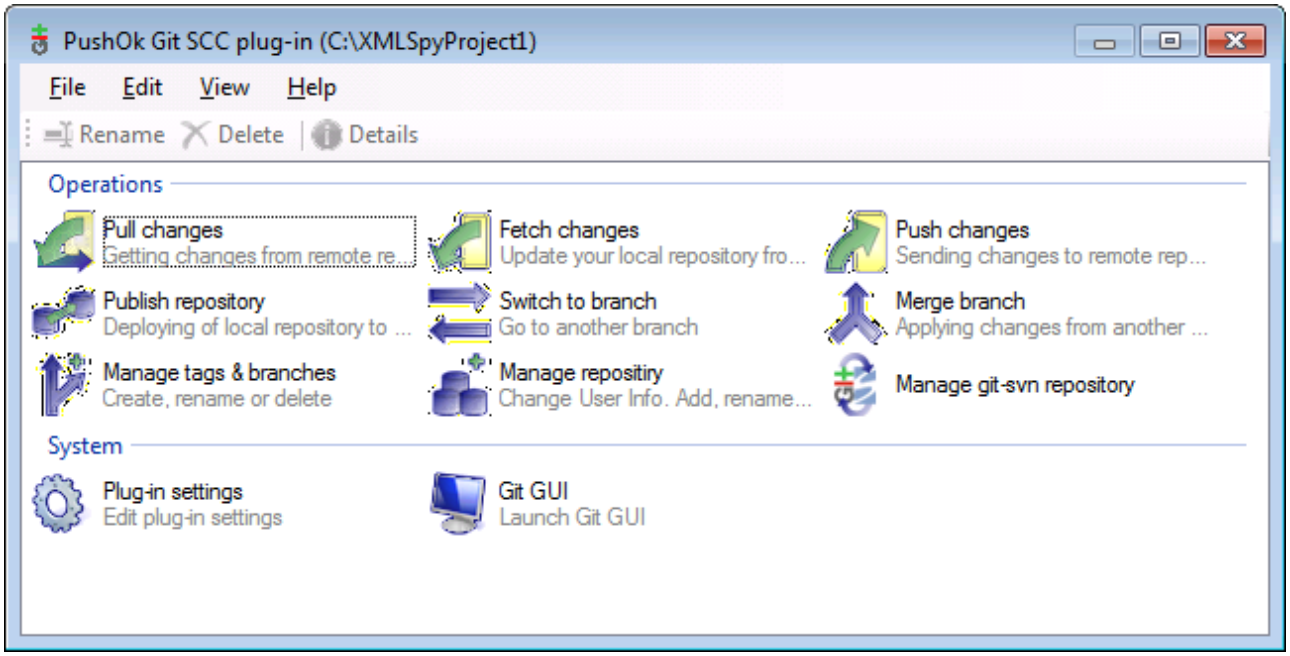
1. Navigieren Sie über die Schaltfläche **Durchsuchen** zum lokalen Ordner und klicken Sie auf die Schaltfläche **Auswählen,** um eines der installierten Versionskontrollsysteme auszuwählen.
2. Anschließend können Sie den lokalen Ordner an eine Repository-Datenbank binden. Klicken Sie dazu auf die Schaltfläche **Binden**. Daraufhin wird das Verbindungsdialogfeld Ihres Versionskontrollsystems aufgerufen.
3. Wenn Sie eine *Anmelde-ID* eingegeben haben, wird diese an das Versionskontrollsystem übergeben; andernfalls müssen Sie Ihre Anmeldeinformationen im Verbindungsdialogfeld eingeben.
4. Wählen Sie im Repository die Datenbank aus, an die Sie diesen lokalen Ordner binden möchten. Eventuell müssen diese Einstellungen in mehreren Dialogfeldern vorgenommen werden.
5. Nachdem Sie alle Einstellungen vorgenommen haben, klicken Sie im Dialogfeld "Versionskontrolle wechseln" auf **OK**.

24.7 Versionskontrolle mit Git

Unterstützung für Git als Versionskontrollsystem in XMLSpy steht in Form eines Drittanbieter-Plug-in namens **GIT SCC Plug-in** (http://www.pushok.com/software/git.html) zur Verfügung.

> Zum Zeitpunkt der Verfassung dieser Dokumentation steht das **GIT SCC Plug-in** zum Experimentieren zur Verfügung. Um das Plug-in verwenden zu können, müssen Sie beim Plug-in-Anbieter registriert sein.

Mit Hilfe des GIT SCC Plug-in können Sie über die Befehle im Menü **Projekt | Versionskontrolle** von XMLSpy mit einem Git Repository arbeiten. Beachten Sie, dass die Befehle im Menü **Projekt | Versionskontrolle** von XMLSpy von der Microsoft Source Control Plug-in API (MSSCCI API), für die eine andere Art von Design als von Git verwendet wird, bereitgestellt werden. Daher bildet das Plug-in eine Zwischenschaltung zwischen "Visual Source Safe"-Funktionalitäten und Git-Funktionalitäten. Das bedeutet einerseits, dass ein Befehl wie z.B. **Aktuellste Version holen** für Git eventuell so nicht verwendet werden kann. Andererseits gibt es einige neue Git-spezifische Aktionen, die über das Plug-in im Dialogfeld "Versionskontrollmanager" zur Verfügung gestellt werden (Menü **Projekt | Versionskontrolle | Versionskontrollmanager** von XMLSpy).



Das Dialogfeld "Versionskontrollmanager"

Andere häufig benötigte Versionskontrollbefehle stehen direkt im Menü **Projekt | Versionskontrolle** zur Verfügung.

In den folgenden Abschnitten wird die Anfangskonfiguration des Plug-in sowie der grundlegende Arbeitsablauf beschrieben:

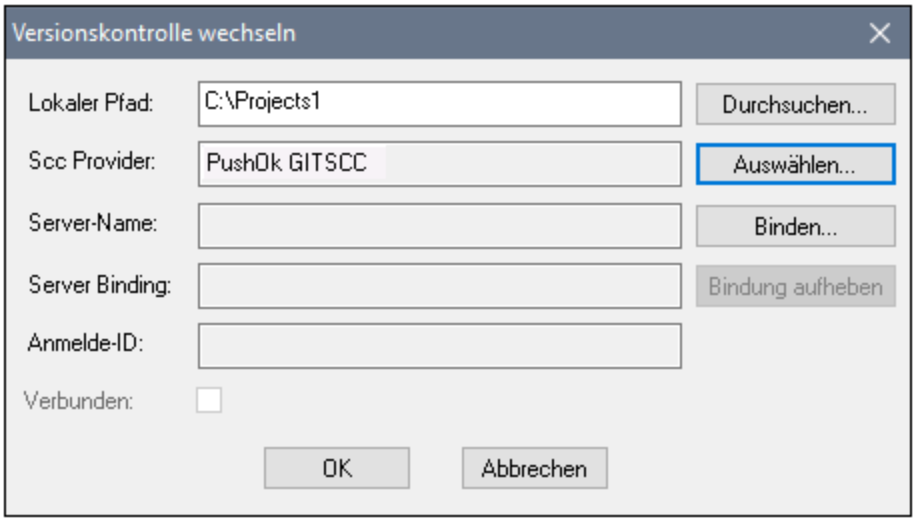
- Aktivieren der Git-Versionskontrolle mit dem Git SCC Plug-in 1137
- Hinzufügen eines Projekts zur Git-Versionskontrolle 1137
- Klonen eines Projekts anhand der Git-Versionskontrolle 1139

© 2016-2022 Altova GmbH

24.7.1 Aktivieren der Git-Versionskontrolle mit dem Git SCC Plug-in

Um die Git-Versionskontrolle mit XMLSpy zu aktivieren, muss zuerst das Drittanbieter-**PushOK GIT SCC Plug-in** installiert, registriert und als Versionskontrollanbieter ausgewählt werden. Dies geschieht folgendermaßen:

1. Laden Sie die Installationsdatei für das Plug-in von der Website des Anbieters (http://www.pushok.com) herunter, starten Sie sie und befolgen Sie die Installationsanweisungen.
2. Klicken Sie im Menü **Projekt** von XMLSpy auf **Versionskontrolle wechseln** und vergewissern Sie sich, dass **PushOk GITSCC** als Versionskontroll-Provider ausgewählt ist. Wenn **Push Ok GITSCC** in der Liste der Provider nicht angezeigt wird, war die Installation des Plug-in wahrscheinlich nicht erfolgreich. Lesen Sie in diesem Fall in der Dokumentation des Anbieters nach, wie Sie das Problem lösen können.



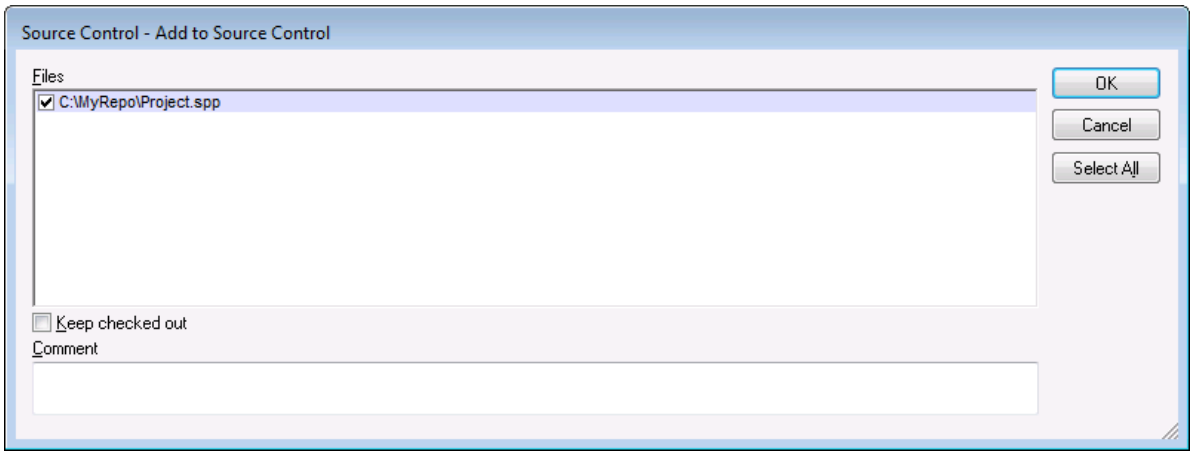
3. Wenn ein Dialogfeld angezeigt wird, in dem Sie aufgefordert werden, das Plug-in zu registrieren, klicken Sie auf **Registrierung** und befolgen Sie die Anweisungen des Assistenten, um die Registrierung abzuschließen.

24.7.2 Hinzufügen eines Projekts zur Git-Versionskontrolle

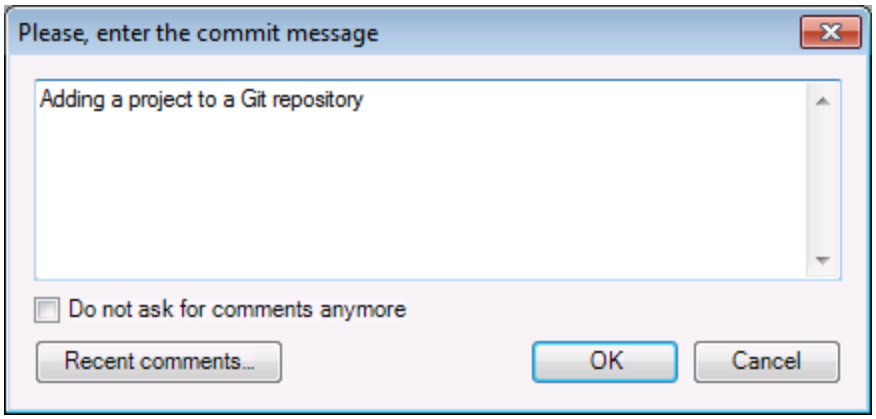
Sie können XMLSpy -Projekte als Git Repositories speichern. Die Struktur der zum Projekt hinzugefügten Dateien oder Ordner würde anschließend der Struktur des Git Repository entsprechen.

So fügen Sie ein Projekt zu einer Git-Versionskontrolle hinzu:

1. Stellen Sie sicher, dass das **PushOK GIT SCC Plug-in** als Versionskontrollanbieter ausgewählt ist (siehe Aktivieren der Git-Versionskontrolle mit dem Git SCC Plug-in 1137).
2. Erstellen Sie mit dem Menübefehl **Projekt | Projekt erstellen** ein neues Projekt.
3. Speichern Sie das Projekt in einem lokalen Ordner, z.B. unter `C:\MyRepo\Project.spp`
4. Klicken Sie im Menü **Projekt** unter **Versionskontrolle** auf **Zu Versionskontrolle hinzufügen**.

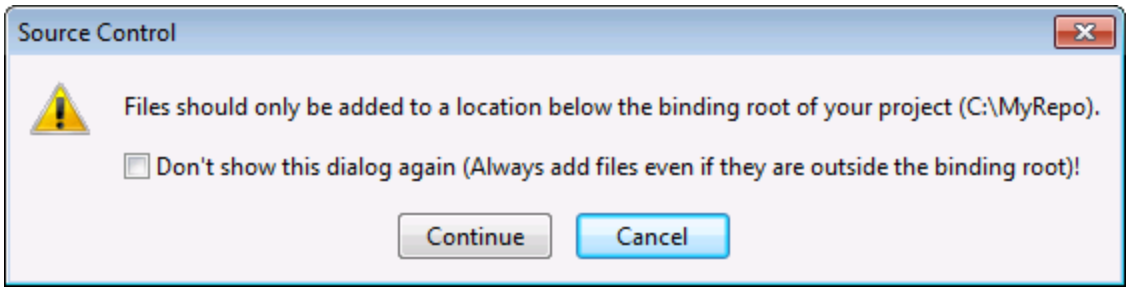


5. Klicken Sie auf **OK**.



6. Geben Sie den Text Ihrer Commit-Meldung ein und klicken Sie auf **OK**.

Sie können jetzt Dateien und Ordner zu Ihrem Projekt hinzufügen. Beachten Sie, dass alle Projektdateien und -ordner sich unter dem Root-Ordner des Projekts befinden müssen. Wenn das Projekt z.B. im Ordner `C:\MyRepo` angelegt wurde, so sollten nur Dateien unter `C:\MyRepo` zur Projekt hinzugefügt werden. Wenn Sie versuchen, Dateien, die sich außerhalb des Projekt-Root-Ordners befinden, zum Projekt hinzuzufügen, wird eine Warnmeldung angezeigt:

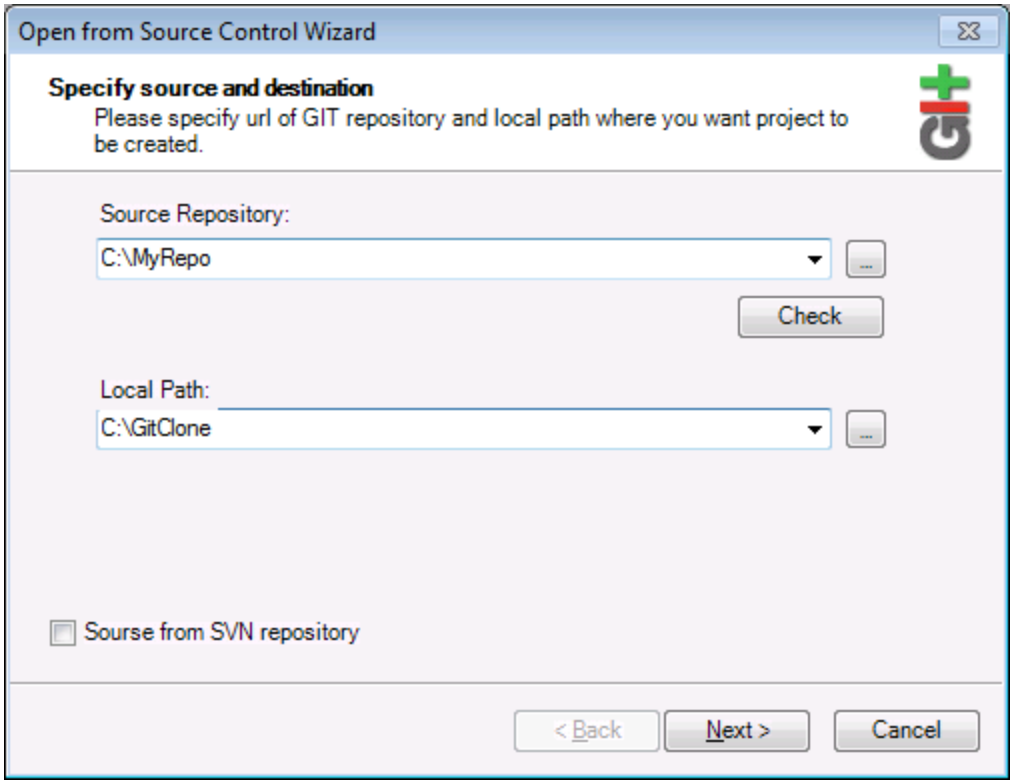


 © 2016-2022 Altova GmbH

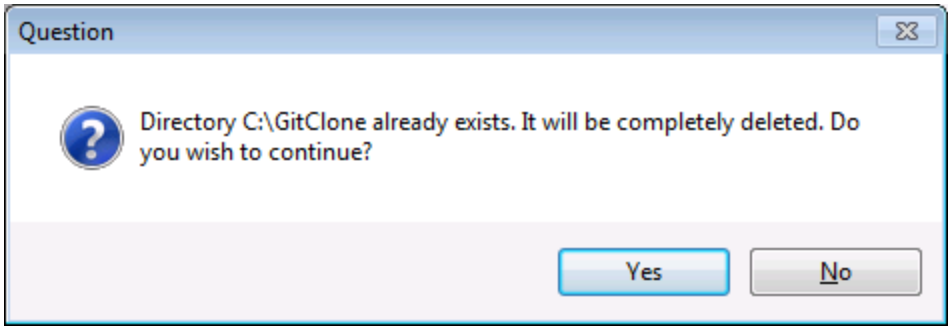
24.7.3 Klonen eines Projekts anhand der Git-Versionskontrolle

Projekte, die bereits zur Git-Versionskontrolle hinzugefügt wurden (siehe Hinzufügen eines Projekts zur Git-Versionskontrolle 1137) können folgendermaßen über das Git Repository geöffnet werden:

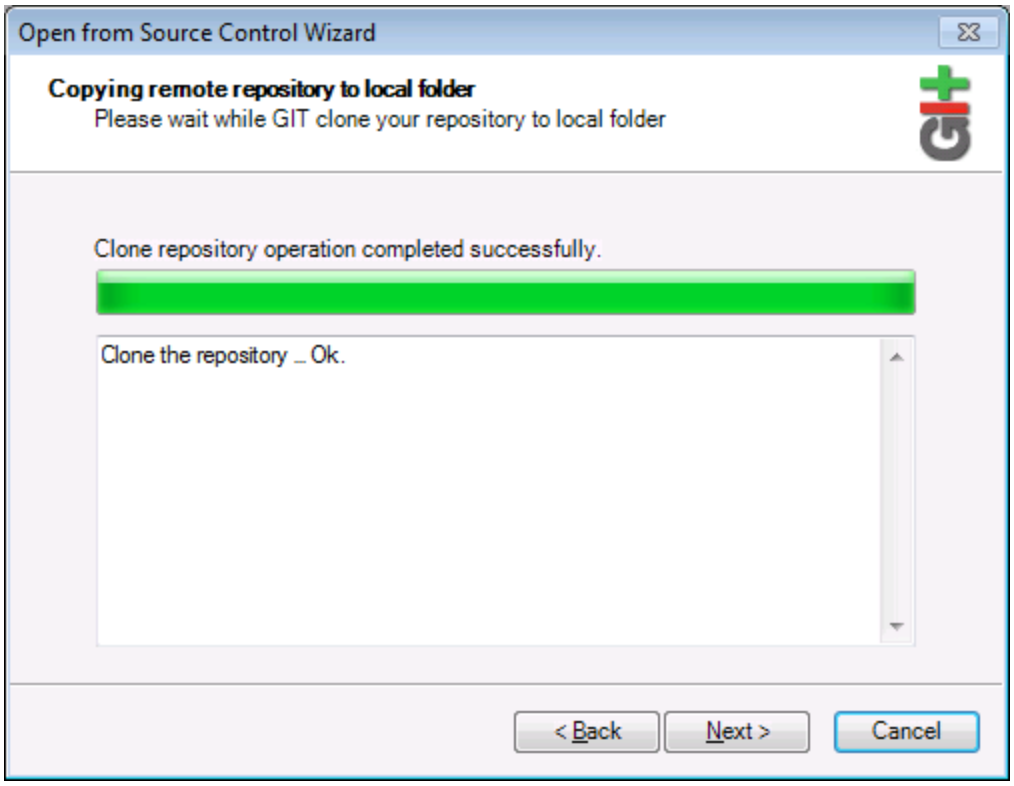
1. Vergewissern Sie sich, dass das **PushOK GIT SCC Plug-in** als Versionskontroll-Provider ausgewählt ist (siehe Aktivieren der Git-Versionskontrolle mit dem Git SCC Plug-in 1137).
2. Klicken Sie im Menü **Projekt** auf **Versionskontrolle | Aus Versionskontrolle öffnen**.
3. Geben Sie den Pfad oder die URL des Versionskontroll-Repository ein. Klicken Sie auf **Überprüfen,** um die Gültigkeit des Pfads oder der URL zu überprüfen.



4. Geben Sie unter **Lokaler Pfad** den Pfad zu dem lokalen Ordner ein, in dem das Projekt erstellt werden soll und klicken Sie auf **Weiter**. Wenn der lokale Ordner vorhanden ist, wird (auch wenn er leer ist) das folgende Dialogfeld aufgerufen:



5. Klicken Sie zur Bestätigung auf **Ja** und anschließend auf **Weiter**.



6. Stellen Sie die restlichen Schritte des Assistenten fertig, wie für Ihr Projekt erforderlich.
7. Nach der Fertigstellung wird ein Durchsuchen-Dialogfeld angezeigt, in dem Sie aufgefordert werden, das XMLSpy-Projekt (*.spp)-Datei zu öffnen. Wählen Sie die Projektdatei aus, um den Projektinhalt in XMLSpy zu laden.

© 2016-2022 Altova GmbH

25 XMLSpy in MS Visual Studio

Sie können XMLSpy in die Microsoft Visual Studio IDE-Versionen 2012/2013/2015/2017/2019/2022 integrieren. Auf diese Weise werden die komplexen XML-Bearbeitungsfunktionen mit den umfangreichen Entwicklungsumgebung von Visual Studio kombiniert.

In diesem Abschnitt wird Folgendes beschrieben:

- der allgemeine Installationsvorgang[1142] und die Integration des XMLSpy Plugin in Visual Studio.
- Unterschiede[1143] zwischen der Visual Studio-Version und der Standalone-Version.
- Die Debugger von XMLSpy[1146] in Visual Studio.

25.1 Installieren des XMLSpy Plugin

Um das XMLSpy Plugin für Visual Studio zu installieren, gehen Sie folgendermaßen vor:

1. Installieren Sie Microsoft Visual Studio 2012/2013/2015/2017/2019/2022. Beachten Sie, dass Visual Studio ab Visual Studio 2022 nur als 64-Bit-Applikation zur Verfügung steht.
2. Installieren Sie XMLSpy (Enterprise oder Professional Edition). Wenn Sie Visual Studio 2022+ installiert haben, müssen Sie die 64-Bit-Version von XMLSpy installieren.
3. Downloaden Sie das XMLSpy Integrationspaket für Microsoft Visual Studio. Dieses Paket steht auf der XMLSpy (Enterprise und Professional Edition) Download-Seite unter www.altova.com zur Verfügung.

Nach Installation des Integrationspakets können Sie XMLSpy in der Visual Studio-Umgebung verwenden.

Achtung

Sie müssen das richtige Integrationspaket für Ihre jeweilige XMLSpy Version (die aktuelle Version ist 2023) verwenden. Das Integrationspaket ist nicht editionsspezifisch und daher sowohl für die Enterprise als auch die Professional Edition verwendet werden.

© 2016-2022 Altova GmbH

25.2 Unterschiede zur XMLSpy-Standalone-Version

In diesem Abschnitt wird beschrieben, inwiefern sich die Visual Studio Versionen von den Standalone-Versionen von XMLSpy unterscheiden. Die Liste beginnt mit Funktionen, die in der Visual Studio-Version nicht unterstützt werden, anschließend werden andere Punkte aufgelistet, in denen sich die Visual Studio-Version von der Standalone-Version unterscheidet.

- Nicht unterstützte Funktionen in Visual Studio 1143
- Zusätzliche XMLSpy Menüs in Visual Studio 1143
- Eingabehilfe in Visual Studio 1143
- Dieselbe Funktionalität, ein anderer Befehl 1144
- XMLSpy-Befehle als Visual Studio-Befehle 1144

Nicht unterstützte Funktionen in Visual Studio

Die folgenden XMLSpy Funktionen stehen in Visual Studio nicht zur Verfügung:

- Die Skripting-Umgebung (**Extras | XMLSpy Optionen | Skripting**) wird derzeit nicht unterstützt. Daher werden Symbolleisten-Schaltflächen, die zur Ausführung von Skripts erstellt wurden, nicht angezeigt.
- Die Textstatussymbole der Authentic 606-Ansicht werden nicht unterstützt.
- Separates Browser-Fenster (eine Option im Abschnitt "Ansicht" (**Extras | Optionen)** wird nicht unterstützt. Das bedeutet, dass sich die Text- und die Browser-Ansicht immer im selben Fenster befinden
- Alle Versionskontrollfunktionen
- Alle Vergleichsfunktionen (im Menü **Extras** der Standalone-Version)

Zusätzliche XMLSpy Menüs in Visual Studio

Die folgenden Befehle sind in Visual Studio XMLSpy-spezifisch:

- **Ansicht | XMLSpy Tool Windows**
- **Ansicht | XMLSpy-Ansicht**
- **XMLSpy** (enthält die Einträge des Menüs "Globale Ressourcen" und die Möglichkeit XMLSpy-Designs zu wechseln)
- **Extras | XMLSPY-Optionen**

Anmerkung: In Versionen ab Visual Studio 2019 können XMLSpy-Funktionalitäten über das Menü **Erweiterungen** von Visual Studio aufgerufen werden. In früheren Versionen von Visual Studio, XMLSpy stehen die Funktionen in Visual Studio-Menüs der obersten Ebene zur Verfügung.

Eingabehilfen (Tool-Fenster in Visual Studio)

Die Eingabehilfen von XMLSpy stehen in Visual Studio als Tools-Fenster zur Verfügung. Beachten Sie dazu die folgenden Punkte. (Eine Beschreibung der Eingabehilfen und der Benutzeroberfläche von XMLSpy finden Sie im Abschnitt Die grafische Benutzeroberfläche 115).

- Sie können Eingabehilfenfenster an jede beliebige Stelle in der Entwicklungsumgebung ziehen.
- Wenn Sie mit der rechten Maustaste auf das Register einer Eingabehilfe klicken, können Sie die Benutzeroberfläche weiter anpassen. Die Konfigurationsoptionen für Eingabehilfen sind: andockbar, ausblenden, frei schwebend und automatisch ausblenden.

Dieselbe Funktionalität, ein anderer Befehl

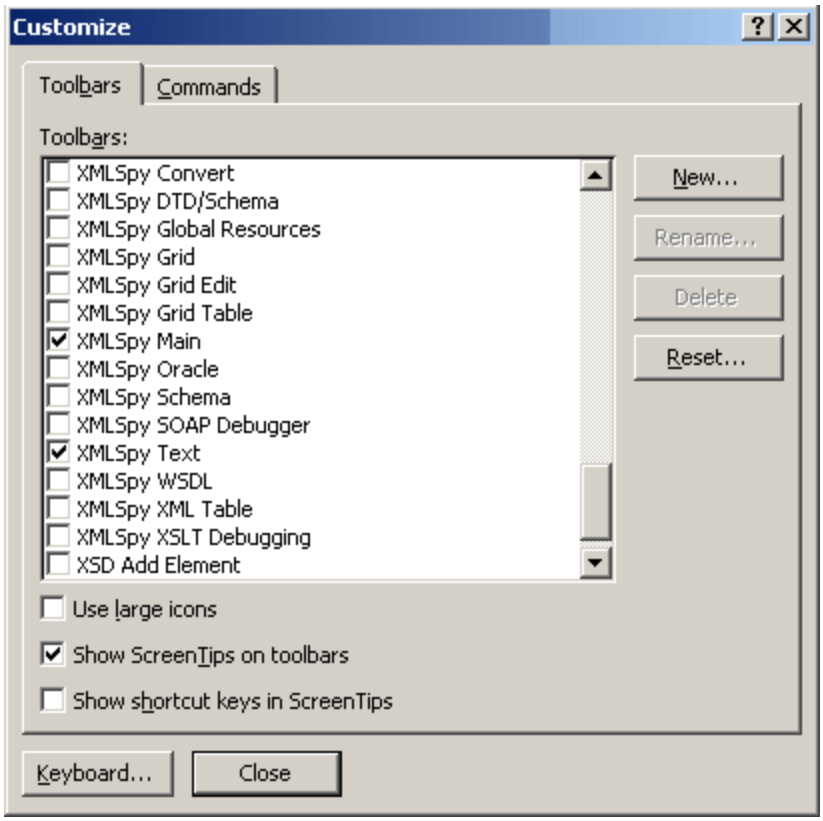
Einige Funktionalitäten von XMLSpy stehen in Visual Studio unter einem anderen Namen zur Verfügung. Diese Befehle sind:

XMLSpy	Visual Studio	Funktionalität
Datei | Öffnen | Zu URL wechseln	Datei | Öffnen | Website	Öffnet die Datei über eine URL
Zu URL wechseln | URL speichern	Datei | XMLSpy-Datei unter URL speichern	Speichert die Datei unter einer URL

XMLSpy-Befehle als Visual Studio-Befehle

Einge XMLSpy-Befehle stehen auf der Benutzeroberfläche von Visual Studio als Visual Studio-Befehle zur Verfügung. Diese Befehle sind:

- ***Rückgängig, Wiederherstellen:*** Diese Visual Studio-Befehle wirken sich auf alle Aktionen in der Visual Studio Entwicklungsumgebung aus.
- ***Projekte:*** XMLSpy-Projekte werden als Visual Studio-Projekte behandelt.
- ***Symbolleisten anpassen, Befehle anpassen:*** Die Register "Symbolleisten" und "Befehle" im Dialogfeld "Anpassen" (*Abbildung unten*) (**Extras | Anpassen**) enthalten sowohl Befehle aus Visual Studio als auch XMLSpy.



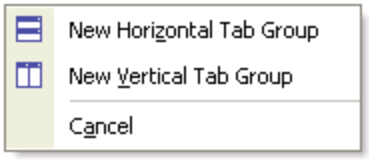
 © 2016-2022 Altova GmbH

- ***Ansichten:*** Im Menü **Ansicht** enthalten die beiden Befehle **XMLSpy Tool-Fenster** und **XMLSpy-Ansicht** Optionen zum Ein- und Ausblenden der Eingabehilfenfenster und anderer Seitenleisten und zum Wechseln zwischen den Bearbeitungsansichten und zum Ein- und Ausblenden von Bearbeitungshilfen.
- ***XMLSpy Hilfe:*** Dieses XMLSpy Menü wird als Untermenü im **Hilfemenü** von Visual Studio angezeigt.

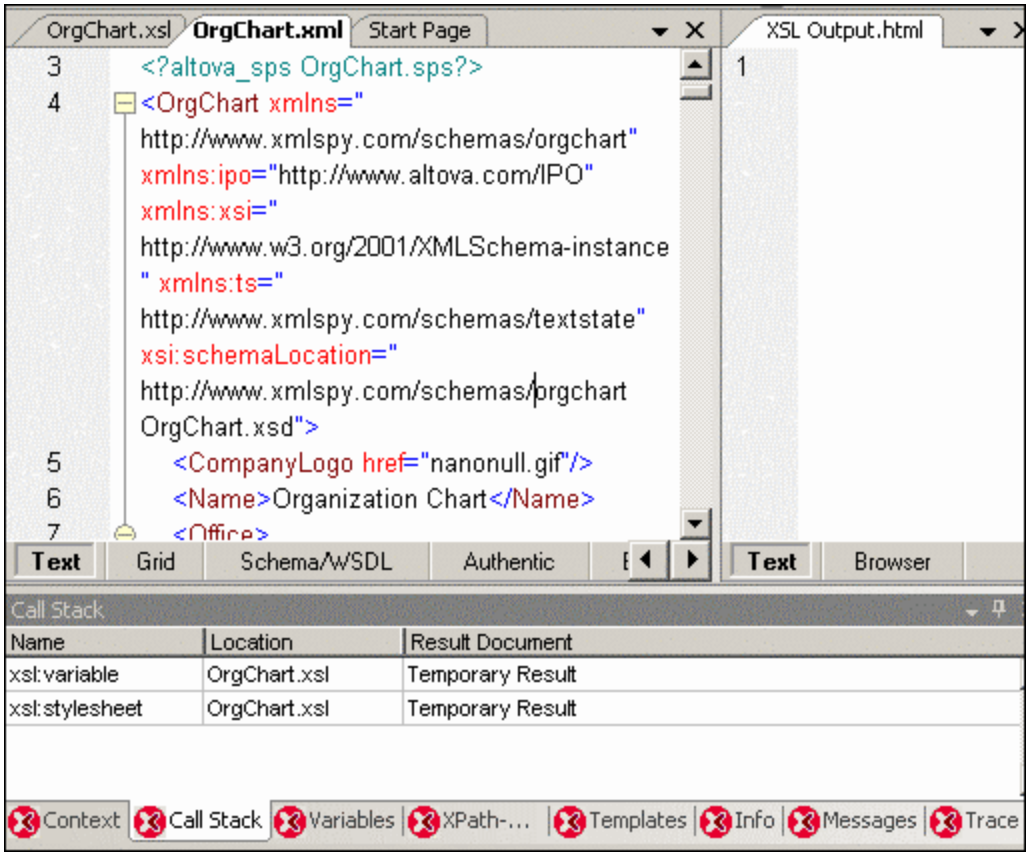
25.3 Die XMLSpy Debugger in Visual Studio

XMLSpy enthält einen XSLT/XQuery Debugger (*Enterprise und Professional Edition*) und einen SOAP Debugger (*Enterprise Edition*). Beim Debugging müssen mehrere Dateien (z.B. XML-, XSLT- und XSLT-Ausgabedatei) angezeigt werden. Alle diese Dateien werden in Visual Studio auf einer Registerkartengruppe angezeigt. Um den Debugging-Vorgang übersichtlicher zu gestalten, können Sie in Visual Studio eine oder mehrere zusätzliche Registerkartengruppen anlegen. Gehen Sie dabei folgendermaßen vor:

1. Klicken Sie auf das Register, das Sie aus der Registerkartengruppe herausnehmen möchten und ziehen Sie es auf das aktuell aktive Register. Hierauf wird ein Popup-Menü angezeigt, in dem Sie die Art der zu erstellenden Registerkarte definieren können.



2. Wählen Sie die Option **New Vertical Tab Group**. Daraufhin wird eine neue Registerkarte angelegt, die nur das ausgewählte Register enthält (*siehe Abbildung unten*).



 © 2016-2022 Altova GmbH

26 XMLSpy in Eclipse

Eclipse ist eine Open-Source-Umgebung, in die verschiedene Arten von Applikationen in Form von Plug-Ins integriert werden können. Mit Hilfe des XMLSpy-Integrationspakets für Eclipse können Sie die Funktionalitäten von XMLSpy in die Eclipse-Plattform für Windows integrieren und aufrufen. Die folgenden Eclipse-Versionen werden unterstützt: 2022-09, 2022-06, 2022-03, 2021-12.

In diesem Abschnitt werden folgende Punkte erläutert:

- Installation des XMLSpy-Integrationspakets für Eclipse und Integration von XMLSpy in Eclipse 1148
- Die XMLSpy-Perspektive in Eclipse 1150
- Andere XMLSpy-Einstiegspunkte in Eclipse 1153
- XMLSpy Debugger-Perspektiven 1155

Anmerkung: Die Versionskontrollfunktionen, die in der Standalone-Version von XMLSpy verfügbar sind, werden in der Eclipse-Version nicht unterstützt.

26.1 Installation des XMLSpy Plugin für Eclipse

Voraussetzungen

- Eclipse 2022-09, 2022-06, 2022-03, 2021-12 (http://www.eclipse.org), 64-Bit.
- Java Runtime Environment (JRE) oder Java Development Kit (JDK) für die 64-Bit-Plattform.
- XMLSpy Enterprise oder Professional Edition 64-Bit.

Anmerkung: Alle oben aufgelisteten Programme müssen die 64-Bit-Plattform haben. Die Integration mit älteren Eclipse 32-Bit-Plattformen wird nicht mehr unterstützt, funktioniert aber eventuell noch.

Mit den obigen Voraussetzungen können Sie das XMLSpy Integrationspaket (64-Bit) installieren, um XMLSpy in Eclipse zu integrieren. Die Integration kann entweder während der Installation des Installationspakets oder nach Installation des Integrationspakets manuell von Eclipse aus durchgeführt werden. Das XMLSpy Integrationspaket kann von https://www.altova.com/de/components/download heruntergeladen werden.

Anmerkung: Eclipse muss während der Installation bzw. Deinstallation des XMLSpy Integrationspakets geschlossen sein.

Integration von XMLSpy während der Installation des Integrationspakets

Sie können XMLSpy während der Installation des XMLSpy-Integrationspakets in Eclipse integrieren. Gehen Sie dazu folgendermaßen vor:

1. Starten Sie das XMLSpyIntegrationspaket, um den Installationsassistenten aufzurufen.
2. Führen Sie mit dem Assistenten die ersten Schritte der Installation durch.
3. Aktivieren Sie beim Integrationsschritt die Option *Altova XMLSpy Plug-in mit diesem Assistenten in Eclipse integrieren* und suchen Sie nach dem Ordner, in dem sich die ausführbare Datei von Eclipse befindet (`eclipse.exe`).
4. Klicken Sie auf **Weiter** und schließen Sie die Installation ab.

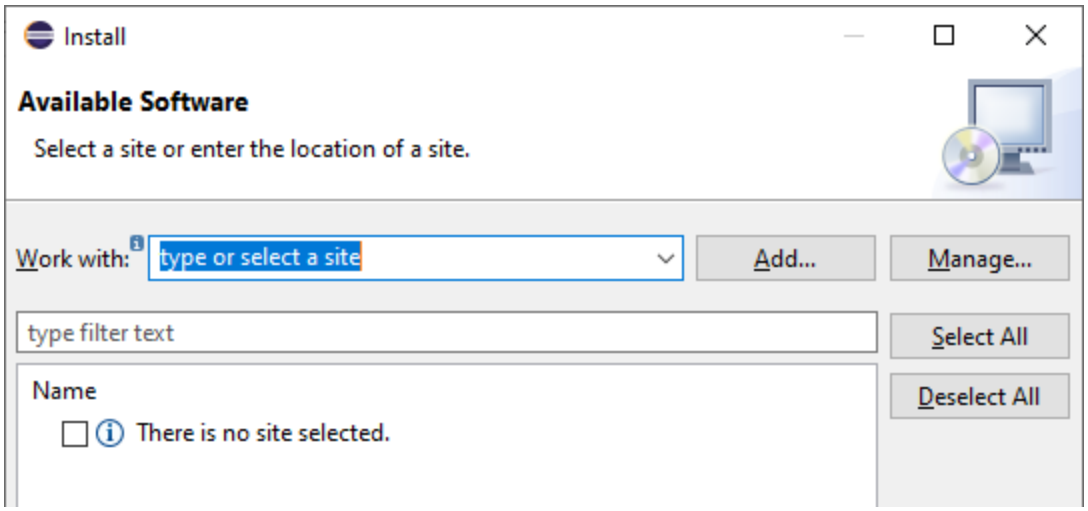
Die XMLSpy-Perspektive und die entsprechenden Menüs stehen daraufhin beim nächsten Start von Eclipse zur Verfügung.

Manuelle Integration von XMLSpy in Eclipse

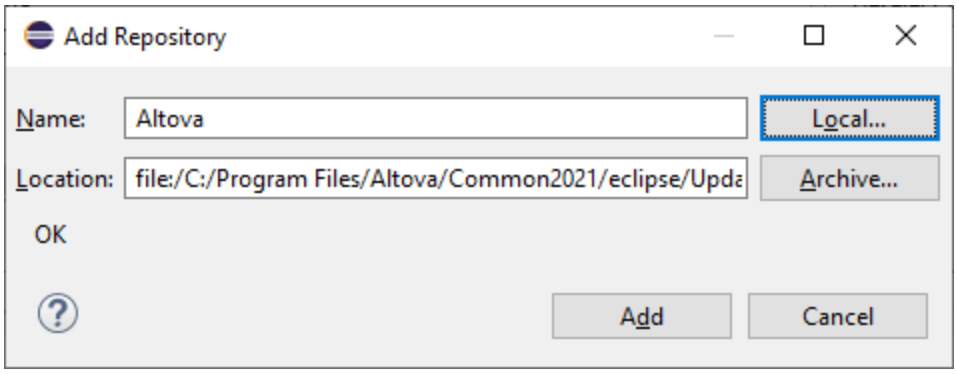
Nach Installation des XMLSpy-Integrationspakets können Sie XMLSpy folgendermaßen manuell in Eclipse integrieren:

1. Klicken Sie in Eclipse auf den Menübefehl **Help | Install new Software**.
2. Klicken Sie im Dialogfeld "Install", das daraufhin angezeigt wird, auf die Schaltfläche **Add**.

© 2016-2022 Altova GmbH



3. Klicken Sie im Dialogfeld "Add Repository" auf **Local**. Navigieren Sie zum Ordner `C:\Programme\Altova\Common2023\eclipse\UpdateSite`, wählen Sie ihn aus und klicken Sie auf OK. Geben Sie einen Namen dafür ein (z.B. "Altova").



4. Wiederholen Sie die Schritte 2 bis 3 und wählen Sie diesmal den Ordner `C:\Programme\Altova\XMLSpy\eclipse\UpdateSite`aus. Geben Sie einen Namen wie z.B. "Altova XMLSpy" ein.
5. Wählen Sie im Installationsdialogfeld *Only Local Sites* aus. Wählen Sie als nächstes den "Altova category"-Ordner aus und klicken Sie auf **Next**.
6. Überprüfen Sie die zu installierenden Objekte und klicken Sie zum Fortfahren auf **Next**.
7. Aktivieren Sie das entsprechende Kontrollkästchen, um die Lizenzvereinbarung zu akzeptieren.
8. Klicken Sie anschließend auf **Finish**, um die Installation fertig zu stellen.

Anmerkung: Falls es Probleme mit dem Plug-in gibt (z.B. fehlende Symbole), starten Sie Eclipse über die Befehlszeile mit dem `-clean` Flag.

26.2 XMLSpy-Perspektive in Eclipse

Eine Perspektive in Eclipse ist eine Ansicht der Benutzeroberfläche, die mit den Funktionalitäten einer bestimmten Applikation konfiguriert ist. Nachdem XMLSpy in Eclipse integriert wurde, steht in Eclipse eine neue Perspektive namens XMLSpy, zur Verfügung. Diese Perspektive ähnelt der Benutzeroberfläche von XMLSpy und enthält eine Reihe ihrer Komponenten.

Wenn eine Datei geöffnet wird, deren Dateityp (z.B. `.xml`) mit XMLSpy verknüpft ist, kann diese Datei in der XMLSpy Perspektive bearbeitet werden. Ebenso kann eine Datei eines anderen Dateityps in einer anderen Perspektive von Eclipse geöffnet werden. Zusätzlich dazu können Sie die Perspektive für jeden Dateityp wechseln (*siehe unten*) und so die Datei in einer anderen Umgebung bearbeiten bzw. verarbeiten.

Perspektiven haben daher zwei wichtige Vorteile:

1. Die Arbeitsumgebung für die aktive Datei kann schnell gewechselt werden und
2. Sie können zwischen Dateien wechseln, ohne eine neue Entwicklungsumgebung öffnen zu müssen (die mit der Datei verknüpfte Umgebung steht in einer Perspektive zur Verfügung).

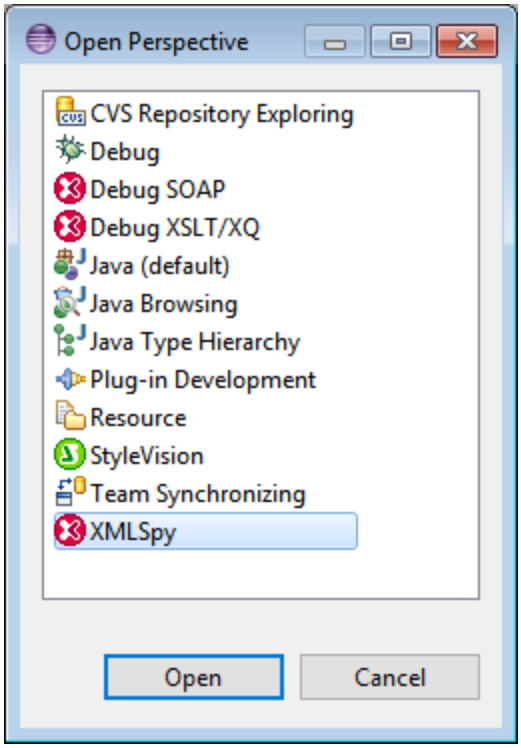
Beim Arbeiten mit der XMLSpy-Perspektive kommen die folgenden unten beschriebenen Verfahren zum Einsatz:

- Wechseln in die XMLSpy Perspektive.
- Festlegen der Einstellungen für die XMLSpy Perspektive.
- Anpassung der XMLSpy Perspektive.

Wechseln in die XMLSpy Perspektive

Wählen Sie in Eclipse den Befehl **Window | Perspective | Open Perspective | Other**. Wählen Sie im Dialogfeld, das jetzt angezeigt wird (*Abbildung unten*),**XMLSpy** aus und klicken Sie auf **Open**.

© 2016-2022 Altova GmbH



Das leere Fenster bzw. das aktive Dokument hat nun die XMLSpy Perspektive. Auf diese Art kann der Benutzer die Perspektive über das Menü wechseln. Um eine Perspektive von einer anderen Perspektive aus schneller aufrufen zu können, kann die gewünschte Perspektive im Untermenü **Open Perspective** oberhalb des Menübefehls **Other** aufgelistet werden; diese Einstellung kann im Anpassungsdialogfeld (*siehe weiter unten*) vorgenommen werden.

Perspektiven können auch gewechselt werden, wenn eine Datei geöffnet oder zur aktiven gemacht wird. Die Perspektive der Applikation, die mit dem Dateityp einer Datei verknüpft ist, wird automatisch geöffnet, wenn diese Datei das erste Mal geöffnet wird. Vor dem Wechseln der Perspektive wird ein Dialogfeld angezeigt, in dem Sie gefragt werden, ob die Standardperspektive automatisch mit diesem Dateityp verknüpft werden soll. Aktivieren Sie die Option *Do Not Ask Again,* wenn die Perspektive mit dem Dateityp verknüpft werden soll, ohne dass Sie jedes Mal gefragt werden, wenn eine Datei dieses Typs geöffnet wird, und klicken Sie anschließend auf **OK**.

Einstellungen für die XMLSpy Perspektive

Wählen Sie den Menübefehl **Window | Preferences**, um das Dialogfeld "Preferences" aufzurufen. Wählen Sie in der Liste der Perspektiven auf der linken Seite XMLSpy aus und wählen Sie dann die gewünschten Einstellungen aus. Klicken Sie anschließend auf **OK**.

Zu den Einstellungen einer Perspektive gehören:

- Der automatische Wechsel in die XMLSpy-Perspektive, wenn eine Datei eines verknüpften Dateityps (*siehe oben*) geöffnet wird
- Optionen zum Inkludieren oder Exkludieren einzelner XMLSpy-Symbolleisten
- Zugriff auf die XMLSpy-Optionen.

Anpassen der XMLSpy-Perspektive

Mit Hilfe der Anpassungsoptionen können Sie festlegen, welche Shortcuts und Befehle in der Perspektive enthalten sein sollen. Um das Dialogfeld "Customize Perspective" aufzurufen, machen Sie die Perspektive zur aktiven und wählen Sie den Befehl **Window | Perspective | Customize Perspective** aus.

- Auf den Registern *Toolbar Visibility* und *Menu Visibility* können Sie festlegen, welche Symbolleisten und Menüs angezeigt werden sollen.
- Auf dem Register *Action Set Availability* können Sie Aktionsgruppen zu den übergeordneten Menüs und zur Symbolleiste hinzufügen. Um eine Aktionsgruppe zu aktivieren, klicken Sie auf das entsprechende Kontrollkästchen.
- Auf dem Register *Shortcuts* des Dialogfelds "Customize Perspective" können Sie die Shortcuts für Untermenüs festlegen. Aktivieren Sie in der Untermenü-Auswahlliste das gewünschte Untermenü. Wählen Sie anschließend eine Shortcut-Kategorie aus und aktivieren Sie die Shortcuts, die in der Perspektive inkludiert werden sollen.

Klicken Sie auf **Apply and Close**, um die Anpassung abzuschließen und damit die Änderungen wirksam werden.

© 2016-2022 Altova GmbH

26.3 XMLSpy-Einstiegspunkte in Eclipse

Zusätzlich zur XMLSpy-Perspektive können die XMLSpy-Funktionalitäten über zwei weitere Einstiegspunkte in Eclipse aufgerufen werden:

- XMLSpy-Menü
- XMLSpy-Symbolleiste

XMLSpy Menü in Eclipse

Das **XMLSpy**-Menü von Eclipse enthält XMLSpy-Befehle, über die wichtige XMLSpy-Funktionalitäten zur Verfügung stehen. Diese Befehle kommen in verschiedenen Menüs der Standalone-Version von XMLSpy vor.

Im unteren Bereich dieses Menüs befinden sich Befehle, mit denen Sie das Design der XMLSpy-Perspektive in Eclipse festlegen können.

XMLSpy-Symbolleiste in Eclipse

Die XMLSpy-Symbolleiste in Eclipse (*Abbildung unten*) enthält zwei Schaltflächen.



Diese Schaltflächen haben die folgenden Funktionen:

- Aufruf der XMLSpy-Hilfe
- Aufruf der XMLSpy-Befehle (anstatt diese über das **XMLSpy**-Menü, *siehe oben*) aufzurufen.

Anmerkung: Symbolleisten-Schaltflächen werden nicht unterstützt. Wenn Sie in XMLSpy eine Symbolleisten-Schaltfläche zur Ausführung eines Befehls oder Skripts definiert haben, steht diese Schaltfläche im Plug-in nicht zur Verfügung.

XMLSpy-Dateiformate und Verhalten von Eclipse-Ansichten

Wenn bestimmte von XMLSpy erkannte Dateitypen in Eclipse aktiv (im Fokus) sind, werden die Ansichten "*Elemente*", "*Attribute*" und "*Entities*" mit dem für dieses Format sinnvollen Namen angezeigt. Wenn z.B. eine `.css`-Datei aktiv ist, wird die "Elemente"-Ansicht mit dem Namen "*CSS-Übersicht*" angezeigt. In der folgenden Tabelle wird aufgelistet, wie sich Ansichtsnamen auf Basis der aktiven Datei ändern:

Wenn dieses Dateiformat aktiv ist	wird die "Elemente"-Ansicht zu...	wird die "Attribute"-Ansicht zu...	wird die "Eigenschaften"-Ansicht zu...
`.css`	CSS-Übersicht	CSS-Eigenschaften	HTML-Elemente
`.xquery, .xq`	XQuery-Schlüsselwörter	XQuery-Variablen	XQuery-Funktionen
`.xsd`	Komponenten	Details	Facets

Wenn Sie eine dieser Ansichten schließen, können Sie diese später mit dem Menübefehl **Fenster | Ansicht anzeigen** wieder aufrufen. Beachten Sie jedoch, dass Ansichten in diesem Menü mit ihren allgemeinen Namen (d.h. *Elemente*, *Attribute* und *Entities*) angezeigt werden. Um z.B. die *CSS-Übersicht* wiederherzustellen, müssen Sie den Menübefehl **Ansicht anzeigen | Elemente** auswählen.

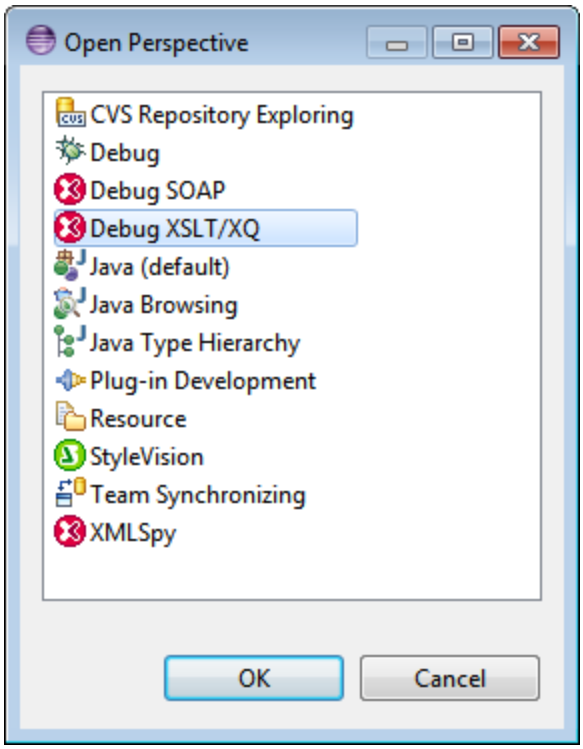
Alternativ dazu können Sie die XMLSpy-Perspektive mit dem Befehl **Fenster| Perspektive zurücksetzen** auf die Standardeinstellung zurücksetzen.

© 2016-2022 Altova GmbH

26.4 Die XMLSpy Debugger-Perspektiven

Es gibt in der Enterprise Edition von XMLSpy zwei Debugger (XSLT/XQuery und SOAP) und in der Professional Edition von XMLSpy einen (XSLT/XQuery). Die in Eclipse verfügbaren Perspektiven für diese Debugger sind abhängig davon, welche XMLSpy Edition installiert ist.

Um in eine Debugger-Perspektive zu wechseln, wählen Sie den Befehl **Window | Open Perspective | Other**. Wählen Sie im daraufhin angezeigten Dialogfeld (*Abbildung unten*) den Debugger aus (z.B., Debug XSLT/XQ) und klicken Sie auf **OK**.



Das leere Fenster bzw. das aktive Dokument hat nun die Perspektive des ausgewählten Debuggers. Auf diese Art kann der Benutzer die Perspektive über das Menü wechseln. Um die Perspektive von einer anderen Perspektive aus schneller aufrufen zu können, kann die gewünschte Perspektive im Untermenü **Open Perspective** oberhalb des Eintrags **Other** aufgelistet werden; diese Einstellung wird im Dialogfeld "Anpassung" vorgenommen.

Eine Beschreibung zur Verwendung der Debugger finden Sie in den entsprechenden Abschnitten dieser Dokumentation: XSLT und XQuery und WSDL und SOAP.

27 Code Generator

XMLSpy enthält einen integrierten Code Generator, der Java-, C++- oder C#-Klassendateien anhand von XML-Schemas generieren kann. Der generierte Code besteht aus stark typisierten Schema Wrapper-Bibliotheken, mit Hilfe derer Sie Software-Applikationen zur Verarbeitung von XML-Daten erstellen können. Mit Hilfe von Schema Wrapper-Bibliotheken können Sie programmatisch mit XML-Daten arbeiten und dabei anhand des Schemas generierte Typen verwenden, ohne sich zu sehr um die zugrunde liegende XML API kümmern zu müssen.

Der Schema Editor von XMLSpy eignet sich gut als Tool zum Entwerfen eines Software-Modells oder Prototyps, da er es gestattet, auf einer hohen Ebene im XML-Schema rasch einen Prototyp einer XML-Applikation zu erzeugen und XML-Applikationen automatisch zu generieren. Änderungen am Inhaltsmodell eines XML-Schemas einer Applikation können einfach durch erneutes Ausführen des Code Generator sofort mit einer Software-Implementierung in Einklang gebracht werden.

Der generierte Code unterstützt die folgenden Operationen:

- Lesen von XML-Dateien in eine speicherresidente Document Object Model (DOM)-Darstellung
- Schreiben von XML-Dateien aus einer DOM-Darstellung zurück in eine Systemdatei
- Konvertieren von Strings in XML-DOM-Strukturen und umgekehrt

Der erzeugte Programmcode wird in den Programmiersprachen C++, Java oder C# ausgegeben.

Zielsprache	C++	C#	Java
Entwicklungsumgebungen	Microsoft Visual Studio 2013, 2015, 2017, 2019, 2022	Microsoft Visual Studio 2013, 2015, 2017, 2019, 2022 Ziel-Frameworks: • .NET Framework • .NET Core 3.1 • .NET 5.0 • .NET 6.0	Java SE JDK 8 oder 11 (einschließlich OpenJDK) Eclipse 4.4 oder höher Apache Ant (build.xml-Datei)
XML DOM Implementierungen	MSXML 6.0 Apache Xerces 3	System.Xml	JAXP

C++

Sie können konfigurieren, ob der generierte C++-Ausgabecode MSXML 6.0 oder Apache Xerces 3 verwenden soll. Bei der C++-Codegenerierung werden in XMLSpy komplette Projekt- (.vcproj) und Solution- (.sln)-Dateien für alle unterstützten Versionen von Visual Studio (siehe Tabelle oben) generiert. Der generierte Code unterstützt optional MFC.

Voraussetzungen:

1. Für die Kompilierung des generierten C++-Codes muss Windows SDK auf Ihrem Rechner installiert sein.

© 2016-2022 Altova GmbH

2. Um Xerces 3 für C++ verwenden zu können, müssen Sie es nach folgender Anleitung installieren und bauen: http://xerces.apache.org/xerces-c/. Dabei muss die Umgebungsvariable XERCES3, die auf das Verzeichnis verweist, in dem Xerces installiert wurde, hinzugefügt werden, z.B: `C:\xerces-c-3.2.2`. Außerdem muss die Umgebungsvariable PATH den Pfad inkludieren, unter dem sich die Xerces-Binärdateien befinden, z.B: `%XERCES3%\bin`.
3. Beim Erstellen von C++-Code für Visual Studio und bei Verwendung einer für Visual C++ vorkompilierten Xerces-Bibliothek muss eine Compiler-Einstellung in allen Projekten der Applikation geändert werden:

   a) Wählen Sie alle Projekte im Solution Explorer aus.
   b) Klicken Sie im Menü **Projekt** auf **Eigenschaften**.
   c) Klicken Sie auf **Konfigurationseigenschaften | C/C++ | Sprache**.
   d) Wählen Sie in der Liste der Konfigurationen *Alle Konfigurationen*.
   e) Ändern Sie **Treat wchar_t as Built-in Type** in **No (/Zc:wchar_t-)**.

C#

Der generierte C#-Code kann von jeder .NET-fähigen Programmiersprache verwendet werden, z.B. VB.NET, Managed C++ oder J#. Projektdateien können für alle unterstützten Versionen von Visual Studio generiert werden (siehe Tabelle oben).

Java

Der generierte Java-Code wird anhand der "Java API for XML Processing" (JAXP) geschrieben und enthält eine Ant build-Datei sowie Projektdateien für die unterstützten Versionen von Java und Eclipse (siehe Tabelle oben).

Generierte Ausgabe

Der Zielordner für den generierten Code enthält alle zum programmatischen Bearbeiten von XML-Dateien erforderlichen Bibliotheken und Dateien. Es sind dies die folgenden:

- Altova Standardbibliotheken
- Schema Wrapper-Bibliotheken
- Eine leere Testapplikation mit Beispielquellcode. Das Applikationsgerüst der Testapplikation ist eine kompilierbare Applikation, die eine leere Example()-Methode aufruft. Sie können in dieser Methode Ihren Testcode hinzufügen, um Ihre neu generierte Bibliothek schnell und einfach testen zu können.

Code Generator-Vorlagen

Der generierte Code unterstützt die Anpassung mittels einer Vorlagensprache[1256] namens SPL (Spy Programming Language). SPL bietet volle Kontrolle über das Mappen von in XML-Schemas integrierten Datentypen auf die primitiven Datentypen einer bestimmten Programmiersprache. Auf diese Art können Sie Ihre eigenen Vorlagen erzeugen, um die Generierung praktisch jedes beliebigen anderen Formats wie EJB's, WSDL-Dateien, SQL Scripts, ASP oder WML-Code zu automatisieren.

27.1 Was ist neu ...

Version 2022 R2

- Unterstützung für Visual Studio 2022.
- Unterstützung für NET 6.0.

Version 2021 R2

- Bei der Generierung von C#-Code kann in den Codegenerierungsoptionen[1254] als Ziel-Framework (zusätzlich zur bestehenden Unterstützung für .NET-Framework-Projekte) .NET Core 3.1, NET 5.0 oder .NET 6.0 ausgewählt werden.

Version 2020 R2

- Für XML-Schema-Wrapper-Bibliotheken generierter Code erlaubt nun mehr Kontrolle über Element-Namespaces und -Präfixe. Zum Deklarieren oder Außerkraftsetzen von Namespaces für ein Element oder zum Anhängen eines Elements mit einem vorangestellten Namespace stehen neue Methoden zur Verfügung. Siehe Beispiel: Bestellung[1197].

Version 2020

- Unterstützung für Visual Studio 2019
- Ende der Unterstützung für Visual Studio 2008

Version 2018

- Unterstützung für Microsoft Visual Studio 2013, 2015, 2017
- Ende der Unterstützung für Visual Studio 2005 und Xerces 2.x

Version 2014

- Kompatibilitätsmodus-Option für die Codegenerierung wurde entfernt

Version 2011

- Enthält Bug Fixes und Verbesserungen

Version 2010 R3

- Unterstützung für Microsoft Visual Studio 2010
- Unterstützung für MSXML 6.0 im generierten C++-Code
- Unterstützung für 64-Bit-Zielkomponenten für C++- und C#-Projekte

Version 2010 R2

- Codegenerierung für C++ für die Linux-Plattform

Version 2010

© 2016-2022 Altova GmbH

- Enumeration Facets aus XML-Schemas stehen nun als Symbolkonstanten in den generierten Klassen zur Verfügung (bei Verwendung von 2007r3 Vorlagen).

Version 2009 sp1

- Wurde um Apache Xerces Version 3.x Unterstützung erweitert. (auch die älteren Versionen ab Xerces 2.6.x werden weiterhin unterstützt.)

Version 2008 R2

- Unterstützung für die Generierung von Visual Studio 2008 Projektdateien für C# und C++.

Version 2008

- Die neuen 2007 SPL-Vorlagen im R3-Stil wurden weiter verbessert:
  - Es können nun einzelne Elemente entfernt werden.
  - Sie haben Zugriff auf Schema-Metadaten (z.B. Elementnamen, Facets, Enumerationen, Occurrence usw.).
  - ComplexType, die durch Erweiterungen abgeleitet werden, werden nun als abgeleitete Klassen generiert.

Version 2007 R3

Der Code Generator wurde in Version 2007 Release 3 zur Vereinfachung des generierten Codes, der Verringerung des Codevolumens und zur Verbesserung der Leistung überarbeitet.

- Behandlung von XML-Dokumenten und Nodes mit expliziter Ownership, um Probleme mit Speicherverwaltung und Multithreading zu vermeiden.
- Neue Syntax zur Vermeidung von Kollisionen
- Neue Datentypen zur einfacheren Verwendung und Leistungsoptimierung (wenn möglich "Native Types", neue Behandlung von Null, ...)
- Attribute werden nicht mehr als Sammlungen generiert.
- Der Inhalt von Simple Elements wird nun aus Konsistenzgründen ebenfalls wie ein spezielles Attribut behandelt.
- Neues internes Objektmodell (wichtig für benutzerdefinierte SPL-Vorlagen)
- Kompatibilitätsmodus zur Generierung von Code im Stile älterer Versionen

27.2 Generieren von Code anhand von XML-Schemas oder DTDs

Sie können mit dem XMLSpy Codegenerator C#-, C++- oder Java-Programmcode anhand von XML-Schemas oder DTDs generieren. Die generierten Schema Wrapper-Bibliotheken können anschließend in Ihre benutzerdefinierte Applikation integriert werden, um XML-Dokumente mittels Programmen lesen oder ändern zu können bzw. in XML-Dokumente schreiben zu können.

Generieren von Programmcode

1. Öffnen Sie das Schema, für das Sie Quellcode generieren möchten.
2. Wählen Sie den Menübefehl **DTD/Schema | Programmcode generieren**.
3. Definieren Sie auf dem Register **Vorlage auswählen** des daraufhin angezeigten Dialogfelds die Codegenerierungsoptionen[1254].
4. Klicken Sie auf **OK.** Darauf hin wird das Dialogfeld *Ordner suchen* angezeigt.
5. Wählen Sie den Zielordner aus und klicken Sie auf **OK**.
6. Sie werden aufgefordert, das neu erstellte Projekt in Microsoft Visual Studio zu öffnen. Klicken Sie auf **Ja**. Wenn Java-Code erzeugt wird, werden Sie aufgefordert, das entsprechende Ausgabeverzeichnis zu öffnen.

Wenn XMLSpy Code anhand eines XML-Schemas oder einer DTD generiert, werden die folgenden Bibliotheken generiert:

C++ oder C#	Java	Verwendungszweck
Altova	`com.altova`	Basisbibliothek, enthält allgemeine Runtime-Unterstützung, ist für alle Schemas identisch.
AltovaXML	`com.altova.xml`	Basisbibliothek, enthält Runtime-Unterstützung für XML, ist für alle Schemas identisch.
[YourSchema]	`com.YourSchema`	Eine Bibliothek, die anhand des Input-Schemas generierte Deklarationen enthält, die denselben Namen wie die Schema-Datei oder DTD hat. Diese Bibliothek ist ein DOM (W3C Document Object Model) Wrapper, der es Ihnen gestattet, XML-Dokumente einfach und sicher zu lesen, zu bearbeiten und zu erstellen. Alle Daten befinden Sie im DOM und es gibt Methoden, um Daten aus dem DOM zu extrahieren und Daten im DOM zu aktualisieren und zu erstellen. Der generierte C++-Code unterstützt entweder Microsoft MSXML oder Apache Xerces 3. Die Syntax zur Verwendung des generierten Codes ist mit Ausnahme geringer Unterschiede (so unterstützt Xerces z.B. mehr überladene Funktionen) bei beiden DOM-Implementierungen fast die gleiche. Dem generierten C#-Code liegt die .NET Standard System.XML-Bibliothek als DOM-Implementierung zugrunde.

© 2016-2022 Altova GmbH

		Dem generierten Java-Code liegt JAXP (Java API for XML Processing) als DOM-Implementierung zugrunde.
[YourSchemaTest]	`com.YourSchemaTest`	Der generierte Code enthält auch ein Test-Applikationsgerüst, das nach Ihrem Schema (z.B. *YourSchemaTest*) benannt ist. Es handelt sich hierbei um eine kompilierbare Applikation, die eine leere Example()-Methode aufruft. Sie können in dieser Methode Ihren Testcode hinzufügen, um Ihre neu generierte Bibliothek schnell und einfach testen zu können.

Beim Erstellen eines Prototyps einer Applikation anhand eines häufig geänderten XML-Schemas müssen Sie eventuell immer wieder Code im selben Verzeichnis generieren, damit die Änderungen am Schema sofort im Code berücksichtigt werden. Beachten Sie, dass die generierte Testapplikation und die Altova-Bibliotheken jedes Mal, wenn Sie Code im selben Zielverzeichnis generieren, überschrieben werden. Fügen Sie daher keinen Code zur generierten Testapplikation hinzu, sondern integrieren Sie stattdessen die Altova-Bibliotheken in Ihr Projekt (siehe Integrieren von Schema Wrapper-Bibliotheken 1170).

Namensgenerierung und Namespaces

XMLSpy generiert für alle deklarierten Elemente oder complexTypes, die einen complexTyp in Ihrem XML-Schema umdefinieren, Klassen, wobei die durch Erweiterungen von complexTyps in Ihrem XML-Schema definierte Klassenableitung beibehalten wird. Bei komplexen Schemas, in die Komponenten aus mehreren Namespaces importiert werden, behält XMLSpy diese Informationen durch Generierung der entsprechenden C#- oder C++-Namespaces oder Java-Pakete bei.

Im Allgemeinen versucht der Code Generator die Namen für generierte Namespaces, Klassen und Member aus dem Original-XML-Schema beizubehalten. Zeichen, die in Identifiern in der Zielsprache ungültig sind, werden durch ein "_" ersetzt. Namen, die mit anderen Namen oder reservierten Wörtern in Konflikt treten würden, werden durch Anhängen einer Nummer eindeutig identifizierbar gemacht. Die Generierung von Namen kann durch Änderung der Standardeinstellungen in der SPL 1256-Vorlage beeinflusst werden.

Die Namespaces aus dem XML-Schema werden in Java in Pakete oder in C#- oder C++-Code in Namespaces konvertiert, wobei das Namespace-Präfix aus dem Schema als Code-Namespace verwendet wird. Die gesamte Bibliothek wird in ein Paket oder einen anhand des Schema-Dateinamens abgeleiteten Namespace eingeschlossen, sodass Sie in einem Programm mehrere generierte Bibliotheken verwenden können, ohne dass es zu Namenskonflikten kommt.

Datentypen

XML-Schema hat ein etwas komplexeres Datentypmodell als Java, C# oder C++. Der Code Generator konvertiert die vordefinierten XML-Schematypen in sprachspezifische primitive Typen oder in mit der Altova-Bibliothek bereitgestellte Klassen. ComplexTypes und abgeleitete Typen, die im Schema definiert sind, werden in der generierten Hierarchie in Klassen konvertiert. Enumeration Facets von simpleTypes werden in Symbolkonstanten konvertiert.

Das Mapping von simpleTypes kann in der SPL-Vorlage konfiguriert werden, siehe SPL-Referenz 1256.

Wenn in Ihren XML-Instanzdateien Schematypen im Zusammenhang mit Uhrzeit und Dauer verwendet werden, so werden diese im Code in native Altova-Klassen konvertiert. Informationen zu den Altova-Bibliotheksklassen finden Sie unter:

- Referenz zu generierten Klassen (C++) 1205
- Referenz zu generierten Klassen (C#) 1221
- Referenz zu generierten Klassen (Java) 1237

Informationen zur Typkonvertierung und anderen Details zu den einzelnen Sprachen finden Sie unter:

- Informationen zu Schema Wrapper-Bibliotheken (C++) 1163
- Informationen zu Schema Wrapper-Bibliotheken (C#) 1166
- Informationen zu Schema Wrapper-Bibliotheken (Java) 1168

Speicherverwaltung

Eine DOM-Struktur besteht aus Nodes, die immer einem bestimmten DOM-Dokument zugeordnet sind - selbst, wenn die Nodes derzeit nicht im Inhalt des Dokuments vorkommen. Bei allen generierten Klassen handelt es sich um Referenzen auf die DOM-Nodes, für die sie stehen, nicht um Werte. Dies bedeutet, dass bei Zuweisung einer Instanz einer generierten Klasse nicht der Wert kopiert wird, sondern nur eine zusätzliche Referenz auf dieselben Daten.

XML Schema-Unterstützung

Die folgenden XML-Schema-Konstrukte werden in Code übersetzt:

a) XML Namespaces

b) Simple Types

- Built-in XML Schema Typen
- Durch Extension abgeleitete Simple Types
- Durch Restriction abgeleitete Simple Types
- Facets
- Enumerations
- Patterns

c) Complex Types:

- Built-in anyType Node
- Benutzerdefinierte Complex Types
- Durch Extension abgeleitete: Auf abgeleitete Klassen gemappte
- Durch Restriction abgeleitete
- Complex Content
- Simple Content
- Mixed Content

Die folgenden komplexen XML-Schema-Funktionen werden in generierten Wrapper-Klassen nicht (oder nicht vollständig) unterstützt:

- Wildcards: `xs:any` und `xs:anyAttribute`
- Inhaltsmodelle (sequence, choice, all): Kompositor der obersten Ebene ist in SPL 1256 vorhanden, wird aber von den generierten Klassen nicht umgesetzt.

© 2016-2022 Altova GmbH

- Standardwerte und festgelegte Werte für Attribute: Diese sind in SPL 1256 verfügbar, werden aber von den generierten Klassen nicht gesetzt oder umgesetzt.
- Attribute `xsi:type`, abstract types: Verwenden Sie die `SetXsiType()`-Methode der generierten Klassen, wenn Sie das xsi:type-Attribut schreiben müssen.
- Union Types: Es werden nicht alle Kombinationen unterstützt.
- Substitution Groups werden zum Teil unterstützt (werden wie "choice" aufgelöst)
- Attribut `nillable="true"` und `xsi:nil`
- Uniqueness Constraints
- Identity Constraints (`key` und `keyref`)

27.2.1 Informationen zu Schema Wrapper-Bibliotheken (C++)

Zeichentypen

Der generierte C++-Code kann mit oder ohne Unicode-Unterstützung generiert werden. Je nachdem, welche Einstellung hier gewählt wurde, werden die Typen string_type und tstring beide als std::string und std::wstring und zwar bestehend aus schmalen oder breiten Zeichen, definiert. Um Unicode-Zeichen in Ihrer XML-Datei verwenden zu können, die nicht durch den aktuellen 8-Bit-Zeichensatz dargestellt werden können, muss die Unicode-Unterstützung aktiviert werden. Seien Sie bei den _T() Makros vorsichtig. Dieses Makro stellt sicher, dass String-Konstanten korrekt gespeichert werden, ob Sie nun für Unicode oder Nicht-Unicode-Programme kompilieren.

Datentypen

Für das Mappen von XML-Schematypen auf C++-Datentypen wird das folgende Standard-Mapping verwendet:

XML Schema	C++	Anmerkungen
`xs:string`	`string_type`	string_type wird definiert als std::string oder std:wstring
`xs:boolean`	`bool`	
`xs:decimal`	`double`	C++ hat keinen Dezimaltyp, daher wird "double" verwendet.
`xs:float, xs:double`	`double`	
`xs:integer`	`__int64`	xs:integer hat keine Bereichsbeschränkung, wird aus Gründen der Effizienz auf __int64 gemappt.
`xs:nonNegativeInteger`	`unsigned __int64`	siehe oben
`xs:int`	`int`	
`xs:unsignedInt`	`unsigned int`	
`xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth`	`altova::DateTime` 1205	
`xs:duration`	`altova::Duration` 1203	

`xs:hexBinary and xs:base64Binary`	`std::vector<unsigned char>`	Die Kodierung und Dekodierung von Binärdateien erfolgt automatisch.
`xs:anySimpleType`	`string_type`	

Alle XML-Schematypen, die nicht in der Liste enthalten sind, sind abgeleitete Typen und werden auf denselben C++-Typ wie der entsprechende Basistyp gemappt.

Generierte Klassen

Für jeden Typ im Schema wird eine Klasse generiert, die ein Member für jedes Attribut und Element des Typs enthält. Die Members erhalten den gleichen Namen wie die Attribute oder Elemente im Originalschema (bei möglichen Konflikten wird eine Zahl angehängt). Für Simple und Mixed Types werden Zuweisungs- und Konvertierungsoperatoren generiert. Für Simple Types mit Enumeration Facets können die Methoden `getEnumerationValue()` und `setEnumerationValue(int)` zusammen mit generierten Konstanten für die einzelnen Enumerationswerte verwendet werden. Des Weiteren können Sie mit Hilfe der Methode `StaticInfo()` Schemainformationen als einen der folgenden Typen abrufen:

`altova::meta::SimpleType` 1214
`altova::meta::ComplexType` 1212

Anhand von Complex Typs generierte Klassen enthalten die Methode `SetXsiType()`, mit Hilfe derer Sie das `xsi:type`-Attribut des Typs definieren können. Diese Methode ist nützlich, wenn Sie XML-Instanzelemente anhand eines Derived Type erstellen möchten.

Zusätzlich zu den Klassen für die im XML-Schema deklarierten Typen, wird eine (unten durch "CDoc" identifizierte) Dokument-Klasse generiert. Sie enthält alle möglichen Root-Elemente als Members sowie verschiedene andere Methoden. Nähere Informationen zur Klasse finden Sie unter `[YourSchema]::[CDoc]` 1215 .

Anmerkung: Der tatsächliche Klassenname ist vom Namen des .xsd-Schemas abhängig.

Für jedes Member-Attribut oder Elemente eines Schematyps wird eine neue Klasse generiert. Nähere Informationen zu solchen Klassen finden Sie unter:

`[YourSchema]::MemberAttribute` 1218
`[YourSchema]::MemberElement` 1219

Anmerkung: Der tatsächliche Klassenname ist vom Namen des Schema-Attributs oder -Elements abhängig.

Siehe auch Beispiel: Verwendung der Schema Wrapper-Bibliotheken 1173.

Behandlung von Fehlern

Fehler werden in Form von Ausnahmeereignissen ausgegeben. Im Namespace `altova` sind die folgenden Ausnahmeereignisklassen definiert:

Klasse	Basisklasse	Beschreibung

© 2016-2022 Altova GmbH

`Error`	`std::logic_error`	Interner Fehler in der Programmlogik (unabhängig von den Eingabedaten)
`Exception`	`std::runtime_error`	Basisklasse für Runtime-Fehler
`InvalidArgumentsException`	`Exception`	Es wurde eine Methode mit ungültigen Argumentwerten aufgerufen.
`ConversionException`	`Exception`	Ausnahmeereignis aufgrund eines Typkonvertierungsfehlers
`StringParseException`	`ConversionException`	Ein Wert im lexikalischen Bereich kann nicht in einen Wertebereich umgewandelt werden.
`ValueNotRepresentableException`	`ConversionException`	Ein Wert im Wertebereich kann nicht in einen lexikalischen Bereich umgewandelt werden.
`OutOfRangeException`	`ConversionException`	Ein Quellwert kann in der Ziel-Domain nicht dargestellt werden.
`InvalidOperationException`	`Exception`	Es wurde versucht, eine Operation auszuführen, die in diesem Kontext nicht gültig ist.
`DataSourceUnavailableException`	`Exception`	Beim Laden einer XML-Instanz ist ein Problem aufgetreten.
`DataTargetUnavailableException`	`Exception`	Beim Speichern einer XML-Instanz ist ein Problem aufgetreten.

Alle Ausnahmeereignisklassen enthalten einen Meldungstext und einen Verweis auf ein mögliches internes Ausnahmeereignis.

Methode	**Verwendungszweck**
`string_type message()`	Gibt eine Textbeschreibung des Ausnahmeereignisses zurück
`std::exception inner()`	Gibt, falls verfügbar, das Ausnahmeereignis zurück, das dieses Ausnahmeereignis verursacht hat, ansonsten wird der Wert NULL zurückgegeben.

Aufrufen von Schemainformationen

Über die generierte Bibliothek können statische Schemainformationen über die folgenden Klassen abgerufen werden. Alle Methoden werden als `const`. deklariert. Die Methoden, die eine der Metadatenklassen zurückgeben, geben ein NULL-Objekt zurück, wenn die entsprechende Eigenschaft nicht existiert.

altova::meta::Attribute 1212
altova::meta::ComplexType 1212
altova::meta::Element 1213
altova::meta::SimpleType 1214

27.2.2 Informationen zu Schema Wrapper-Bibliotheken (C#)

Das Standardmapping von XML-Schematypen auf C#-Datentypen ist:

XML-Schema	C#	Anmerkungen
xs:string	string	
xs:boolean	bool	
xs:decimal	decimal	xs:decimal hat keine Bereichsbeschränkung und Genauigkeit, wird aus Gründen der Effizienz auf decimal gemappt.
xs:float, xs:double	double	
xs:long	long	
xs:unsignedLong	ulong	
xs:int	int	
xs:unsignedInt	uint	
xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth	Altova.Types.DateTime 1221	
xs:duration	Altova.Types.Duration 1226	
xs:hexBinary and xs:base64Binary	byte[]	Die Kodierung und Dekodierung von Binärdateien erfolgt automatisch.
xs:anySimpleType	string	

Alle XML-Schematypen, die nicht in der Liste enthalten sind, sind abgeleitete Typen und werden auf denselben C#-Typ wie der entsprechende Basistyp gemappt.

Generierte Klassen

Für jeden Typ im Schema wird eine Klasse generiert, die ein Member für jedes Attribut und Element des Typs enthält. Die Members erhalten den gleichen Namen wie die Attribute oder Elemente im Originalschema (bei möglichen Konflikten wird eine Zahl angehängt). Für Simple und Mixed Types werden Zuweisungs- und Konvertierungsoperatoren generiert. Für Simple Types mit Enumeration Facets können die Methoden `getEnumerationValue()` und `setEnumerationValue(int)` zusammen mit generierten Konstanten für die einzelnen Enumerationswerte verwendet werden. Des Weiteren können Sie mit Hilfe der Methode `StaticInfo()` Schemainformationen als einen der folgenden Typen abrufen:

Altova.Xml.Meta.SimpleType 1230
Altova.Xml.Meta.ComplexType 1229

© 2016-2022 Altova GmbH

Anhand von Complex Typs generierte Klassen enthalten die Methode `SetXsiType()`, mit Hilfe derer Sie das `xsi:type`-Attribut des Typs definieren können. Diese Methode ist nützlich, wenn Sie XML-Instanzelemente anhand eines Derived Type erstellen möchten.

Zusätzlich zu den Klassen für die im XML-Schema deklarierten Typen, wird eine (unten durch "CDoc" identifizierte) Dokument-Klasse generiert. Sie enthält alle möglichen Root-Elemente als Members sowie verschiedene andere Methoden. Nähere Informationen zur Klasse finden Sie unter [YourSchema].[Doc] 1231.

Anmerkung: Der tatsächliche Klassenname ist vom Namen des .xsd-Schemas abhängig.

Für jedes Member-Attribut oder Elemente eines Schematyps wird eine neue Klasse generiert. Nähere Informationen zu solchen Klassen finden Sie unter:

[YourSchemaType].MemberAttribute 1234
[YourSchemaType].MemberElement 1235

Anmerkung: Der tatsächliche Klassenname ist vom Namen des Schema-Attributs oder -Elements abhängig.

Behandlung von Fehlern

Fehler werden in Form von Ausnahmeereignissen ausgegeben. Im Namespace com.altova sind die folgenden Ausnahmeereignisklassen definiert:

Klasse	Basisklasse	Beschreibung
`ConversionException`	`Exception`	Ausnahmeereignis aufgrund eines Typkonvertierungsfehlers
`StringParseException`	`ConversionException`	Ein Wert im lexikalischen Bereich kann nicht in einen Wertebereich umgewandelt werden.
`DataSourceUnavailableException`	`System.Exception`	Beim Laden einer XML-Instanz ist ein Problem aufgetreten.
`DataTargetUnavailableException`	`System.Exception`	Beim Speichern einer XML-Instanz ist ein Problem aufgetreten.

Zusätzlich dazu werden die folgenden .NET-Ausnahmeereignisse verwendet:

Klasse	Beschreibung
`System.Exception`	Basisklasse für Runtime-Fehler
`System.ArgumentException`	Es wurde eine Methode mit ungültigen Argumentwerten aufgerufen, oder es gab einen Fehler bei einer Typkonvertierung.
`System.FormatException`	Ein Wert im lexikalischen Bereich kann nicht in einen Wertebereich umgewandelt werden.
`System.InvalidCastException`	Ein Wert kann nicht in einen anderen Typ umgewandelt werden.

System.OverflowException	Ein Quellwert kann in der Ziel-Domain nicht dargestellt werden.

Aufrufen von Schemainformationen

Über die generierte Bibliothek können statische Schemainformationen über die folgenden Klassen abgerufen werden.

Altova.Xml.Meta.Attribute 1228
Altova.Xml.Meta.ComplexType 1229
Altova.Xml.Meta.Element 1230
Altova.Xml.Meta.SimpleType 1230

Die Eigenschaften, die eine der Metadatenklassen zurückgeben, geben Null zurück, wenn die entsprechende Eigenschaft nicht existiert.

27.2.3 Informationen zu Schema Wrapper-Bibliotheken (Java)

Für das Mappen von XML-Schematypen auf Java-Datentypen wird das folgende Standard-Mapping verwendet:

XML Schema	Java	Remarks
xs:string	String	
xs:boolean	boolean	
xs:decimal	java.math.BigDecimal	
xs:float, xs:double	double	
xs:integer	java.math.BigInteger	
xs:long	long	
xs:unsignedLong	java.math.BigInteger	Java hat keine unsignierten Typen.
xs:int	int	
xs:unsignedInt	long	Java hat keine unsignierten Typen.
xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth	com.altova.types.DateTime 1237	
xs:duration	com.altova.types.Duration 1241	
xs:hexBinary and xs:base64Binary	byte[]	Die Kodierung und Dekoiderung binärer Daten erfolgt automatisch.
xs:anySimpleType	string	

© 2016-2022 Altova GmbH

Alle XML-Schematypen, die nicht in der Liste enthalten sind, sind abgeleitete Typen und werden auf denselben Java-Typ wie der entsprechende Basistyp gemappt.

Generierte Klassen

Für jeden Typ im Schema wird eine Klasse generiert, die ein Member für jedes Attribut und Element des Typs enthält. Die Members erhalten den gleichen Namen wie die Attribute oder Elemente im Originalschema (bei möglichen Konflikten wird eine Zahl angehängt). Für Simple Types werden Zuweisungs- und Konvertierungsoperatoren generiert. Für Simple Types mit Enumeration Facets können die Methoden `getEnumerationValue()` und `setEnumerationValue(int)` zusammen mit generierten Konstanten für die einzelnen Enumerationswerte verwendet werden. Des Weiteren können Sie mit Hilfe der Methode `StaticInfo()` Schemainformationen als einen der folgenden Typen abrufen:

`com.altova.xml.meta.SimpleType` 1247
`com.altova.xml.meta.ComplexType` 1246

Anhand von Complex Types generierte Klassen enthalten die Methode `SetXsiType()`, mit Hilfe derer Sie das `xsi:type`-Attribut des Typs definieren können. Diese Methode ist nützlich, wenn Sie XML-Instanzelemente anhand eines Derived Type erstellen möchten.

Zusätzlich zu den Klassen für die im XML-Schema deklarierten Typen, wird eine (unten durch "CDoc" identifizierte) Dokument-Klasse generiert. Sie enthält alle möglichen Root-Elemente als Members sowie verschiedene andere Methoden. Nähere Informationen zur Klasse finden Sie unter `com.[YourSchema].[Doc]` 1248.

Anmerkung: Der tatsächliche Klassenname ist vom Namen des .xsd-Schemas abhängig.

Für jedes Member-Attribut oder Elemente eines Schematyps wird eine neue Klasse generiert. Nähere Informationen zu solchen Klassen finden Sie unter:

`com.[YourSchema].[YourSchemaType].MemberAttribute` 1251
`com.[YourSchema].[YourSchemaType]` 1251

Anmerkung: Der tatsächliche Klassenname ist vom Namen des Schema-Attributs oder -Elements abhängig.

Behandlung von Fehlern

Fehler werden in Form von Ausnahmeereignissen ausgegeben. Im Namespace com.altova sind die folgenden Ausnahmeereignisklassen definiert:

Klasse	Basisklasse	Beschreibung
`SourceInstanceUnvailableException`	Exception	Beim Laden einer XML-Instanz ist ein Problem aufgetreten.
`TargetInstanceUnavailableException`	Exception	Beim Speichern einer XML-Instanz ist ein Problem aufgetreten.

Zusätzlich dazu werden die folgenden Java-Ausnahmeereignisse verwendet:

Klasse	Beschreibung
java.lang.Error	Interner Fehler in der Programmlogik (unabhängig von den Eingabedaten)
java.lang.Exception	Basisklasse für Runtime-Fehler
java.lang.IllegalArgumentsException	Es wurde eine Methode mit ungültigen Argumentwerten aufgerufen, oder es gab einen Fehler bei einer Typkonvertierung.
java.lang.ArithmeticException	Ausnahmeereignis aufgrund eines Fehlers bei der Konvertierung eines numerischen Typs.

Aufrufen von Schemainformationen

Über die generierte Bibliothek können statische Schemainformationen über die folgenden Klassen abgerufen werden.

com.altova.xml.meta.Attribute 1245
com.altova.xml.meta.ComplexType 1246
com.altova.xml.meta.Element 1246
com.altova.xml.meta.SimpleType 1247

Die Eigenschaften, die eine der Metadatenklassen zurückgeben, geben Null zurück, wenn die entsprechende Eigenschaft nicht existiert.

27.2.4 Integrieren von Altova-Bibliotheken in vorhandene Projekte

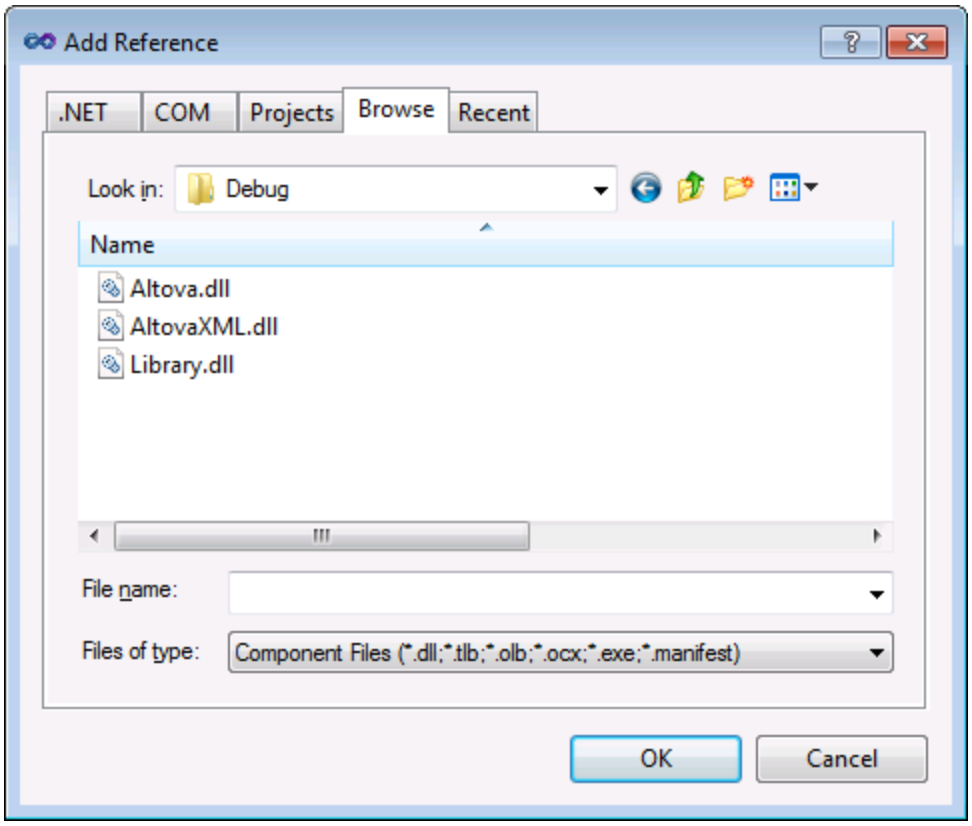
Um die Altova-Bibliotheken in Ihrem benutzerdefinierten Projekt zu verwenden, referenzieren Sie die Bibliotheken von Ihrem Projekt aus (oder inkludieren Sie diese in Ihr Projekt), wie unten für die einzelnen Sprachen gezeigt.

C#

So integrieren Sie die Altova-Bibliotheken in ein bestehendes C#-Projekt:

1. Nachdem XMLSpy Code anhand eines Schemas generiert hat (z.B. **YourSchema.xsd**), erstellen Sie die generierte Lösung **YourSchema.sln** in Visual Studio. Diese Lösung befindet sich in einem Projektordner, der denselben Name wie das Schema hat.
2. Klicken Sie in Visual Studio mit der rechten Maustaste auf Ihr bestehendes Projekt und wählen Sie den Befehl **Add Reference**.
3. Suchen Sie auf dem Register Browse nach den folgenden Bibliotheken: **Altova.dll**, **AltovaXML.dll** und **YourSchema.dll** im Ausgabeverzeichnis **der generierten Projekte (z.B. bin\Debug).**

© 2016-2022 Altova GmbH



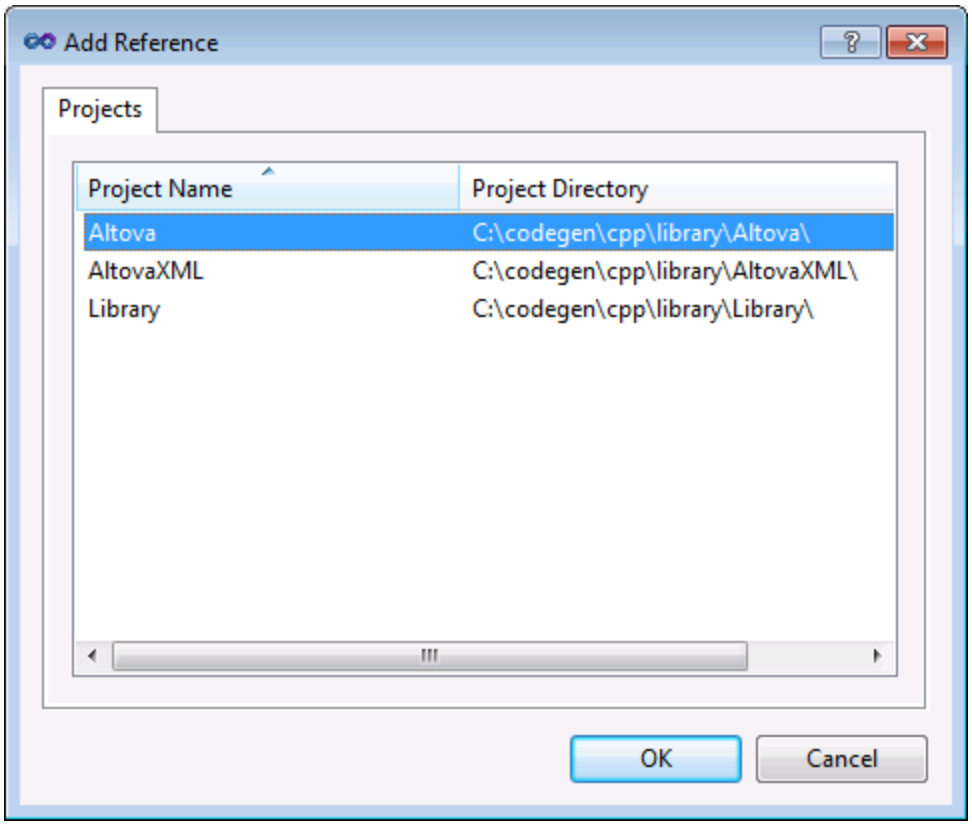
C++

Die einfachste Methode, um die Bibliotheken in ein bestehendes C++-Projekt zu integrieren ist, die generierten Projektdateien zu Ihrer Lösung hinzuzufügen. Angenommen, Sie haben anhand eines Schemas namens **Library.xsd** Code generiert und **c:\codegen\cpp\library** als Zielverzeichnis ausgewählt. In diesem Fall stehen die generierten Bibliotheken unter folgendem Pfad zur Verfügung:

- c:\codegen\cpp\library\Altova.vcxproj
- c:\codegen\cpp\library\AltovaXML\AltovaXML.vcxproj
- c:\codegen\cpp\library\Library.vcxproj

Öffnen Sie zuerst die generierte Lösung **c:\codegen\cpp\library\Library.sln** und bauen Sie sie in Visual Studio.

Öffnen Sie anschließend Ihre bestehende Visual Studio-Lösung (in diesem Beispiel in Visual Studio 2010), klicken Sie mit der rechten Maustaste darauf und wählen Sie **Add | Existing Project** und fügen Sie die oben aufgelisteten Projektdateien einzeln nacheinander hinzu. Haben Sie einen Moment Geduld, während Visual Studio die Dateien parst. Klicken Sie anschließend mit der rechten Maustaste auf Ihr Projekt und wählen Sie **Properties**. Wählen Sie im Dialogfeld "Property Pages" die Option **Common Properties | Framework and References** und klicken Sie anschließend auf **Add New Reference**. Wählen Sie anschließend jedes einzelne der folgenden Projekte aus und fügen Sie es hinzu: *Altova*, *AltovaXML* und *Library*.



Informationen zur Verwendung von Funktionalitäten aus einer benutzerdefinierten Bibliothek finden Sie außerdem in der MSDN-Dokumentation zu Ihrer jeweiligen Visual Studio-Version, z.B.:

- Wenn Sie statische Bibliotheken generieren möchten, siehe https://msdn.microsoft.com/en-us/library/ms235627(v=vs.100).aspx
- Wenn Sie dynamische Bibliotheken generieren möchten: siehe https://msdn.microsoft.com/en-us/library/ms235636(v=vs.100).aspx

Die Option zum Generieren statischer oder dynamischer Bibliotheken finden Sie in den Code Generator-Optionen (siehe Code Generator-Optionen 1254).

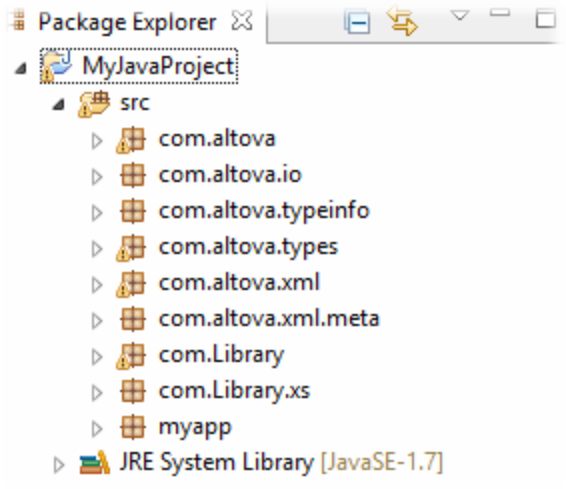
Java

Eine der Möglichkeiten Altova-Pakete in Ihr Java-Projekt zu integrieren, ist, das **com**-Verzeichnis in das Verzeichnis zu kopieren, das die Quellpakete Ihres Java-Projekts enthält (z.B. **C:\Workspace\MyJavaProject\src**). Angenommen, Sie haben Code in **c:\codegen\java\library** generiert. In diesem Fall befinden sich die generierten Altova-Klassen unter **c:\codegen\java\library\com**.

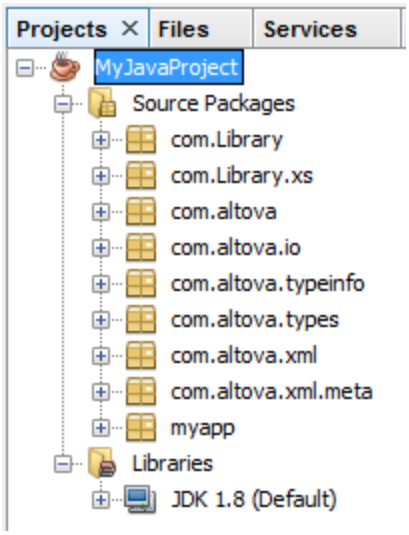
Aktualisieren Sie das Projekt, nachdem Sie die Bibliotheken kopiert haben. Um das Projekt in Eclipse zu aktualisieren, wählen Sie es im Paket-Explorer aus und drücken Sie **F5. Um das Projekt in NetBeans IDE 8.0 zu aktualisieren, wählen Sie den Menübefehl Source | Scan for External Changes.**

Nach dem Kopiervorgang stehen die Altova-Pakete (bei Eclipse) im Paket-Explorer oder (bei NetBeans) im Projektbereich unter "Source Packages" zur Verfügung.

 © 2016-2022 Altova GmbH



Altova-Pakete in Eclipse 4.4



Altova-Pakete in NetBeans IDE 8.0.2

27.2.5 Beispiel: Verwendung der Schema Wrapper-Bibliotheken

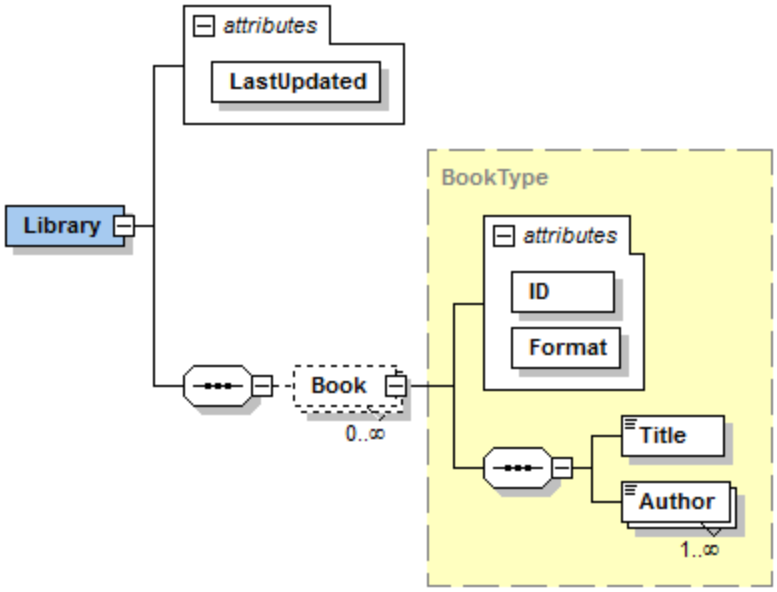
Anhand dieses Beispiels wird gezeigt, wie Sie mit Hilfe der generierten Schema Wrapper-Bibliotheken schemakonforme XML-Dokumente programmatisch schreiben oder lesen. Werfen Sie zuerst einen Blick auf die Struktur des inkludierten Beispielschemas, bevor Sie den Beispielcode verwenden.

In dem in diesem Beispiel verwendeten Schema ist eine Bibliothek mit Büchern beschrieben. Unten sehen Sie die vollständige Definition des Schemas. Speichern Sie dieses Codefragment als **`Library.xsd`**, wenn Sie dieselben Ergebnisse wie in diesem Beispiel erhalten möchten. Sie benötigen dieses Schema zur Generierung der in diesem Beispiel verwendeten Codebibliotheken.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="http://www.nanonull.com/LibrarySample"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.nanonull.com/LibrarySample" elementFormDefault="qualified"
attributeFormDefault="unqualified">
   <xs:element name="Library">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="Book" type="BookType" minOccurs="0" maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="LastUpdated" type="xs:dateTime"/>
      </xs:complexType>
   </xs:element>
   <xs:complexType name="BookType">
      <xs:sequence>
         <xs:element name="Title" type="xs:string"/>
         <xs:element name="Author" type="xs:string" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="ID" type="xs:integer" use="required"/>
      <xs:attribute name="Format" type="BookFormatType" use="required"/>
   </xs:complexType>
   <xs:complexType name="DictionaryType">
      <xs:complexContent>
         <xs:extension base="BookType">
            <xs:sequence>
               <xs:element name="FromLang" type="xs:string"/>
               <xs:element name="ToLang" type="xs:string"/>
            </xs:sequence>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
   <xs:simpleType name="BookFormatType">
      <xs:restriction base="xs:string">
         <xs:enumeration value="Hardcover"/>
         <xs:enumeration value="Paperback"/>
         <xs:enumeration value="Audiobook"/>
         <xs:enumeration value="E-book"/>
      </xs:restriction>
   </xs:simpleType>
</xs:schema>
```

Library ist ein Root-Element vom Typ `complexType`. Grafisch kann es in der Schema-Ansicht von XMLSpy folgendermaßen dargestellt werden:

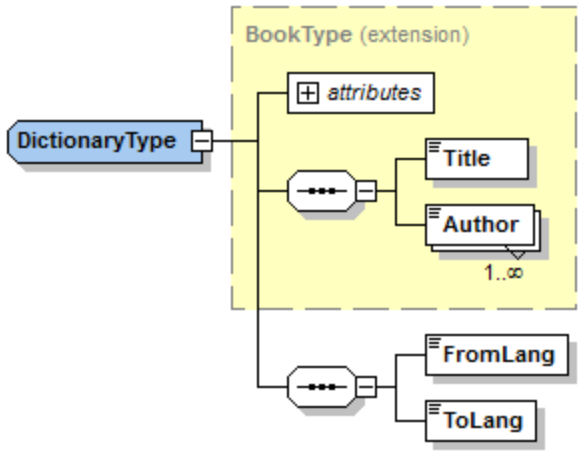
© 2016-2022 Altova GmbH



Wie oben gezeigt, hat die Bibliothek ein Attribut **LastUpdated** (definiert als `xs:dateTime`) und enthält eine Sequenz von Büchern. Jedes Buch hat den Typ `xs:complexType` und verfügt über zwei Attribute: eine **ID** (definiert als `xs:integer`) und ein **Format**. Das Format eines Buchs kann hardcover, paperback, audiobook oder e-book sein. Im Schema ist **Format** als `xs:simpleType` definiert, für den eine Enumeration der oben angeführten Werte verwendet wird..

Außerdem hat jedes Buch ein **Title**-Element (definiert als `xs:string`) sowie ein oder mehrere **Author**-Elemente (definiert als `xs:string`).

Die Bibliothek kann auch Bücher enthalten, die Wörterbücher sind. Wörterbücher haben den Typ `DictionaryType`, der durch Erweiterung vom Typ `BookType` abgeleitet ist. Ein Wörterbuch erbt also alle Attribute und Elemente eines Buchs plus zwei weitere Elemente: **FromLang** und **ToLang**, wie unten gezeigt.



In den Elementen **FromLang** und **ToLang** ist die Quell- und Zielsprache des Wörterbuchs gespeichert.

Eine XML-Instanzdatei, die gemäß dem obigen Schema gültig ist, könnte also wie im Codefragment unten aussehen (Vorausgesetzt sie befindet sich im selben Verzeichnis wie die Schemadatei):

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample"
xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd" LastUpdated="2016-
02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/LibrarySample"
xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

In den nächsten Kapiteln wird beschrieben, wie eine solche Datei programmatisch gelesen bzw. in diese Datei programmatisch geschrieben werden kann. Generieren Sie zuerst den Schema Wrapper-Code anhand des obigen Schemas. Gehen Sie dabei vor, wie unter Generieren von Code anhand von XML-Schemas oder DTDs 1160 beschrieben.

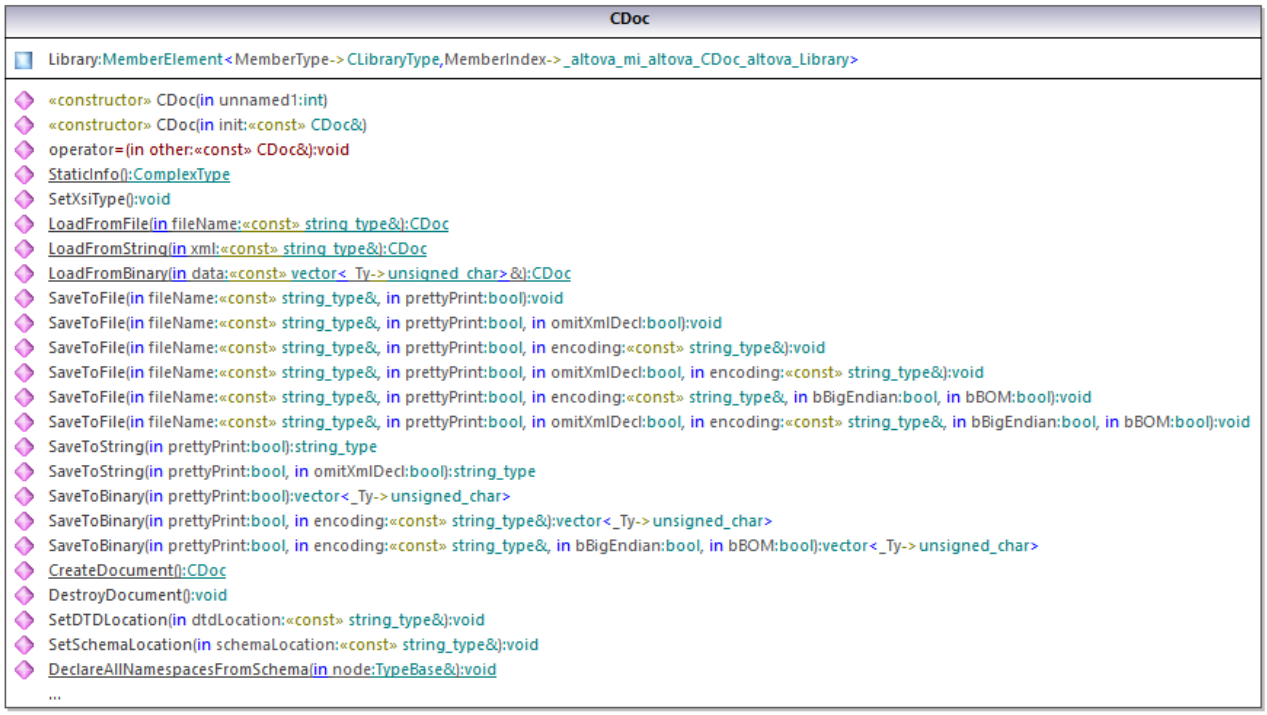
27.2.5.1 Lesen und Schreiben von XML-Dokumenten (C++)

Nachdem Sie anhand des Schemas "Library" (siehe Beispielschema 1173) Code generiert haben, wird eine C#-Testapplikation sowie eine Reihe von unterstützenden Altova-Bibliotheken generiert.

Informationen zu den generierten C++-Bibliotheken

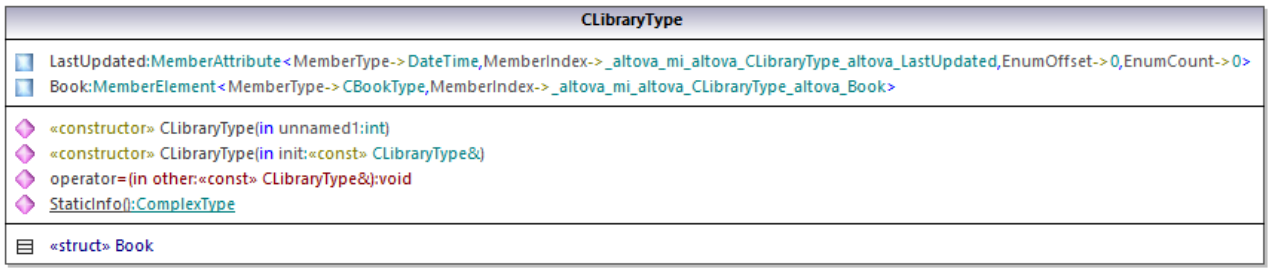
Die zentrale Klasse des generierten Codes ist die Klasse `CDoc`. Sie repräsentiert das XML-Dokument. Eine solche Klasse wird für jedes Schema generiert. Ihr Name hängt vom Namen der Schemadatei ab. Wie im Diagramm gezeigt, bietet diese Klasse Methoden zum Laden von Dokumenten aus Dateien, Binär-Streams oder Strings (oder zum Speichern von Dokumenten in Dateien, Streams, Strings). In der Referenz zur Klasse (`[YourSchema]::[CDoc]` 1215) finden Sie eine Beschreibung aller von dieser Klasse bereitgestellten Members.

 © 2016-2022 Altova GmbH

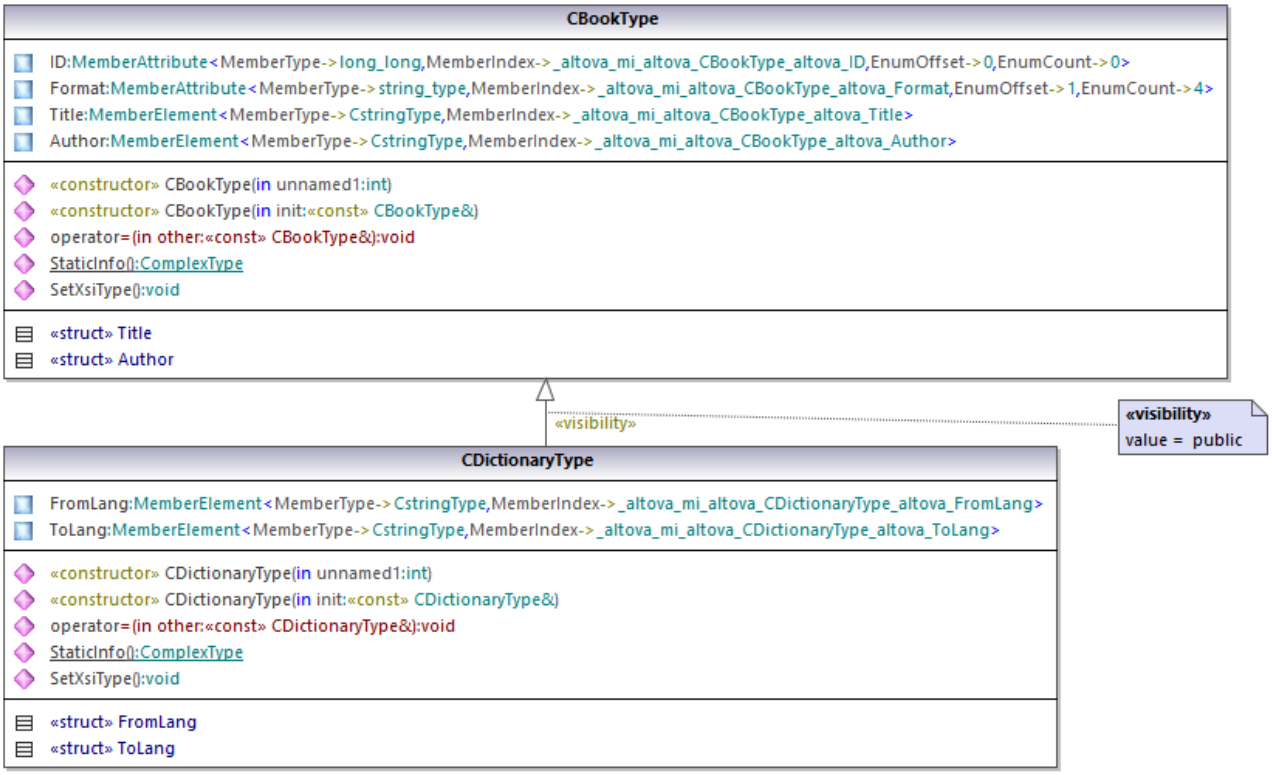


Das Feld `Library` der Klasse `CDoc` stellt die eigentliche Root des Dokuments dar. **Library** ist ein Element in der XML-Datei, daher hat es im C++-Code eine Vorlagenklasse als Typ (`MemberElement`). Die Vorlagenklasse stellt Methoden und Eigenschaften für die Interaktion mit dem Element **Library** bereit. Im Allgemeinen erhält jedes Attribut und jedes Element eines bestimmten Typs im Schema im generierten Code die Vorlagenklasse `MemberAttribute` bzw. `MemberElement`. Nähere Informationen dazu finden Sie in der Referenz zur Klasse unter `[YourSchema]::MemberAttribute`[1218] und `[YourSchema]::MemberElement`[1219].

Die Klasse `CLibraryType` wird anhand des complexType **LibraryType** im Schema generiert. Beachten Sie, dass die Klasse `CLibraryType` ein Feld `Book` und ein Feld `LastUpdated` enthält. Gemäß der oben erwähnten Logik entsprechen diese dem Element **Book** und dem Attribut **LastUpdated** im Schema. Sie ermöglichen die programmatische Bearbeitung (Anhängen, Entfernen, usw.) von Elementen und Attributen im XML-Instanzdokument.



Der Typ `DictionaryType` ist ein complexType, der im Schema von **BookType** abgeleitet wurde, daher wird diese Beziehung auch in den generierten Klassen übernommen. Im Diagramm sehen Sie, dass die Klasse `CDictionaryType` die Klasse `CBookType` erbt.



Wenn in Ihrem XML-Schema simpleTypes als Enumerationen definiert sind, so stehen die enumerierten Werte im generierten Code als `Enum`-Werte zur Verfügung. In dem in diesem Beispiel verwendeten Schema gibt es die Buchformate hardcover, paperback, e-book, usw. Im generierten Code stehen diese Werte folglich in Form einer Enum, d.h. als Member der Klasse `CBookFormatType`, zur Verfügung.

Schreiben eines XML-Dokuments

1. Öffnen Sie die anhand des Schemas "Library" generierte Lösung **LibraryTest.sln** in Visual Studio.

Beim Erstellen eines Prototyps einer Applikation anhand eines häufig geänderten XML-Schemas müssen Sie eventuell immer wieder Code im selben Verzeichnis generieren, damit die Änderungen am Schema sofort im Code berücksichtigt werden. Beachten Sie, dass die generierte Testapplikation und die Altova-Bibliotheken jedes Mal, wenn Sie Code im selben Zielverzeichnis generieren, überschrieben werden. Fügen Sie daher keinen Code zur generierten Testapplikation hinzu, sondern integrieren Sie stattdessen die Altova-Bibliotheken in Ihr Projekt (siehe Integrieren von Schema Wrapper-Bibliotheken 1170).

2. Öffnen Sie die Datei **LibraryTest.cpp** im Solution Explorer und bearbeiten Sie die Methode `Example()` wie unten gezeigt.

```
#include <ctime> // required to get current time
using namespace Doc; // required to work with Altova libraries

void Example()
```

 © 2016-2022 Altova GmbH

```
{
   // Create a new, empty XML document
   CDoc libDoc = CDoc::CreateDocument();

   // Create the root element <Library> and add it to the document
   CLibraryType lib = libDoc.Library.append();

   // Get current time and set the "LastUpdated" attribute using Altova classes
   time_t t = time(NULL);
   struct tm * now = localtime( & t );
   altova::DateTime dt = altova::DateTime(now->tm_year + 1900, now->tm_mon + 1, now-
>tm_mday, now->tm_hour, now->tm_min, now->tm_sec);
   lib.LastUpdated = dt;

   // Create a new <Book> and add it to the library
   CBookType book = lib.Book.append();

   // Set the "ID" attribute of the book
   book.ID = 1;

   // Set the "Format" attribute of the <Book> using an enumeration constant
   book.Format.SetEnumerationValue( CBookFormatType::k_Paperback );

   // Add the <Title> and <Author> elements, and set values
   book.Title.append() = _T("The XML Spy Handbook");
   book.Author.append() = _T("Altova");

   // Append a dictionary (book of derived type) and populate its attributes and elements
   CDictionaryType dictionary = CDictionaryType(lib.Book.append().GetNode());
   dictionary.ID = 2;
   dictionary.Format.SetEnumerationValue( CBookFormatType::k_E_book);
   dictionary.Title.append() = _T("English-German Dictionary");
   dictionary.Author.append() = _T("John Doe");
   dictionary.FromLang.append() = _T("English");
   dictionary.ToLang.append() = _T("German");

   // Since dictionary a derived type, set the xsi:type attribute of the book element
   dictionary.SetXsiType();

   // Optionally, set the schema location
   libDoc.SetSchemaLocation(_T("Library.xsd"));

   // Save the XML document to a file with default encoding (UTF-8),
   // "true" causes the file to be pretty-printed.
   libDoc.SaveToFile(_T("GeneratedLibrary.xml"), true);

   // Destroy the document
   libDoc.DestroyDocument();
}
```

3. Drücken Sie **F5,** um den Debugger zu starten. Wenn der Code erfolgreich ausgeführt wurde, wird im Lösungsausgabeverzeichnis die Datei **GeneratedLibrary.xml** erstellt.

Lesen eines XML-Dokuments

1. Öffnen Sie die Lösung **LibraryTest.sln** in Visual Studio.
2. Speichern Sie den unten gezeigten Code unter dem Namen **Library1.xml** in einem Verzeichnis, das vom Programmcode gelesen werden kann (z.B im selben Verzeichnis wie **LibraryTest.sln**).

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample"
xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd" LastUpdated="2016-
02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/LibrarySample"
xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

3. Öffnen Sie die Datei **LibraryTest.cpp** im Solution Explorer und bearbeiten Sie die Methode `Example()` wie unten gezeigt.

```
using namespace Doc;
void Example()
{
   // Load XML document
   CDoc libDoc = CDoc::LoadFromFile(_T("Library1.xml"));

   // Get the first (and only) root element <Library>
   CLibraryType lib = libDoc.Library.first();

   // Check whether an element exists:
   if (!lib.Book.exists())
   {
      tcout << "This library is empty." << std::endl;
      return;
   }

   // iteration: for each <Book>...
   for (Iterator<CBookType> itBook = lib.Book.all(); itBook; ++itBook)
   {
      // output values of ISBN attribute and (first and only) title element
      tcout << "ID: " << itBook->ID << std::endl;
      tcout << "Title: " << tstring(itBook->Title.first()) << std::endl;

      // read and compare an enumeration value
```


© 2016-2022 Altova GmbH

```
      if (itBook->Format.GetEnumerationValue() == CBookFormatType::k_Paperback)
         tcout << "This is a paperback book." << std::endl;

      // for each <Author>...
      for (CBookType::Author::iterator itAuthor = itBook->Author.all(); itAuthor; +
+itAuthor)
         tcout << "Author: " << tstring(itAuthor) << std::endl;

      // alternative: use count and index
      for (unsigned int j = 0; j < itBook->Author.count(); ++j)
         tcout << "Author: " << tstring(itBook->Author[j]) << std::endl;
   }

   // Destroy the document
   libDoc.DestroyDocument();
}
```

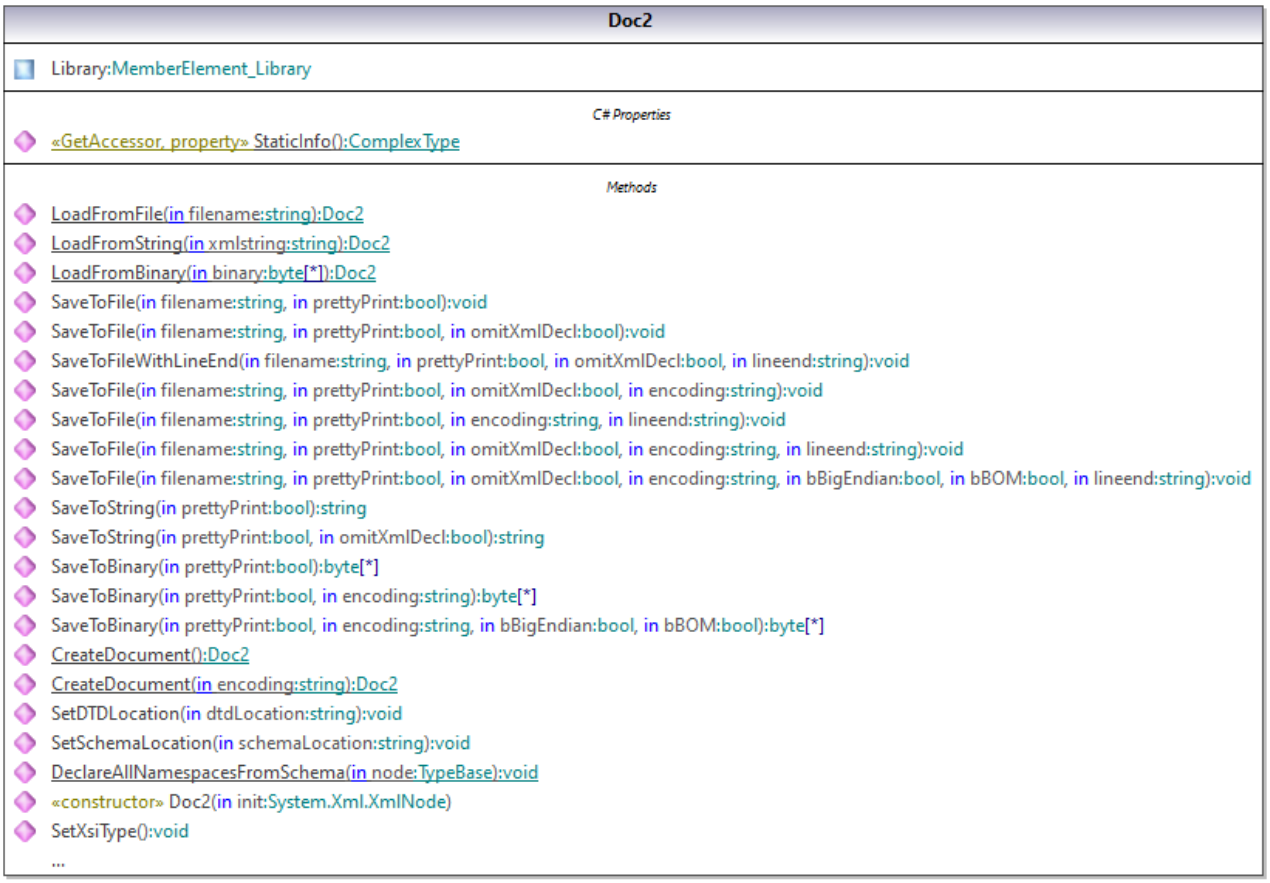
4. Drücken Sie **F5,** um den Debugger zu starten.

27.2.5.2 Lesen und Schreiben von XML-Dokumenten (C#)

Nachdem Sie anhand des Schemas "Library" (siehe Beispielschema[1173]) Code generiert haben, wird eine C#-Testapplikation sowie eine Reihe von unterstützenden Altova-Bibliotheken generiert.

Informationen zu den generierten C#-Bibliotheken

Die zentrale Klasse des generierten Codes ist die Klasse `Doc2`. Sie repräsentiert das XML-Dokument. Eine solche Klasse wird für jedes Schema generiert. Ihr Name hängt vom Namen der Schemadatei ab. Beachten Sie, dass diese Klasse den Namen `Doc2` erhält, um einen möglichen Konflikt mit dem Namespace-Namen zu vermeiden. Wie im Diagramm gezeigt, bietet diese Klasse Methoden zum Laden von Dokumenten aus Dateien, Binär-Streams oder Strings (oder zum Speichern von Dokumenten in Dateien, Streams, Strings). In der Referenz zur Klasse ([YourSchema].[Doc][1231]) finden Sie eine Beschreibung dieser Klasse.

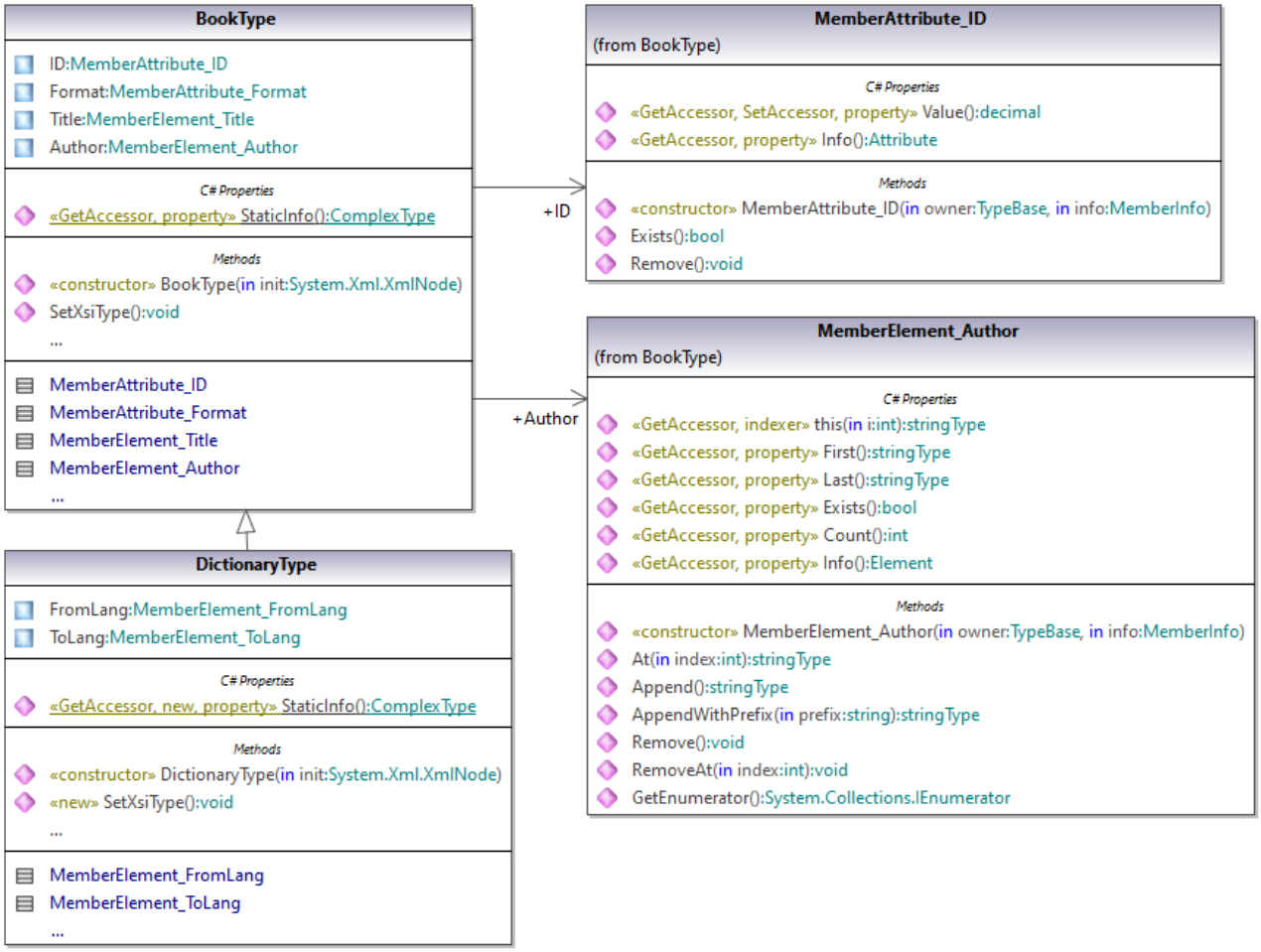


Der Member `Library` der Klasse `Doc2` stellt die eigentliche Root des Dokuments dar.

Gemäß den im Kapitel Informationen zu Schema Wrapper-Bibliotheken (C#) 1166 angeführten Codegenerierungsregeln werden für jedes Attribut und jedes Element eines Typs Member-Klassen generiert. Der Name solcher Member-Klassen erhält im generierten Code das Präfix `MemberAttribute_` bzw. `MemberElement_`. Beispiele für solche Klassen sind `MemberAttribute_ID` und `MemberElement_Author`, die anhand des Elements **Author** bzw. des Attributs **ID** eines Buchs generiert wurden (Im Diagramm unten sind dies unter `BookType` verschachtelte Klassen). Sie ermöglichen die programmatische Bearbeitung (z.B. Anhängen, Entfernen, Wert definieren, usw.) der entsprechenden Elemente und Attribute im XML-Instanzdokument. Nähere Informationen dazu finden Sie in der Referenz zur Klasse unter `[YourSchemaType].MemberAttribute` 1234 und `[YourSchemaType].MemberElement` 1235.

Der Typ **DictionaryType** ist ein complexType, der im Schema von **BookType** abgeleitet wurde, daher wird diese Beziehung auch in den generierten Klassen übernommen. Im Diagramm unten sehen Sie, dass die Klasse `DictionaryType` die Klasse `BookType` erbt.

 © 2016-2022 Altova GmbH



Wenn in Ihrem XML-Schema simpleTypes als Enumerationen definiert sind, so stehen die enumerierten Werte im generierten Code als `Enum`-Werte zur Verfügung. In dem in diesem Beispiel verwendeten Schema gibt es die Buchformate hardcover, paperback, e-book, usw. Im generierten Code stehen diese Werte folglich in Form einer Enum, d.h. als Member der Klasse CBookFormatType, `zur Verfügung`.

Schreiben eines XML-Dokuments

1. Öffnen Sie die anhand des Schemas "Library" generierte Lösung **LibraryTest.sln** in Visual Studio.

Beim Erstellen eines Prototyps einer Applikation anhand eines häufig geänderten XML-Schemas müssen Sie eventuell immer wieder Code im selben Verzeichnis generieren, damit die Änderungen am Schema sofort im Code berücksichtigt werden. Beachten Sie, dass die generierte Testapplikation und die Altova-Bibliotheken jedes Mal, wenn Sie Code im selben Zielverzeichnis generieren, überschrieben werden. Fügen Sie daher keinen Code zur generierten Testapplikation hinzu, sondern integrieren Sie stattdessen die Altova-Bibliotheken in Ihr Projekt (siehe Integrieren von Schema Wrapper-Bibliotheken 1170).

2. Öffnen Sie die Datei **LibraryTest.cs** im Solution Explorer und bearbeiten Sie die Methode `Example()` wie unten gezeigt.

```
protected static void Example()
{
    // Create a new XML document
    Doc2 doc = Doc2.CreateDocument();
    // Append the root element
    LibraryType root = doc.Library.Append();

    // Create the generation date using Altova DateTime class
    Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);
    // Append the date to the root
    root.LastUpdated.Value = dt;

    // Add a new book
    BookType book = root.Book.Append();
    // Set the value of the ID attribute
    book.ID.Value = 1;
    // Set the format of the book (enumeration)
    book.Format.EnumerationValue = BookFormatType.EnumValues.eHardcover;
    // Set the Title and Author elements
    book.Title.Append().Value = "The XMLSpy Handbook";
    book.Author.Append().Value = "Altova";

    // Append a dictionary (book of derived type) and populate its attributes and
elements
    DictionaryType dictionary = new DictionaryType(root.Book.Append().Node);
    dictionary.ID.Value = 2;
    dictionary.Title.Append().Value = "English-German Dictionary";
    dictionary.Format.EnumerationValue = BookFormatType.EnumValues.eE_book;
    dictionary.Author.Append().Value = "John Doe";
    dictionary.FromLang.Append().Value = "English";
    dictionary.ToLang.Append().Value = "German";
    // Since it's a derived type, make sure to set the xsi:type attribute of the
book element
    dictionary.SetXsiType();

    // Optionally, set the schema location (adjust the path if
    // your schema is not in the same folder as the generated instance file)
    doc.SetSchemaLocation("Library.xsd");

    // Save the XML document with the "pretty print" option enabled
    doc.SaveToFile("GeneratedLibrary.xml", true);
}
```

3. Drücken Sie **F5,** um den Debugger zu starten. Wenn der Code erfolgreich ausgeführt wurde, wird im Lösungsausgabeverzeichnis (normalerweise **bin/Debug**) die Datei **GeneratedLibrary.xml** erstellt.

Lesen eines XML-Dokuments

1. Öffnen Sie die Lösung **LibraryTest.sln** in Visual Studio.
2. Speichern Sie den unten gezeigten Code unter dem Namen **Library.xml** im Ausgabeverzeichnis (standardmäßig **bin/Debug**). Dies ist die Datei, die vom Programmcode gelesen wird.

© 2016-2022 Altova GmbH

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample"
xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd" LastUpdated="2016-
02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/LibrarySample"
xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

3. Öffnen Sie die Datei **LibraryTest.cs** im Solution Explorer und bearbeiten Sie die Methode `Example()` wie unten gezeigt.

```
protected static void Example()
{
    // Load the XML file
    Doc2 doc = Doc2.LoadFromFile("Library.xml");
    // Get the root element
    LibraryType root = doc.Library.First;

    // Read the library generation date
    Altova.Types.DateTime dt = root.LastUpdated.Value;
    string dt_as_string = dt.ToString(DateTimeFormat.W3_dateTime);
    Console.WriteLine("The library generation date is: " + dt_as_string);

    // Iteration: for each <Book>...
    foreach (BookType book in root.Book)
    {
      // Output values of ID attribute and (first and only) title element
      Console.WriteLine("ID:    " + book.ID.Value);
      Console.WriteLine("Title: " + book.Title.First.Value);

      // Read and compare an enumeration value
      if (book.Format.EnumerationValue == BookFormatType.EnumValues.ePaperback)
         Console.WriteLine("This is a paperback book.");

      // Iteration: for each <Author>
      foreach (xs.stringType author in book.Author)
         Console.WriteLine("Author: " + author.Value);

      // Determine if this book is of derived type
      if (book.Node.Attributes.GetNamedItem("xsi:type") != null)
      {
         // Find the value of the xsi:type attribute
```

```
         string xsiTypeValue =
book.Node.Attributes.GetNamedItem("xsi:type").Value;
         // Get the namespace URI and the lookup prefix of this namespace
         string namespaceUri = book.Node.NamespaceURI;
         string prefix = book.Node.GetPrefixOfNamespace(namespaceUri);

         // if this book has DictionaryType
         if (namespaceUri == "http://www.nanonull.com/LibrarySample" &&
xsiTypeValue.Equals(prefix + ":DictionaryType"))
         {
            // output additional fields
            DictionaryType dictionary = new DictionaryType(book.Node);
            Console.WriteLine("Language from: " +
dictionary.FromLang.First.Value);
            Console.WriteLine("Language to: " + dictionary.ToLang.First.Value);
         }
         else
         {
            throw new Exception("Unexpected book type");
         }
      }
   }

   Console.ReadLine();
}
```

4. Drücken Sie **F5,** um den Debugger zu starten. Wenn der Code erfolgreich ausgeführt wurde, wird **Library.xml** vom Programmcode gelesen und ihr Inhalt wird als Konsolenausgabe angezeigt.

Lesen und Schreiben von Elemente und Attributen

Die Werte von Attributen und Elementen können über die Eigenschaft `Value` der generierten Member-Element- bzw. Attribut-Klasse gelesen werden, z.B.:

```
// output values of ID attribute and (first and only) title element
System.out.println("ID:    " + book.ID.Value);
Console.WriteLine("Title: " + book.Title.First.Value);
```

Um in diesem konkreten Beispiel den Wert des Elements **Title** abzurufen, haben wir auch die Methode `First()` verwendet, da es sich hier um des erste (und einzige) **Title**-Element eines Buchs handelt. In Fällen, in denen ein bestimmtes Element nach dem Index aus einer Liste gewählt werden soll, verwenden Sie die Methode `At()`.

Die für die einzelnen Member-Elemente eines Typs generierte Klasse implementiert die Standardschnittstelle `System.Collections.IEnumerable`. Auf diese Art können mehrere Elemente desselben Typs in einer Schleife verarbeitet werden. In diesem konkreten Beispiel können Sie alle Bücher eines Library-Objekts folgendermaßen in einer Schleife verarbeiten:

```
// Iteration: for each <Book>...
foreach (BookType book in root.Book)
{
```

 © 2016-2022 Altova GmbH

```
    // your code here...
}
```

Mit Hilfe der Methode `Append()` können Sie ein neues Element hinzufügen. Im folgenden Code wird z.B. das Root-Element an das Dokument angehängt:

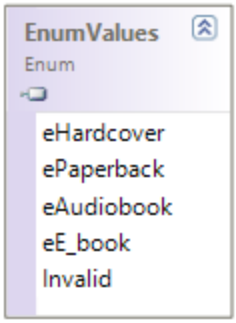
```
// Append the root element to the library
LibraryType root = doc.Library.Append();
```

Sie können den Wert eines Attributs (wie ID in diesem Beispiel) folgendermaßen definieren:

```
// Set the value of the ID attribute
book.ID.Value = 1;
```

Lesen und Schreiben von Enumerationswerten

Wenn in Ihrem XML-Schema simpleTypes als Enumerationen definiert sind, so stehen die enumerierten Werte im generierten Code als `Enum`-Werte zur Verfügung. In dem in diesem Beispiel verwendeten Schema gibt es die Buchformate hardcover, paperback, e-book, usw. Im generierten Code stehen diese Werte folglich in Form einer Enum zur Verfügung.



Mit Hilfe von Code wie dem unten gezeigten können Sie einem Objekt Enumerationswerte zuweisen:

```
// Set the format of the book (enumeration)
book.Format.EnumerationValue = BookFormatType.EnumValues.eHardcover;
```

Solche Enumerationswerte können folgendermaßen aus XML-Instanzdokumenten ausgelesen werden:

```
// Read and compare an enumeration value
if (book.Format.EnumerationValue == BookFormatType.EnumValues.ePaperback)
Console.WriteLine("This is a paperback book.");
```

Wenn eine "if"-Bedingung nicht genügt, erstellen Sie einen Switch, um jeden Enumerationswert zu ermitteln und wie erforderlich zu verarbeiten.

Arbeiten mit den Typen xs:dateTime und xs:duration

Wenn im Schema, anhand dessen Sie Code generiert haben, Uhrzeit- und Zeitdauer-Typen wie `xs:dateTime` oder `xs:duration` vorkommen, so werden diese im generierten Code in native Altova-Klassen konvertiert. Gehen Sie daher folgendermaßen vor, um einen Datums- oder Zeitdauerwert in das XML-Dokument zu schreiben:

1. Erstellen Sie ein `Altova.Types.DateTime`[1221]- oder `Altova.Types.Duration`[1226]-Objekt (entweder anhand von `System.DateTime` oder durch Verwendung von Teilen wie Stunden und Minuten; nähere Informationen siehe `Altova.Types.DateTime`[1221] und `Altova.Types.Duration`[1226]).
2. Definieren Sie das Objekt als Wert des benötigten Elements oder Attributs, z.B.:

```
// Create the library generation date using Altova DateTime class
Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);
// Append the date to the root
root.LastUpdated.Value = dt;
```

Um ein Datum oder eine Zeitdauer aus einem XML-Dokument zu lesen, gehen Sie folgendermaßen vor:

1. Deklarieren Sie den Elementwert (oder den Attributwert) als `Altova.Types.DateTime`[1221]- oder `Altova.Types.Duration`[1226]-Objekt.
2. Formatieren Sie das benötige Element oder Attribut, z.B.:

```
// Read the library generation date
Altova.Types.DateTime dt = root.LastUpdated.Value;
string dt_as_string = dt.ToString(DateTimeFormat.W3_dateTime);
Console.WriteLine("The library generation date is: " + dt_as_string);
```

Nähere Informationen dazu finde Sie in der Referenz zu den Klassen `Altova.Types.DateTime`[1221] und `Altova.Types.Duration`[1226].

Arbeiten mit abgeleiteten Typen

Wenn in Ihrem XML-Schema abgeleitete Typen (derived types) definiert sind, so können Sie die Typableitung in programmatisch erstellten oder geladenen XML-Dokumenten beibehalten. Im folgenden Codefragment wird gezeigt, wie Sie anhand des in diesem Beispiel verwendeten Schemas ein neues Buch mit dem abgeleiteten Typ `DictionaryType` erstellen.

```
// Append a dictionary (book of derived type) and populate its attributes and elements
DictionaryType dictionary = new DictionaryType(root.Book.Append().Node);
dictionary.ID.Value = 2;
dictionary.Title.Append().Value = "English-German Dictionary";
dictionary.Author.Append().Value = "John Doe";
dictionary.FromLanguage.Append().Value = "English";
dictionary.ToLanguage.Append().Value = "German";

// Since it's a derived type, make sure to set the xsi:type attribute of the book element
dictionary.SetXsiType();
```


© 2016-2022 Altova GmbH

Beachten Sie, dass es wichtig ist, dass Sie das `xsi:type` Attribut des neu erstellten Buchs definieren. Damit stellen Sie sicher, dass der Buchtyp korrekt vom Schema interpretiert wird, wenn das XML-Dokument validiert wird.

Im folgenden Codefragment gezeigt, wie ein Buch vom abgeleiteten Typ `DictionaryType` in der geladenen XML-Instanz identifiziert wird, wenn Sie Daten aus einem XML-Dokument laden. Zuerst wird im Code der Wert des `xsi:type`-Attributs des Buchs gefunden. Wenn die Namespace URI dieses Node `http://www.nanonull.com/LibrarySample` lautet und wenn das URI-Lookup-Präfix und der Typ mit dem Wert des `xsi:type`-Attributs übereinstimmen, so handelt es sich um ein Wörterbuch:

```
      // Determine if this book is of derived type
      if (book.Node.Attributes.GetNamedItem("xsi:type") != null)
      {
          // Find the value of the xsi:type attribute
          string xsiTypeValue = book.Node.Attributes.GetNamedItem("xsi:type").Value;
          // Get the namespace URI and the lookup prefix of this namespace
          string namespaceUri = book.Node.NamespaceURI;
          string prefix = book.Node.GetPrefixOfNamespace(namespaceUri);

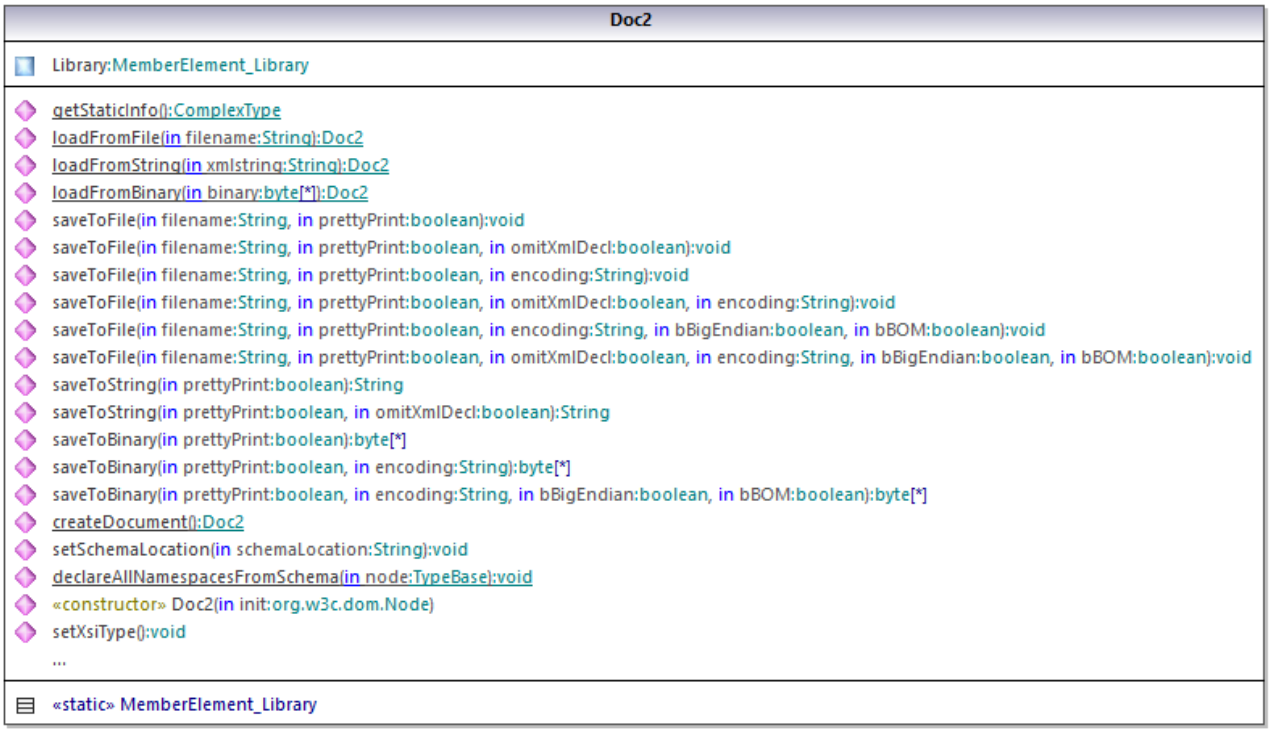
          // if this book has DictionaryType
          if (namespaceUri == "http://www.nanonull.com/LibrarySample" &&
xsiTypeValue.Equals(prefix + ":DictionaryType"))
          {
             // output additional fields
             DictionaryType dictionary = new DictionaryType(book.Node);
             Console.WriteLine("Language from: " + dictionary.FromLang.First.Value);
             Console.WriteLine("Language to: " + dictionary.ToLang.First.Value);
          }
          else
          {
              throw new Exception("Unexpected book type");
          }
      }
```

27.2.5.3 Lesen und Schreiben von XML-Dokumenten (Java)

Nachdem Sie anhand des Beispielschemas[1173]) Code generiert haben, wird ein Java-Testprojekt sowie eine Reihe von unterstützenden Altova-Bibliotheken generiert.

Informationen zu den generierten Java-Bibliotheken

Die zentrale Klasse des generierten Codes ist die Klasse `Doc2`. Sie repräsentiert das XML-Dokument. Eine solche Klasse wird für jedes Schema generiert. Ihr Name hängt vom Namen der Schemadatei ab. Beachten Sie, dass diese Klasse den Namen `Doc2` erhält, um einen möglichen Konflikt mit dem Namespace-Namen zu vermeiden. Wie im Diagramm gezeigt, bietet diese Klasse Methoden zum Laden von Dokumenten aus Dateien, Binär-Streams oder Strings (oder zum Speichern von Dokumenten in Dateien, Streams, Strings). In der Referenz zur Klasse `com.[YourSchema].[Doc]`[1248] finden Sie eine Beschreibung dieser Klasse.

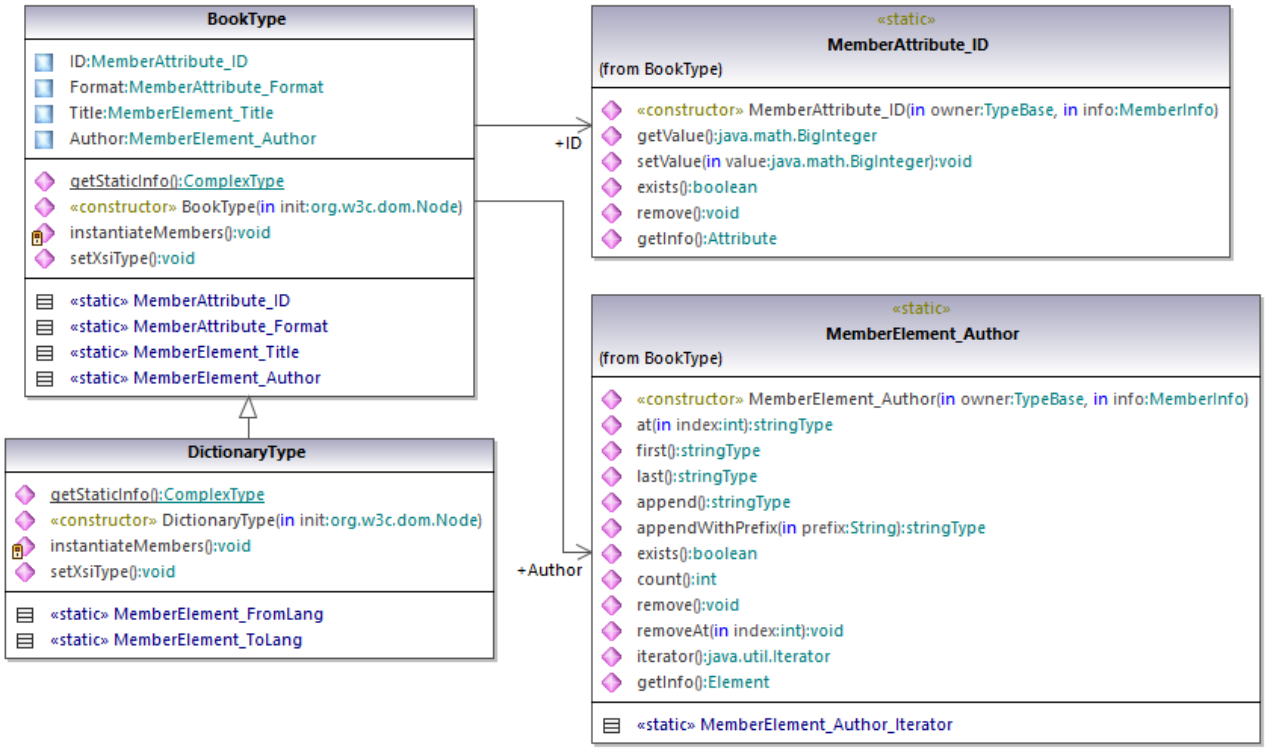


Der Member `Library` der Klasse `Doc2` stellt die eigentliche Root des Dokuments dar.

Gemäß den im Kapitel Informationen zu Schema Wrapper-Bibliotheken (Java)[1168] angeführten Codegenerierungsregeln werden für jedes Attribut und jedes Element eines Typs Member-Klassen generiert. Der Name solcher Member-Klassen erhält im generierten Code das Präfix `MemberAttribute_` bzw. `MemberElement_`. Im Diagramm unten sind Beispiele für solche Klassen `MemberAttribute_ID` und `MemberElement_Author`, die anhand des Elements **Author** bzw. des Attributs **ID** eines Buchs generiert wurden. Sie ermöglichen die programmatische Bearbeitung (z.B. Anhängen, Entfernen, Wert definieren, usw.) der entsprechenden Elemente und Attribute im XML-Instanzdokument. Nähere Informationen dazu finden Sie in der Referenz zur Klasse unter `com.[YourSchema].[YourSchemaType].MemberAttribute`[1251] und `com.[YourSchema].[YourSchemaType].MemberElement`[1251].

Der Typ **DictionaryType** ist ein complexType, der im Schema von **BookType** abgeleitet wurde, daher wird diese Beziehung auch in den generierten Klassen übernommen. Im Diagramm unten sehen Sie, dass die Klasse `DictionaryType` die Klasse `BookType` erbt.

 © 2016-2022 Altova GmbH



Wenn in Ihrem XML-Schema simpleTypes als Enumerationen definiert sind, so stehen die enumerierten Werte im generierten Code als `Enum`-Werte zur Verfügung. In dem in diesem Beispiel verwendeten Schema gibt es die Buchformate hardcover, paperback, e-book, usw. Im generierten Code stehen diese Werte folglich in Form einer Enum, d.h. als Member der Klasse CBookFormatType, `zur Verfügung.`

Schreiben eines XML-Dokuments

1. Klicken Sie im Menü **File** von Eclipse auf **Import**, wählen Sie **Existing Projects into Workspace**, und klicken Sie auf **Next**.
2. Klicken Sie neben **Select root directory** auf **Browse**, wählen Sie das Verzeichnis, in dem der Java-Code generiert werden soll aus und klicken Sie auf **Finish**.
3. Erweitern Sie im Eclipse Package Explorer das Paket **com.LibraryTest** und öffnen Sie die Datei **LibraryTest.java**.

Beim Erstellen eines Prototyps einer Applikation anhand eines häufig geänderten XML-Schemas müssen Sie eventuell immer wieder Code im selben Verzeichnis generieren, damit die Änderungen am Schema sofort im Code berücksichtigt werden. Beachten Sie, dass die generierte Testapplikation und die Altova-Bibliotheken jedes Mal, wenn Sie Code im selben Zielverzeichnis generieren, überschrieben werden. Fügen Sie daher keinen Code zur generierten Testapplikation hinzu, sondern integrieren Sie stattdessen die Altova-Bibliotheken in Ihr Projekt (siehe Integrieren von Schema Wrapper-Bibliotheken 1170).

4. Bearbeiten Sie die Methode `Example()` wie unten gezeigt.

```
protected static void example() throws Exception {
      // create a new, empty XML document
      Doc2 libDoc = Doc2.createDocument();

      // create the root element <Library> and add it to the document
      LibraryType lib = libDoc.Library3.append();

      // set the "LastUpdated" attribute
      com.altova.types.DateTime dt = new com.altova.types.DateTime(DateTime.now());
      lib.LastUpdated.setValue(dt);

      // create a new <Book> and populate its elements and attributes
      BookType book = lib.Book.append();
      book.ID.setValue(java.math.BigInteger.valueOf(1));
      book.Format.setEnumerationValue( BookFormatType.EPAPERBACK );
      book.Title.append().setValue("The XML Spy Handbook");
      book.Author.append().setValue("Altova");

      // create a dictionary (book of derived type)  and populate its elements and
attributes
      DictionaryType dict = new DictionaryType(lib.Book.append().getNode());
      dict.ID.setValue(java.math.BigInteger.valueOf(2));
      dict.Title.append().setValue("English-German Dictionary");
      dict.Format.setEnumerationValue(BookFormatType.EE_BOOK);
      dict.Author.append().setValue("John Doe");
      dict.FromLang.append().setValue("English");
      dict.ToLang.append().setValue("German");
      dict.setXsiType();

      // set the schema location (this is optional)
      libDoc.setSchemaLocation("Library.xsd");

      // save the XML document to a file with default encoding (UTF-8). "true" causes the
file to be pretty-printed.
      libDoc.saveToFile("Library1.xml", true);
   }
```

5. Bauen Sie das Java-Projekt und führen Sie es aus. Wenn der Code erfolgreich ausgeführt wurde, wird im Projektverzeichnis die Datei **Library1.xml** erstellt.

Lesen eines XML-Dokuments

1. Klicken Sie im Menü **File** von Eclipse auf **Import**, wählen Sie **Existing Projects into Workspace**, und klicken Sie auf **Next**.
2. Klicken Sie neben **Select root directory** auf **Browse**, wählen Sie das Verzeichnis, in dem der Java-Code generiert werden soll aus und klicken Sie auf **Finish**.
3. **Speichern Sie den unten gezeigten Code unter dem Namen Library1.xml** in einem lokalen Verzeichnis (Sie müssen den Pfad der Datei **Library1.xml** aus dem Beispielcode unten referenzieren).

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.nanonull.com/LibrarySample"
```


© 2016-2022 Altova GmbH

```
xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd" LastUpdated="2016-
02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/LibrarySample"
xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

4. Erweitern Sie im Eclipse Package Explorer das Paket **com.LibraryTest** und öffnen Sie die Datei **LibraryTest.java**.
5. Bearbeiten Sie die Methode `Example()` wie unten gezeigt.

```
protected static void example() throws Exception {
      // load XML document from a path, make sure to adjust the path as necessary
      Doc2 libDoc = Doc2.loadFromFile("Library1.xml");

      // get the first (and only) root element <Library>
      LibraryType lib = libDoc.Library3.first();

      // check whether an element exists:
     if (!lib.Book.exists()) {
        System.out.println("This library is empty.");
        return;
     }

     // read a DateTime schema type
     com.altova.types.DateTime dt = lib.LastUpdated.getValue();
     System.out.println("The library was last updated on: " + dt.toDateString());

     // iteration: for each <Book>...
     for (java.util.Iterator itBook = lib.Book.iterator(); itBook.hasNext();) {
        BookType book = (BookType) itBook.next();
        // output values of ID attribute and (first and only) title element
        System.out.println("ID: " + book.ID.getValue());
        System.out.println("Title: " + book.Title.first().getValue());

        // read and compare an enumeration value
        if (book.Format.getEnumerationValue() == BookFormatType.EPAPERBACK)
           System.out.println("This is a paperback book.");

        // for each <Author>...
        for (java.util.Iterator itAuthor = book.Author.iterator(); itAuthor
              .hasNext();)
           System.out.println("Author: " + ((com.Doc.xs.stringType)
itAuthor.next()).getValue());
```

```
        // find the derived type of this book
        // by looking at the value of the xsi:type attribute, using DOM
        org.w3c.dom.Node bookNode = book.getNode();
        if (bookNode.getAttributes().getNamedItem("xsi:type") != null) {
           // Get the value of the xsi:type attribute
           String xsiTypeValue =
bookNode.getAttributes().getNamedItem("xsi:type").getNodeValue();

           // Get the namespace URI and lookup prefix of this namespace
           String namespaceUri = bookNode.getNamespaceURI();
           String lookupPrefix = bookNode.lookupPrefix(namespaceUri);

           // If xsi:type matches the namespace URI and type of the book node
           if (namespaceUri == "http://www.nanonull.com/LibrarySample"
                 && (  xsiTypeValue.equals(lookupPrefix + ":DictionaryType" )))    {
              // ...then this is a book of derived type (dictionary)
              DictionaryType dictionary = new DictionaryType(   book.getNode());
              // output the value of the "FromLang" and "ToLang" elements
              System.out.println("From language: " +
dictionary.FromLang.first().getValue());
              System.out.println("To language: " + dictionary.ToLang.first().getValue());
           }
           else
           {
              // throw an error
              throw new java.lang.Error("This book has an unknown type.");
           }
        }
     }
   }
```

6. Bauen Sie das Java-Projekt und führen Sie es aus. Wenn der Code erfolgreich ausgeführt wurde, wird **Library1.xml** vom Programmcode gelesen und ihr Inhalt in der Konsolenansicht angezeigt.

Lesen und Schreiben von Elemente und Attributen

Die Werte von Attributen und Elementen können über die Methode `getValue()` der generierten Member-Element- bzw. Attribut-Klasse aufgerufen werden, z.B.:

```
// output values of ID attribute and (first and only) title element
System.out.println("ID: " + book.ID.getValue());
System.out.println("Title: " + book.Title.first().getValue());
```

Um in diesem konkreten Beispiel den Wert des Elements **Title** abzurufen, haben wir auch die Methode `First()` verwendet, da es sich hier um des erste (und einzige) **Title**-Element eines Buchs handelt. In Fällen, in denen ein bestimmtes Element nach dem Index aus einer Liste gewählt werden soll, verwenden Sie die Methode `At()`.

Um über mehrere Elemente zu iterieren, verwenden Sie entweder eine Index-basierte Iteration oder `java.util.Iterator`. So können Sie etwa folgendermaßen über die Bücher einer Bibliothek iterieren:

© 2016-2022 Altova GmbH

```
// index-based iteration
for (int j = 0; j < lib.Book.count(); ++j ) {
   // your code here
}

// alternative iteration using java.util.Iterator
for (java.util.Iterator itBook = lib.Book.iterator(); itBook.hasNext();) {
      // your code here
   }
```

Mit Hilfe der Methode `Append()` können Sie ein neues Element hinzufügen. Im folgenden Code wird z.B. ein leeres Root-Element **Library** an das Dokument angehängt:

```
// create the root element <Library> and add it to the document
LibraryType lib = libDoc.Library3.append();
```

Nachdem ein Element angehängt wurde, können Sie den Wert eines seiner Elemente oder Attribute mit Hilfe der setValue()-Methode definieren:

```
// set the value of the Title element
book.Title.append().setValue("The XML Spy Handbook");
// set the value of the ID attribute
book.ID.setValue(java.math.BigInteger.valueOf(1));
```

Lesen und Schreiben von Enumerationswerten

Wenn in Ihrem XML-Schema simpleTypes als Enumerationen definiert sind, so stehen die enumerierten Werte im generierten Code als `Enum`-Werte zur Verfügung. In dem in diesem Beispiel verwendeten Schema gibt es die Buchformate hardcover, paperback, e-book, usw. Im generierten Code stehen diese Werte folglich in Form einer `Enum zur Verfügung` (siehe die Klasse `BookFormatType` im Diagramm oben). Mit Hilfe von Code wie dem unten gezeigten können Sie einem Objekt Enumerationswerte zuweisen:

```
// set an enumeration value
book.Format.setEnumerationValue( BookFormatType.EPAPERBACK );
```

Solche Enumerationswerte können folgendermaßen aus XML-Instanzdokumenten ausgelesen werden:

```
// read an enumeration value
if (book.Format.getEnumerationValue() == BookFormatType.EPAPERBACK)
         System.out.println("This is a paperback book."
```

Wenn eine "if"-Bedingung nicht genügt, erstellen Sie einen Switch, um jeden Enumerationswert zu ermitteln und wie erforderlich zu verarbeiten.

Arbeiten mit den Typen xs:dateTime und xs:duration

Wenn im Schema, anhand dessen Sie Code generiert haben, Uhrzeit- und Zeitdauer-Typen wie `xs:dateTime` oder `xs:duration` vorkommen, so werden diese im generierten Code in native Altova-Klassen konvertiert.

Gehen Sie daher folgendermaßen vor, um einen Datums- oder Zeitdauerwert in das XML-Dokument zu schreiben:

1. Erstellen Sie ein com.altova.types.DateTime[1237]- oder com.altova.types.Duration[1241]-Objekt.
2. Definieren Sie das Objekt als Wert des benötigten Elements oder Attributs, z.B.:

```
// set the value of an attribute of DateTime type
com.altova.types.DateTime dt = new com.altova.types.DateTime(DateTime.now());
lib.LastUpdated.setValue(dt);
```

Um ein Datum oder eine Zeitdauer aus einem XML-Dokument zu lesen, gehen Sie folgendermaßen vor:

1. Deklarieren Sie den Elementwert (oder den Attributwert) als com.altova.types.DateTime[1237]- oder com.altova.types.Duration[1241]-Objekt.
2. Formatieren Sie das benötige Element oder Attribut, z.B.:

```
// read a DateTime type
com.altova.types.DateTime dt = lib.LastUpdated.getValue();
   System.out.println("The library was last updated on: " + dt.toDateString());
```

Nähere Informationen dazu finde Sie in der Referenz zu den Klassen com.altova.types.DateTime[1237] und com.altova.types.Duration[1241].

Arbeiten mit abgeleiteten Typen

Wenn in Ihrem XML-Schema abgeleitete Typen (derived types) definiert sind, so können Sie die Typableitung in programmatisch erstellten oder geladenen XML-Dokumenten beibehalten. Im folgenden Codefragment wird gezeigt, wie Sie anhand des in diesem Beispiel verwendeten Schemas ein neues Buch mit dem abgeleiteten Typ `DictionaryType` erstellen.

```
// create a dictionary (book of derived type)  and populate its elements and attributes
DictionaryType dict = new DictionaryType(lib.Book.append().getNode());
dict.ID.setValue(java.math.BigInteger.valueOf(2));
dict.Title.append().setValue("English-German Dictionary");
dict.Format.setEnumerationValue(BookFormatType.EE_BOOK);
dict.Author.append().setValue("John Doe");
dict.FromLang.append().setValue("English");
dict.ToLang.append().setValue("German");
dict.setXsiType();
```

Beachten Sie, dass es wichtig ist, dass Sie das `xsi:type` Attribut des neu erstellten Buchs definieren. Damit stellen Sie sicher, dass der Buchtyp korrekt vom Schema interpretiert wird, wenn das XML-Dokument validiert wird.

Im folgenden Codefragment gezeigt, wie ein Buch vom abgeleiteten Typ `DictionaryType` in der geladenen XML-Instanz identifiziert wird, wenn Sie Daten aus einem XML-Dokument laden. Zuerst wird im Code der Wert des `xsi:type`-Attributs des Buchs gefunden. Wenn die Namespace URI dieses Node `http://www.nanonull.com/LibrarySample` lautet und wenn das URI-Lookup-Präfix und der Typ mit dem Wert des `xsi:type`-Attributs übereinstimmen, so handelt es sich um ein Wörterbuch:

© 2016-2022 Altova GmbH

```
         // find the derived type of this book
         // by looking at the value of the xsi:type attribute, using DOM
         org.w3c.dom.Node bookNode = book.getNode();
         if (bookNode.getAttributes().getNamedItem("xsi:type") != null) {
            // Get the value of the xsi:type attribute
            String xsiTypeValue =
bookNode.getAttributes().getNamedItem("xsi:type").getNodeValue();

            // Get the namespace URI and lookup prefix of the book node
            String namespaceUri = bookNode.getNamespaceURI();
            String lookupPrefix = bookNode.lookupPrefix(namespaceUri);

            // If xsi:type matches the namespace URI and type of the book node
            if (namespaceUri == "http://www.nanonull.com/LibrarySample"
                  && (  xsiTypeValue.equals(lookupPrefix + ":DictionaryType" )))    {
               // ...then this is a book of derived type (dictionary)
               DictionaryType dictionary = new DictionaryType(   book.getNode());
               // output the value of the "FromLang" and "ToLang" elements
               System.out.println("From language: " +
dictionary.FromLang.first().getValue());
               System.out.println("To language: " +
dictionary.ToLang.first().getValue());
            }
            else
            {
               // throw an error
               throw new java.lang.Error("This book has an unknown type.");
            }
         }
```

27.2.6 Beispiel: Bestellung

In diesem Beispiel wird gezeigt, wie Sie mit Programmcode arbeiten, der anhand eines XML-Hauptschemas, durch das andere Schemas importiert wurden, generiert wurde. Jedes der importierten Schemas hat einen anderen Ziel-Namespace. Ziel ist es hier, ein XML-Dokument, in dem alle Elemente ein Präfix entsprechend ihrem Namespace erhalten, programmatisch zu erstellen. Das anhand Ihres C++-, C#- oder Java-Codes erstellte XML-Dokument sollte dabei wie das unten gezeigte aussehen:

```
<?xml version="1.0" encoding="utf-8"?>
<p:Purchase xsi:schemaLocation="http://NamespaceTest.com/Purchase Main.xsd"
            xmlns:p="http://NamespaceTest.com/Purchase"
            xmlns:o="http://NamespaceTest.com/OrderTypes"
            xmlns:c="http://NamespaceTest.com/CustomerTypes"
            xmlns:cmn="http://NamespaceTest.com/CommonTypes"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <p:OrderDetail>
        <o:Item>
            <o:ProductName>Lawnmower</o:ProductName>
            <o:Quantity>1</o:Quantity>
            <o:UnitPrice>148.42</o:UnitPrice>
        </o:Item>
```

```
    </p:OrderDetail>
    <p:PaymentMethod>VISA</p:PaymentMethod>
    <p:CustomerDetails>
        <c:Name>Alice Smith</c:Name>
        <c:DeliveryAddress>
            <cmn:Line1>123 Maple Street</cmn:Line1>
            <cmn:Line2>Mill Valley</cmn:Line2>
        </c:DeliveryAddress>
        <c:BillingAddress>
            <cmn:Line1>8 Oak Avenue</cmn:Line1>
            <cmn:Line2>Old Town</cmn:Line2>
        </c:BillingAddress>
    </p:CustomerDetails>
</p:Purchase>
```

Das in diesem Beispiel verwendete Hauptschema hat den Namen **Main.xsd**. Wie Sie im unten gezeigten Codefragment sehen, werden darin drei weitere Schemas importiert: **CommonTypes.xsd**, **CustomerTypes.xsd** und **OrderTypes.xsd**. Um dieselben Ergebnisse wie in diesem Beispiel zu erhalten, speichern Sie alle unten gezeigten Codefragmente in Dateien und verwenden Sie dieselben Namen wie oben. Beachten Sie, dass jedes der Präfixe `ord`, `pur`, `cmn` und `cust` im Schema einem Namespace zugeordnet ist (Order types, Purchase types, Common types bzw. Customer types). Im generierten Code stehen die entsprechenden Klassen für Orders, Purchases, Customers usw. dann unter ihrem jeweiligen Namespace zur Verfügung.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://NamespaceTest.com/Purchase"
      xmlns:ord="http://NamespaceTest.com/OrderTypes"
      xmlns:pur="http://NamespaceTest.com/Purchase"
      xmlns:cmn="http://NamespaceTest.com/CommonTypes"
      xmlns:cust="http://NamespaceTest.com/CustomerTypes"
      elementFormDefault="qualified">
    <xs:import schemaLocation="CommonTypes.xsd"
namespace="http://NamespaceTest.com/CommonTypes" />
    <xs:import schemaLocation="CustomerTypes.xsd"
namespace="http://NamespaceTest.com/CustomerTypes" />
    <xs:import schemaLocation="OrderTypes.xsd"
namespace="http://NamespaceTest.com/OrderTypes" />
    <xs:element name="Purchase">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="OrderDetail" type="ord:OrderType" />
                <xs:element name="PaymentMethod" type="cmn:PaymentMethodType" />
                <xs:element ref="pur:CustomerDetails" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="CustomerDetails" type="cust:CustomerType" />
</xs:schema>
```


© 2016-2022 Altova GmbH

Main.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://NamespaceTest.com/CommonTypes"
      elementFormDefault="qualified">
   <xs:complexType name="AddressType">
      <xs:sequence>
         <xs:element name="Line1" type="xs:string"/>
         <xs:element name="Line2" type="xs:string"/>
      </xs:sequence>
   </xs:complexType>
   <xs:simpleType name="PriceType">
      <xs:restriction base="xs:decimal">
         <xs:fractionDigits value="2"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:simpleType name="PaymentMethodType">
      <xs:restriction base="xs:string">
         <xs:enumeration value="VISA"/>
         <xs:enumeration value="MasterCard"/>
         <xs:enumeration value="Cash"/>
         <xs:enumeration value="AMEX"/>
      </xs:restriction>
   </xs:simpleType>
</xs:schema>
```

CommonTypes.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://NamespaceTest.com/CustomerTypes"
      xmlns:cmn="http://NamespaceTest.com/CommonTypes"
      elementFormDefault="qualified">
    <xs:import schemaLocation="CommonTypes.xsd"
namespace="http://NamespaceTest.com/CommonTypes" />
    <xs:complexType name="CustomerType">
        <xs:sequence>
            <xs:element name="Name" type="xs:string" />
            <xs:element name="DeliveryAddress" type="cmn:AddressType" />
            <xs:element name="BillingAddress" type="cmn:AddressType" />
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

CustomerTypes.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://NamespaceTest.com/OrderTypes"
      xmlns:cmn="http://NamespaceTest.com/CommonTypes"
```

```
      elementFormDefault="qualified">
    <xs:import schemaLocation="CommonTypes.xsd"
namespace="http://NamespaceTest.com/CommonTypes" />
    <xs:complexType name="OrderType">
        <xs:sequence>
            <xs:element maxOccurs="unbounded" name="Item">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="ProductName" type="xs:string" />
                        <xs:element name="Quantity" type="xs:int" />
                        <xs:element name="UnitPrice" type="cmn:PriceType" />
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

OrderTypes.xsd

Um dieses Beispiel fertig zu stellen, gehen Sie folgendermaßen vor:

1. Speichern Sie alle Schemas aus den obigen Codefragmenten auf der Festplatte und stellen Sie dabei sicher, dass die angegebenen Dateinamen beibehalten werden.
2. Generieren Sie den Schema Wrapper-Code anhand des obigen Schemas **Main.xsd**. Gehen Sie dabei vor, wie unter Generieren von Code anhand von XML-Schemas oder DTDs[1160] beschrieben. Sie sollten dadurch ein kompilierbares Programm in der Sprache Ihrer Wahl (C++, C# oder Java) erhalten.
3. Fügen Sie je nach Bedarf aus einem der folgenden Beispiel-Codefragmente Code zu Ihrem C++-, C#- oder Java-Programm hinzu.

   - XML Namespaces und Präfixe (C++)[1200]
   - XML Namespaces und Präfixe (C#)[1202]
   - XML Namespaces und Präfixe (Java)[1203]

27.2.6.1 XML Namespaces und Präfixe (C++)

Nachdem Sie anhand des Schemas "Library" (siehe Beispielschema[1197]) Code generiert haben, wird eine C#-Testapplikation sowie eine Reihe von unterstützenden Altova-Bibliotheken generiert. Wie bereits erwähnt, hat das Beispielschema (**Main.xsd**) mehrere Namespace-Deklarationen. Der generierte Code enthält daher die folgenden Namespaces, die Namespace-Aliassen (Präfixen) aus dem Schema entsprechen: `Main::ord`, `Main::pur`, `Main::cmn` und `Main::cust`.

Wenn Sie XML Namespaces und -Präfixe mit Hilfe von Schema Wrapper-Bibliotheken festlegen möchten, stehen Ihnen die folgenden Methoden zur Verfügung:

- DeclareAllNamespacesFromSchema()[1215]. Mit Hilfe dieser Methode können Sie in Ihrer XML-Instanzdatei dieselben Namespaces wie im Schema deklarieren. Wenn Sie, wie in diesem Beispiel andere Namespaces benötigen, verwenden Sie `DeclareNamespace()`, Die Methode `DeclareAllNamespacesFromSchema()` wird in diesem Beispiel nicht verwendet, weil wir hier XML-Elemente erstellen möchten, deren Präfixe sich von den im Schema deklarierten geringfügig unterscheiden.

© 2016-2022 Altova GmbH

- DeclareNamespace()[1217]. Mit Hilfe dieser Methode können Sie ein Namespace-Präfix-Attribut in einem Element erstellen oder das vorhandene überschreiben. Das Element muss bereits erstellt worden sein - entweder mit der Methode **`append()`** oder mit **`appendWithPrefix()`**, wie weiter unten gezeigt.
- appendWithPrefix().[1219] Mit Hilfe dieser Methode können Sie ein bestimmtes Präfix an ein Instanzelement anhängen. Um das in diesem Beispiel gezeigte XML-Instanzdokument zu erstellen, genügte es, diese Methode nur für das Root-Element aufzurufen. Alle anderen Elemente wurden durch Aufruf von append()[1219] erstellt. Ihr Präfix wurde automatisch gemäß den obigen Regeln auf Basis des Namespace hinzugefügt.

Im Codefragment unten sehen Sie, wie Sie ein XML-Dokument mit mehreren Namespace-Deklarationen und als Präfix vorangestellten Elementnamen erstellen. Dabei wird eine Bestellungsinstanz, wie im Beispiel: Bestellung[1197] gezeigt, generiert. Beachten Sie bitte, dass einige Präfixe in der XML-Instanz zur Darstellungszwecken außer Kraft gesetzt werden (d.h. Sie sind nicht genau gleich wie die im Schema deklarierten).

```
void Example()
{
   // Create the XML document and append the root element
   Main::pur::CMain doc = Main::pur::CMain::CreateDocument();
   Main::pur::CPurchaseType purchase = doc.Purchase.appendWithPrefix(_T("p"));

   // Set schema location
   doc.SetSchemaLocation(_T("Main.xsd"));

   // Declare namespaces on root element
   purchase.DeclareNamespace(_T("o"), _T("http://NamespaceTest.com/OrderTypes"));
   purchase.DeclareNamespace(_T("c"), _T("http://NamespaceTest.com/CustomerTypes"));
   purchase.DeclareNamespace(_T("cmn"), _T("http://NamespaceTest.com/CommonTypes"));

   // Append the OrderDetail element
   Main::ord::COrderType order = purchase.OrderDetail.append();
   Main::ord::CItemType item = order.Item.append();
   item.ProductName.append() = _T("Lawnmower");
   item.Quantity.append() = 1;
   item.UnitPrice.append() = 148.42;

   // Append the PaymentMethod element
   Main::cmn::CPaymentMethodTypeType paymentMethod = purchase.PaymentMethod.append();
   paymentMethod.SetEnumerationValue(Main::cmn::CPaymentMethodTypeType::k_VISA);

   // Append the CustomerDetails element
   Main::cust::CCustomerType customer = purchase.CustomerDetails.append();
   customer.Name.append() = _T("Alice Smith");
   Main::cmn::CAddressType deliveryAddress = customer.DeliveryAddress.append();
   deliveryAddress.Line1.append() = _T("123 Maple Street");
   deliveryAddress.Line2.append() = _T("Mill Valley");
   Main::cmn::CAddressType billingAddress = customer.BillingAddress.append();
   billingAddress.Line1.append() = _T("8 Oak Avenue");
   billingAddress.Line2.append() = _T("Old Town");

   // Save to file and release object from memory
   doc.SaveToFile(_T("Main1.xml"), true);
   doc.DestroyDocument();
}
```

27.2.6.2 XML Namespaces und Präfixe (C#)

Nachdem Sie anhand des Schemas "Library" (siehe Beispielschema[1197]) Code generiert haben, wird eine C#-Testapplikation sowie eine Reihe von unterstützenden Altova-Bibliotheken generiert. Wie bereits erwähnt, hat das Beispielschema (**Main.xsd**) mehrere Namespace-Deklarationen. Der generierte Code enthält daher die folgenden Namespaces, die Namespace-Aliassen (Präfixen) aus dem Schema entsprechen: `Main.ord`, `Main.pur`, `Main.cmn` und `Main.cust`.

Wenn Sie XML Namespaces und -Präfixe mit Hilfe von Schema Wrapper-Bibliotheken festlegen möchten, stehen Ihnen die folgenden Methoden zur Verfügung:

- DeclareAllNamespacesFromSchema()[1231]. Mit Hilfe dieser Methode können Sie in Ihrer XML-Instanzdatei dieselben Namespaces wie im Schema deklarieren. Wenn Sie, wie in diesem Beispiel andere Namespaces benötigen, verwenden Sie `DeclareNamespace()`, Die Methode `DeclareAllNamespacesFromSchema()` wird in diesem Beispiel nicht verwendet, weil wir hier XML-Elemente erstellen möchten, deren Präfixe sich von den im Schema deklarierten geringfügig unterscheiden.
- DeclareNamespace()[1233]. Mit Hilfe dieser Methode können Sie ein Namespace-Präfix-Attribut in einem Element erstellen oder das vorhandene überschreiben. Das Element muss bereits erstellt worden sein - entweder mit der Methode `Append()` oder mit `AppendWithPrefix()`, wie weiter unten gezeigt.
- AppendWithPrefix().[1235] Mit Hilfe dieser Methode können Sie ein bestimmtes Präfix an ein Instanzelement anhängen. Um das in diesem Beispiel gezeigte XML-Instanzdokument zu erstellen, genügte es, diese Methode nur für das Root-Element aufzurufen. Alle anderen Elemente wurden durch Aufruf von Append()[1235] erstellt. Ihr Präfix wurde automatisch gemäß den obigen Regeln auf Basis des Namespace hinzugefügt.

Im Codefragment unten sehen Sie, wie Sie ein XML-Dokument mit mehreren Namespace-Deklarationen und als Präfix vorangestellten Elementnamen erstellen. Dabei wird eine Bestellungsinstanz, wie im Beispiel: Bestellung[1197] gezeigt, generiert. Beachten Sie bitte, dass einige Präfixe in der XML-Instanz zur Darstellungszwecken außer Kraft gesetzt werden (d.h. Sie sind nicht genau gleich wie die im Schema deklarierten).

```
protected static void Example()
{
   // Create the XML document and append the root element
   pur.Main2 doc = pur.Main2.CreateDocument();
   pur.PurchaseType purchase = doc.Purchase.AppendWithPrefix("p");

   // Set schema location
   doc.SetSchemaLocation(@"Main.xsd");

   // Declare namespaces on root element
   purchase.DeclareNamespace("o", "http://NamespaceTest.com/OrderTypes");
   purchase.DeclareNamespace("c", "http://NamespaceTest.com/CustomerTypes");
   purchase.DeclareNamespace("cmn", "http://NamespaceTest.com/CommonTypes");

   // Append the OrderDetail element
   ord.OrderType order = purchase.OrderDetail.Append();
   ord.ItemType item = order.Item.Append();
   item.ProductName.Append().Value = "Lawnmower";
   item.Quantity.Append().Value = 1;
```


© 2016-2022 Altova GmbH

```
    item.UnitPrice.Append().Value = 148.42M;

    // Append the PaymentMethod element
    cmn.PaymentMethodTypeType paymentMethod = purchase.PaymentMethod.Append();
    paymentMethod.EnumerationValue = cmn.PaymentMethodTypeType.EnumValues.eVISA;

    // Append the CustomerDetails element
    cust.CustomerType customer = purchase.CustomerDetails.Append();
    customer.Name.Append().Value = "Alice Smith";
    cmn.AddressType deliveryAddress = customer.DeliveryAddress.Append();
    deliveryAddress.Line1.Append().Value = "123 Maple Street";
    deliveryAddress.Line2.Append().Value = "Mill Valley";
    cmn.AddressType billingAddress = customer.BillingAddress.Append();
    billingAddress.Line1.Append().Value = "8 Oak Avenue";
    billingAddress.Line2.Append().Value = "Old Town";

    // Save to file
    doc.SaveToFile("PurchaseOrder.xml", true);
}
```

27.2.6.3 XML Namespaces und Präfixe (Java)

Nachdem Sie anhand des Beispielschemas[1197]) Code generiert haben, wird eine Java-Testapplikation sowie eine Reihe von unterstützenden Altova-Bibliotheken generiert. Wie bereits erwähnt, hat das Beispielschema (**Main.xsd**) mehrere Namespace-Deklarationen. Der generierte Code enthält daher die folgenden Namespaces, die Namespace-Aliassen (Präfixen) aus dem Schema entsprechen: `com.Main.ord`, `com.Main.pur`, `com.Main.cmn` und `com.Main.cust`.

Wenn Sie XML Namespaces und -Präfixe mit Hilfe von Schema Wrapper-Bibliotheken festlegen möchten, stehen Ihnen die folgenden Methoden zur Verfügung:

- declareAllNamespacesFromSchema()[1248]. Mit Hilfe dieser Methode können Sie in Ihrer XML-Instanzdatei dieselben Namespaces wie im Schema deklarieren. Wenn Sie, wie in diesem Beispiel andere Namespaces benötigen, verwenden Sie `declareNamespace()`, Die Methode `declareAllNamespacesFromSchema()` wird in diesem Beispiel nicht verwendet, weil wir hier XML-Elemente erstellen möchten, deren Präfixe sich von den im Schema deklarierten geringfügig unterscheiden.
- declareNamespace()[1249]. Mit Hilfe dieser Methode können Sie ein Namespace-Präfix-Attribut in einem Element erstellen oder das vorhandene überschreiben. Das Element muss bereits erstellt worden sein - entweder mit der Methode `append()` oder mit `appendWithPrefix()`, wie weiter unten gezeigt.
- appendWithPrefix().[1251] Mit Hilfe dieser Methode können Sie ein bestimmtes Präfix an ein Instanzelement anhängen. Um das in diesem Beispiel gezeigte XML-Instanzdokument zu erstellen, genügte es, diese Methode nur für das Root-Element aufzurufen. Alle anderen Elemente wurden durch Aufruf von append()[1251] erstellt. Ihr Präfix wurde automatisch gemäß den obigen Regeln auf Basis des Namespace hinzugefügt.

Im Codefragment unten sehen Sie, wie Sie ein XML-Dokument mit mehreren Namespace-Deklarationen und als Präfix vorangestellten Elementnamen erstellen. Dabei wird eine Bestellungsinstanz, wie im Beispiel: Bestellung[1197] gezeigt, generiert. Beachten Sie bitte, dass einige Präfixe in der XML-Instanz zur Darstellungszwecken außer Kraft gesetzt werden (d.h. Sie sind nicht genau gleich wie die im Schema deklarierten).

```
protected static void example() throws Exception {
   // Create the XML document and append the root element
   com.Main.pur.Main2 doc = com.Main.pur.Main2.createDocument();
   com.Main.pur.PurchaseType purchase = doc.Purchase.appendWithPrefix("p");

   // Set schema location
   doc.setSchemaLocation("Main.xsd");

   // Declare namespaces on root element
   purchase.declareNamespace("o", "http://NamespaceTest.com/OrderTypes");
   purchase.declareNamespace("c", "http://NamespaceTest.com/CustomerTypes");
   purchase.declareNamespace("cmn", "http://NamespaceTest.com/CommonTypes");

   // Append the OrderDetail element
   com.Main.ord.OrderType order = purchase.OrderDetail.append();
   com.Main.ord.ItemType item = order.Item.append();
   item.ProductName.append().setValue("Lawnmower");
   item.Quantity.append().setValue(1);
   java.math.BigDecimal price = new java.math.BigDecimal("148.42");
   item.UnitPrice.append().setValue(price);

   // Append the PaymentMethod element
   com.Main.cmn.PaymentMethodTypeType paymentMethod = purchase.PaymentMethod.append();
   paymentMethod.setEnumerationValue(com.Main.cmn.PaymentMethodTypeType.EVISA);

   // Append the CustomerDetails element
   com.Main.cust.CustomerType customer = purchase.CustomerDetails.append();
   customer.Name.append().setValue("Alice Smith");
   com.Main.cmn.AddressType deliveryAddress = customer.DeliveryAddress.append();
   deliveryAddress.Line1.append().setValue("123 Maple Street");
   deliveryAddress.Line2.append().setValue("Mill Valley");
   com.Main.cmn.AddressType billingAddress = customer.BillingAddress.append();
   billingAddress.Line1.append().setValue("8 Oak Avenue");
   billingAddress.Line2.append().setValue("Old Town");

   // Save to file
   doc.saveToFile("PurchaseOrder.xml", true);
}
```


© 2016-2022 Altova GmbH

27.3 Referenz zu generierten Klassen (C++)

Dieses Kapitel enthält eine Beschreibung der mit XMLSpy anhand einer DTD oder eines XML-Schemas (siehe Generieren von Code anhand von XML-Schemas oder DTDs[1160]) generierten C++-Klassen. Sie können diese Klassen in Ihren Code integrieren, um XML-Dokumente zu lesen, zu bearbeiten und zu schreiben.

Anmerkung: Der generierte Code enthält andere unterstützende Klassen, die hier nicht aufgelistet sind und Änderungen unterworfen sind.

27.3.1 altova::DateTime

Mit Hilfe dieser Klasse können XML-Attribute oder -Elemente, die Datums- und Uhrzeittypen wie z.B. `xs:dateTime` haben, verarbeitet werden.

Konstruktoren

Name	Beschreibung
`DateTime()`	Initialisiert eine neue Instanz der `DateTime`-Klasse auf 12:00:00 Mitternacht, 1. Jänner 0001.
`DateTime(__int64 value, short timezone)`	Initialisiert eine neue Instanz der `DateTime`-Klasse. Der Parameter `value` repräsentiert die Anzahl der Zeiteinheiten (100-Nanosekunden-Intervalle), die seit 12:00:00 Mitternacht, 1. Jänner 0001 verstrichen sind.
`DateTime(int year, unsigned char month, unsigned char day, unsigned char hour, unsigned char minute, double second)`	Initialisiert eine neue Instanz der `DateTime`-Klasse auf das als Argument bereitgestellte Jahr, den Monat, Tag, die Stunde, Minute und Sekunde.
`DateTime(int year, unsigned char month, unsigned char day, unsigned char hour, unsigned char minute, double second, short timezone)`	Initialisiert eine neue Instanz der `DateTime`-Klasse auf das als Argument bereitgestellte Jahr, den Monat, Tag, die Stunde, Minute, Sekunde und Zeitzone. Die Zeitzone wird in Minuten ausgedrückt und kann positiv oder negativ sein. So wird z.B. die Zeitzone "UTC-01:00" als "-60" ausgedrückt.

Methoden

Name	Beschreibung
`unsigned char Day() const`	Gibt den Tag des Monats des aktuellen `DateTime`-Objekts zurück. Die Rückgabewerte liegen im Bereich von 1 bis 31.
`int DayOfYear() const`	Gibt den Tag des Jahres des aktuellen `DateTime`-Objekts zurück. Die Rückgabewerte liegen im Bereich von 1 bis 366.

Name	Beschreibung
`bool HasTimezone() const`	Gibt den Booleschen Wert **true** zurück, wenn für das aktuelle `DateTime`-Objekt eine Zeitzone definiert ist; gibt andernfalls **false** zurück.
`unsigned char Hour() const`	Gibt die Stunde des aktuellen `DateTime`-Objekts zurück. Die Rückgabewerte liegen im Bereich von 0 bis 23.
`static bool IsLeapYear(int year)`	Gibt den Booleschen Wert **true** zurück, wenn das Jahr der `DateTime`-Klasse ein Schaltjahr ist; gibt andernfalls **false** zurück.
`unsigned char Minute() const`	Gibt die Minute des aktuellen `DateTime`-Objekts zurück. Die Rückgabewerte liegen im Bereich von 0 bis 59.
`unsigned char Month() const`	Gibt den Monat des aktuellen `DateTime`-Objekts zurück. Die Rückgabewerte liegen im Bereich von 1 bis 12.
`__int64 NormalizedValue() const`	Gibt den als Coordinated Universal Time (UTC) ausgedrückten Wert des aktuellen `DateTime`-Objekts zurück.
`double Second() const`	Gibt die Sekunde des aktuellen `DateTime`-Objekts zurück. Die Rückgabewerte liegen im Bereich von 0 bis 59.
`void SetTimezone(short tz)`	Setzt die Zeitzone des aktuellen `DateTime`-Objekts auf den als Argument bereigestellten Zeitzonenwert. Das Argument **tz** wird in Minuten ausgedrückt und kann positiv oder negativ sein.
`short Timezone() const`	Gibt die Zeitzone des aktuellen `DateTime`-Objekts in Minuten zurück. Stellen Sie vor Verwendung dieser Methode durch Aufruf der `HasTimezone()`-Methode sicher, dass das Objekt tatsächlich eine Zeitzone hat.
`__int64 Value() const`	Gibt den Wert des `DateTime` Objekts zurück, ausgedrückt in der Anzahl der Zeiteinheiten (100-Nanosekunden-Intervalle), die seit 12:00:00 Mitternacht, 1. Jänner 0001 verstrichen sind.
`int Weekday() const`	Gibt den Wochentag des aktuellen `DateTime`-Objekts als Ganzzahl zurück. Die Werte liegen im Bereich von 0 bis 6, wobei 0 Montag ist (ISO-8601).
`int Weeknumber() const`	Gibt den Nummer der Kalenderwoche des aktuellen `DateTime`-Objekts zurück. Die Werte entsprechen der ISO-8601-Norm.
`int WeekOfMonth() const`	Gibt den Nummer der Woche des Monats des aktuellen `DateTime`-Objekts zurück. Die Werte entsprechen der ISO-8601-Norm.
`int Year() const`	Gibt das Jahr des aktuellen `DateTime`-Objekts zurück.

Beispiel

```
void Example()
{
	// initialize a new DateTime instance to 12:00:00 midnight, January 1st, 0001
```


© 2016-2022 Altova GmbH

```
	altova::DateTime dt1 = altova::DateTime();

	// initialize a new DateTime instance using the year, month, day, hour, minute,
and second
	altova::DateTime dt2 = altova::DateTime(2015, 11, 10, 9, 8, 7);

	// initialize a new DateTime instance using the year, month, day, hour, minute,
second, and UTC +01:00 timezone
	 altova::DateTime dt = altova::DateTime(2015, 11, 22, 13, 53, 7, 60);

	// Get the value of this DateTime object
	std::cout << "The number of ticks of the DateTime object is: " << dt.Value() <<
std::endl;

	// Get the year
	cout << "The year is: " << dt.Year() << endl;
	// Get the month
	cout << "The month is: " << (int)dt.Month() << endl;
	// Get the day of the month
	cout << "The day of the month is: " << (int) dt.Day() << endl;
	// Get the day of the year
	cout << "The day of the year is: " << dt.DayOfYear() << endl;
	// Get the hour
	cout << "The hour is: " << (int) dt.Hour() << endl;
	// Get the minute
	cout << "The minute is: " << (int) dt.Minute() << endl;
	// Get the second
	cout << "The second is: " << dt.Second() << endl;
	// Get the weekday
	cout << "The weekday is: " << dt.Weekday() << endl;
	// Get the week number
	cout << "The week of year is: " << dt.Weeknumber() << endl;
	// Get the week in month
	cout << "The week of month is: " << dt.WeekOfMonth() << endl;

	// Check whether a DateTime instance has a timezone
	if (dt.HasTimezone() == TRUE)
	{
		// output the value of the Timezone
		cout <<  "The timezone is: " << dt.Timezone() << endl;
	}
	else
	{
		cout <<  "No timezone has been defined." << endl;
	}

	// Construct a DateTime object with a timezone UTC+01:00 (Vienna)
	altova::DateTime vienna_dt = DateTime(2015, 11, 23, 14, 30, 59, +60);
	// Output the result in readable format
	cout << "The Vienna time: "
		 << (int) vienna_dt.Month()
		 << "-" << (int) vienna_dt.Day()
		 << " " << (int) vienna_dt.Hour()
		 << ":" << (int) vienna_dt.Minute()
		 << ":" << (int) vienna_dt.Second()
```

```
                << endl;

        // Convert the value to UTC time
        DateTime utc_dt = DateTime(vienna_dt.NormalizedValue());
        // Output the result in readable format
        cout << "The UTC time:    "
               << (int) utc_dt.Month()
               << "-" << (int) utc_dt.Day()
               << " " << (int) utc_dt.Hour()
               << ":" << (int) utc_dt.Minute()
               << ":" << (int) utc_dt.Second()
               << endl;

        // Check if a year is a leap year
        int year = 2016;
        if( altova::DateTime::IsLeapYear(year) )
        { cout << year << " is a leap year" << endl; }
        else
        { cout << year << " is not a leap year" << endl; }
}
```

27.3.2 altova::Duration

This class enables you to process XML attributes or elements of type `xs:duration`.
Mit Hilfe dieser Klasse können XML-Attribute oder -Elemente vom Typ `xs:duration` verarbeitet werden.

Konstruktoren

Name	Beschreibung
`Duration()`	Initialisiert eine neue Instanz der `Duration`-Klasse auf einen leeren Wert.
`Duration(const DayTimeDuration& dt)`	Initialisiert eine neue Instanz der `Duration`-Klasse auf eine durch das Argument `dt` definierte Zeitdauer (siehe altova::DayTimeDuration 1210).
`Duration(const YearMonthDuration& ym)`	Initialisiert eine neue Instanz der `Duration`-Klasse auf eine durch das Argument `ym` definierte Zeitdauer (siehe altova::YearMonthDuration 1211).
`Duration(const YearMonthDuration& ym, const DayTimeDuration& dt)`	Initialisiert eine neue Instanz der `Duration`-Klasse auf eine durch die Argumente `dt` und `ym` definierte Zeitdauer (siehe altova::YearMonthDuration 1211 und altova::DayTimeDuration 1210).

Methoden

Name	Beschreibung
`int Days() const`	Gibt die Anzahl der Tage in der aktuellen `Duration`-Instanz zurück.

© 2016-2022 Altova GmbH

Name	Beschreibung
`DayTimeDuration DayTime() const`	Gibt die als `DayTimeDuration` Objekt ausgedrückte Zeitdauer in Tagen und Uhrzeitwerten zurück (siehe altova::DayTimeDuration 1210).
`int Hours() const`	Gibt die Anzahl der Stunden in der aktuellen `Duration`-Instanz zurück.
`bool IsNegative() const`	Gibt den Booleschen Wert **true** zurück, wenn die aktuelle `Duration`-Instanz negativ ist.
`bool IsPositive() const`	Gibt den Booleschen Wert **true** zurück, wenn die aktuelle `Duration`-Instanz positiv ist.
`int Minutes() const`	Gibt die Anzahl der Minuten in der aktuellen `Duration`-Instanz zurück.
`int Months() const`	Gibt die Anzahl der Monate in der aktuellen `Duration`-Instanz zurück.
`double Seconds() const`	Gibt die Anzahl der Sekunden in der aktuellen `Duration`-Instanz zurück.
`YearMonthDuration YearMonth() const`	Gibt die als `YearMonthDuration`-Objekt ausgedrückte Zeitdauer in Jahren und Monaten zurück (siehe altova::YearMonthDuration 1211).
`int Years() const`	Gibt die Anzahl der Jahre in der aktuellen `Duration`-Instanz zurück.

Beispiel

Im den folgenden Codefragment wird gezeigt, wie ein neues `Duration`-Objekten erstellt und Werte daraus ausgelesen werden.

```
void ExampleDuration()
{
	// Create an empty Duration object
	altova::Duration empty_duration = altova::Duration();

	// Create a Duration object using an existing duration value
	altova::Duration duration1 = altova::Duration(empty_duration);

	// Create a YearMonth duration of six years and five months
	altova::YearMonthDuration yrduration = altova::YearMonthDuration(6, 5);

	// Create a DayTime duration of four days, three hours, two minutes, and one
second
	altova::DayTimeDuration dtduration = altova::DayTimeDuration(4, 3, 2, 1);

	// Create a Duration object by combining the two previously created durations
	altova::Duration duration = altova::Duration(yrduration, dtduration);

	// Get the number of years in this Duration instance
	cout << "Years:  " << duration.Years() << endl;
```

```
        // Get the number of months in this Duration instance
        cout << "Months: " << duration.Months() << endl;

        // Get the number of days in this Duration instance
        cout << "Days:   " << duration.Days() << endl;

        // Get the number of hours in this Duration instance
        cout << "Hours:  " << duration.Hours() << endl;

        // Get the number of hours in this Duration instance
        cout << "Minutes: " << duration.Minutes() << endl;

        // Get the number of seconds in this Duration instance
        cout << "Seconds: " << duration.Seconds() << endl;
}
```

27.3.3 altova::DayTimeDuration

Mit Hilfe dieser Klasse können XML-Schema-Zeitdauertypen, die aus einem Tages- und Uhrzeitteil bestehen, verarbeitet werden.

Konstruktoren

Name	Beschreibung
`DayTimeDuration()`	Initialisiert eine neue Instanz der `DayTimeDuration`-Klasse auf einen leeren Wert.
`DayTimeDuration(int days, int hours, int minutes, double seconds)`	Initialisiert eine neue Instanz der `DayTimeDuration`-Klasse auf die als Argumente bereitgestellte Anzahl von Tagen, Stunden, Minuten und Sekunden.
`explicit DayTimeDuration(__int64 value)`	Initialisiert eine neue Instanz der `DayTimeDuration`-Klasse auf eine Zeitdauer, die aus der im Argument **value** bereitgestellte Anzahl von Zeiteinheiten (100-Nanosekunden-Intervalle) besteht.

Methoden

Name	Beschreibung
`int Days() const`	Gibt die Anzahl der Tage in der aktuellen `DayTimeDuration`-Instanz zurück.
`int Hours() const`	Gibt die Anzahl der Stunden in der aktuellen `DayTimeDuration`-Instanz zurück.
`bool IsNegative() const`	Gibt den Booleschen Wert **true** zurück, wenn die aktuelle `DayTimeDuration`-Instanz negativ ist.

© 2016-2022 Altova GmbH

`bool IsPositive() const`	Gibt den Booleschen Wert **true** zurück, wenn die aktuelle `DayTimeDuration`-Instanz positiv ist.
`int Minutes() const`	Gibt die Anzahl der Minuten in der aktuellen `DayTimeDuration`-Instanz zurück.
`double Seconds() const`	Gibt die Anzahl der Sekunden in der aktuellen `DayTimeDuration`-Instanz zurück.
`__int64 Value() const`	Gibt den Wert (in Zeiteinheiten) der aktuellen `DayTimeDuration`-Instanz zurück.

27.3.4 altova::YearMonthDuration

Mit Hilfe dieser Klasse können XML-Schema-Zeitdauertypen, die aus einem Jahres- und Monatsteil bestehen, verarbeitet werden.

Konstruktoren

Name	Beschreibung
`YearMonthDuration()`	Initialisiert eine neue Instanz der `YearMonthDuration`-Klasse auf einen leeren Wert.
`YearMonthDuration(int years, int months)`	Initialisiert eine neue Instanz der `YearMonthDuration`-Klasse auf die in den Argumenten **years** und **months** bereitgestellte Anzahl von Tagen und Monaten.
`explicit YearMonthDuration(int value)`	Initialisiert eine neue Instanz der `YearMonthDuration`-Klasse auf eine Zeitdauer, die aus der im Argument **value** bereitgestellte Anzahl von Zeiteinheiten (100-Nanosekunden-Intervalle) besteht.

Methoden

Name	Beschreibung
`bool IsNegative() const`	Gibt den Booleschen Wert **true** zurück, wenn die aktuelle `YearMonthDuration`-Instanz negativ ist.
`bool IsPositive() const`	Gibt den Booleschen Wert **true** zurück, wenn die aktuelle `YearMonthDuration`-Instanz positiv ist.
`int Months() const`	Gibt die Anzahl der Monate in der aktuellen `YearMonthDuration`-Instanz zurück.
`int Value() const`	Gibt den Wert (in Zeiteinheiten) der aktuellen `YearMonthDuration`-Instanz zurück.
`int Years()`	Gibt die Anzahl der Jahre in der aktuellen `YearMonthDuration`-Instanz zurück.

27.3.5 altova::meta::Attribute

Mit Hilfe dieser Klasse können Sie Schemainformationen über anhand von Attributen generierte Klassen aufrufen. Beachten Sie, dass diese Klasse nicht dazu gedacht ist, dynamische Informationen über bestimmte Instanzen eines Attributs in einem XML-Dokument bereitzustellen. Es dient stattdessen dazu, Informationen über ein bestimmtes im XML-Schema definiertes Attribut programmatisch abzurufen.

Methoden

Name	Beschreibung
`SimpleType GetDataType()`	Gibt den Typ des Attributinhalts zurück.
`string_type GetLocalName()`	Gibt den lokalen Namen des Attributs zurück.
`string_type GetNamespaceURI()`	Gibt die Namespace URI des Attributs zurück.
`bool IsRequired()`	Gibt "true" zurück, wenn das Attribut obligatorisch ist.

Operatoren

Name	Beschreibung
`bool operator()`	Gibt "true" zurück, wenn dies nicht das NULL-Attribut ist.
`bool operator!()`	Gibt "true" zurück, wenn dies das NULL-Attribut ist.

27.3.6 altova::meta::ComplexType

Mit Hilfe dieser Klasse können Sie Schemainformationen über anhand von complexTypes generierte Klassen aufrufen. Beachten Sie, dass diese Klasse nicht dazu gedacht ist, dynamische Informationen über bestimmte Instanzen eines complexType in einem XML-Dokument bereitzustellen. Es dient stattdessen dazu, Informationen über einen bestimmten im XML-Schema definierten complexType programmatisch abzurufen.

Methoden

Name	Beschreibung
`Attribute FindAttribute(const char_type* localName, const char_type* namespaceURI)`	Sucht das Attribut mit dem angegebenen lokalen Namen und der angegebenen Namespace URI.
`Element FindElement(const char_type* localName, const char_type* namespaceURI)`	Sucht das Element mit dem angegebenen lokalen Namen und der angegebenen Namespace URI.
`std::vector<Attribute> GetAttributes()`	Gibt eine Liste aller Attribute zurück.
`ComplexType GetBaseType()`	Gibt den Basistyp dieses Typs zurück.

© 2016-2022 Altova GmbH

Name	Beschreibung
`SimpleType GetContentType()`	Gibt den simpleType des Inhalts zurück.
`std::vector<Element> GetElements()`	Gibt eine Liste aller Elemente zurück.
`string_type GetLocalName()`	Gibt den lokalen Namen des Typs zurück.
`string_type GetNamespaceURI()`	Gibt die Namespace URI des Typs zurück.

Operatoren

Name	Beschreibung
`bool operator()`	Gibt "true" zurück, wenn dies nicht der NULL complexType ist.
`bool operator!()`	Gibt "true" zurück, wenn dies der NULL complexType ist.

27.3.7 altova::meta::Element

Mit Hilfe dieser Klasse können Sie Informationen über anhand von Schema-Elementen generierte Klassen aufrufen. Beachten Sie, dass diese Klasse nicht dazu gedacht ist, dynamische Informationen über bestimmte Instanzen eines Elements in einem XML-Dokument bereitzustellen. Es dient stattdessen dazu, Informationen über ein bestimmtes im XML-Schema definiertes Element programmatisch abzurufen.

Methoden

Name	Beschreibung
`ComplexType GetDataType()`	Gibt den Typ des Elements zurück. Beachten Sie, dass dies immer ein complexType ist, selbst wenn er im Originalschema als simpleType deklariert wurde. Verwenden Sie getContentType() des zurückgegebenen Objekts zum den simple content type abzurufen.
`string_type GetLocalName()`	Gibt den lokalen Namen des Elements zurück.
`unsigned int GetMaxOccurs()`	Gibt den im Schema definierten maxOccurs-Wert zurück.
`unsigned int GetMinOccurs()`	Gibt den im Schema definierten minOccurs-Wert zurück.
`string_type GetNamespaceURI()`	Gibt die Namespace URI des Elements zurück.

Operatoren

Name	Beschreibung
`bool operator()`	Gibt "true" zurück, wenn dies nicht das NULL-Element ist.
`bool operator!()`	Gibt "true" zurück, wenn dies das NULL-Element ist.

27.3.8 altova::meta::SimpleType

Mit Hilfe dieser Klasse können Sie Schemainformationen über anhand von simpleTypes generierte Klassen aufrufen. Beachten Sie, dass diese Klasse nicht dazu gedacht ist, dynamische Informationen über bestimmte Instanzen eines simpleType in einem XML-Dokument bereitzustellen. Es dient stattdessen dazu, Informationen über einen bestimmten im XML-Schema definierten simpleTypes programmatisch abzurufen.

Methoden

Name	Beschreibung
`SimpleType GetBaseType()`	Gibt den Basistyp dieses Typs zurück.
`std::vector<string_type> GetEnumerations()`	Gibt eine Liste aller Enumeration Facets zurück.
`unsigned int GetFractionDigits()`	Gibt den Wert dieses Facet zurück.
`unsigned int GetLength()`	Gibt den Wert dieses Facet zurück.
`string_type GetLocalName()`	Gibt den lokalen Namen des Tys zurück.
`string_type GetMaxExclusive()`	Gibt den Wert dieses Facet zurück.
`string_type GetMaxInclusive()`	Gibt den Wert dieses Facet zurück.
`unsigned int GetMaxLength()`	Gibt den Wert dieses Facet zurück.
`string_type GetMinExclusive()`	Gibt den Wert dieses Facet zurück.
`string_type GetMinInclusive()`	Gibt den Wert dieses Facet zurück.
`unsigned int GetMinLength()`	Gibt den Wert dieses Facet zurück.
`string_type GetNamespaceURI()`	Gibt die Namespace URI des Typs zurück.
`std::vector<string_type> GetPatterns()`	Gibt eine Liste aller Pattern Facets zurück.
`unsigned int GetTotalDigits()`	Gibt den Wert dieses Facet zurück.
`WhitespaceType GetWhitespace()`	Gibt den Wert des Whitespace Facet zurück. Er kann einer der folgenden Werte sein: • Whitespace_Unknown • Whitespace_Preserve • Whitespace_Replace • Whitespace_Collapse

© 2016-2022 Altova GmbH

Operatoren

Name	Beschreibung
`bool operator()`	Gibt "true" zurück, wenn dies nicht der NULL-simpleType ist.
`bool operator!()`	Gibt "true" zurück, wenn dies der NULL-simpleType ist.

27.3.9 [YourSchema]::[CDoc]

Wenn anhand eines XML-Schemas Code generiert wird, stellt der generierte Code eine Dokumentklasse mit demselben Namen wie dem des Schemas bereit. Diese Klasse enthält alle möglichen Root-Elemente als Members sowie die folgenden Methoden. Beachten Sie, dass "CDoc" in den unten aufgelisteten Methodennamen für den Namen der generierten Dokumentklasse selbst steht.

Methoden

Name	Beschreibung
`static CDoc CreateDocument()`	Erstellt ein neues leeres XML-Dokument. Muss mit DestroyDocument() freigegeben werden.
`static void DeclareAllNamespacesFromSchema(ElementType& node)`	Deklariert alle Namespaces aus dem XML-Schema an dem als Argument angegebenen Element (normalerweise dem XML-Root-Element). Diese Methode eignet sich dann, wenn Ihr Schema mehrere Namespace-Deklarationen, von denen jedes auf ein Präfix gemappt ist, hat, und Sie alle davon für das als Argument angegebene Element deklarieren möchten.
`void DestroyDocument()`	Löscht ein Dokument. Alle Referenzen auf das Dokument und seine Nodes werden ungültig gemacht. Dieses Methode muss aufgerufen werden, wenn Sie mit einem Dokument fertig sind.
`static CDoc LoadFromBinary(const std:vector<unsigned char>& xml)`	Lädt ein XML-Dokument aus einem Byte Array.
`static CDoc LoadFromFile(const string_type& fileName)`	Lädt ein XML-Dokument aus einer Datei.
`static CDoc LoadFromString(const string_type& xml)`	Lädt ein XML-Dokument aus einem String.
`std::vector<unsigned char> SaveToBinary(bool prettyPrint)`	Speichert ein XML-Dokument in einem Byte-Array. Wenn das Argument `prettyPrint` auf "'true" gesetzt ist, wird das XML-Dokument aus Gründen der besseren Lesbarkeit neu formatiert.
`std::vector<unsigned char> SaveToBinary(bool prettyPrint, const string_type & encoding)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung in einem Byte-Array.

Name	Beschreibung
`std::vector<unsigned char> SaveToBinary(bool prettyPrint, const string_type & encoding, bool bBigEndian, bool bBOM)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung in einem Byte-Array. Für Unicode-Kodierungen können die Bytefolge und Unicode-Bytefolge-Markierung definiert werden.
`void SaveToFile(const string_type & fileName, bool prettyPrint)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung in einer Datei.
`void SaveToFile(const string_type & fileName, bool omitXmlDecl)`	Speichert ein XML-Dokument in einer Datei. Wenn das Argument `omitXmlDecl` auf "true" gesetzt ist, wird die XML-Deklaration nicht geschrieben.
`void SaveToFile(const string_type & fileName, bool omitXmlDecl, const string_type & encoding)`	Speichert ein XML-Dokument mit der angegebenen Kodierung in einer Datei. Wenn das Argument `omitXmlDecl` auf "true" gesetzt ist, wird die XML-Deklaration nicht geschrieben.
`void SaveToFile(const string_type & fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, bool bBigEndian, bool bBOM)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung in einer Datei. Für Unicode-Kodierungen können die Bytefolge und Unicode-Bytefolge-Markierung definiert werden.
`void SaveToFile(const string_type & fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, bool bBigEndian, bool bBOM, const string_type & lineend)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung und dem angegebenen Zeilenende in einer Datei. Für Unicode-Kodierungen können die Bytefolge und Unicode-Bytefolge-Markierung definiert werden. Diese Methode steht nur zur Verfügung, wenn Sie den Code für die Xerces3 XML-Bibliothek generiert haben (siehe Code Generator-Optionen 1254).
`void SaveToFile(const string_type& fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, const string_type & lineend)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung und dem angegebenen Zeilenende in einer Datei. Diese Methode steht nur zur Verfügung, wenn Sie den Code für die Xerces3 XML-Bibliothek generiert haben (siehe Code Generator-Optionen 1254).
`void SaveToFile(const string_type & fileName, bool prettyPrint, const string_type & encoding)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung in einer Datei.
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, bool bBigEndian, bool bBOM)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung in einer Datei. Für Unicode-Kodierungen können die Bytefolge und Unicode-Bytefolge-Markierung definiert werden.
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, bool`	Speichert ein XML-Dokument mit der definierten Kodierung und dem definierten Zeilenende in einer Datei. Für Unicode-

© 2016-2022 Altova GmbH

Name	Beschreibung
`bBigEndian, bool bBOM, const string_type & lineend)`	Kodierungen können die Bytefolge und Unicode-Bytefolge-Markierung definiert werden. Diese Methode steht nur zur Verfügung, wenn Sie den Code für die Xerces3 XML-Bibliothek generiert haben (siehe Code Generator-Optionen 1254).
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, const string_type & lineend)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung und dem angegebenen Zeilenende in einer Datei. Diese Methode steht nur zur Verfügung, wenn Sie den Code für die Xerces3 XML-Bibliothek generiert haben (siehe Code Generator-Optionen 1254).
`string_type SaveToString(bool prettyPrint)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung in einem String.
`string_type SaveToString(bool prettyPrint, bool omitXmlDecl)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung in einem String. Wenn das Argument `omitXmlDecl` auf "true" gesetzt ist, wird die XML-Deklaration nicht geschrieben.
`void SetDTDLocation(const string_type & dtdLocation)`	Fügt eine DOCTYPE-Deklaration mit der angegebenen System-ID hinzu. Es muss bereits ein Root-Element vorhanden sein. Diese Methode wird für MSXML nicht unterstützt, weil es nicht möglich ist, eine DOCTYPE-Deklaration zu einem Dokument im Arbeitsspeicher hinzuzufügen.
`void SetSchemaLocation(const string_type & schemaLocation)`	Fügt ein xsi:schemaLocation- oder xsi:noNamespaceSchemaLocation-Attribut zum Root-Element hinzu. Es muss bereits ein Root-Element vorhanden sein.

27.3.10 [YourSchema]::[ElementType]

Diese Klasse enthält Methoden zur Bearbeitung von XML-Elementen aus Ihrem Schema. Die Methoden dieser Klasse können für Elemente, nicht aber für das XML-Dokument selbst, aufgerufen werden. Beachten Sie, dass Sie die Klasse nicht direkt instantiieren müssen, um Methoden dieser Klasse aufzurufen. Jedes mit den Methoden **`append()`** oder **`appendWithPrefix()`** erstelltes Element hat den Typ `[ElementType]`.

Methoden

Name	Beschreibung
`void DeclareNamespace(const string_type prefix, const string_type nsURI)`	Diese Methode erhält zwei Argumente, beide vom Typ "String": das gewünschte Präfix und die gewünschte Namespace URI. Das als Argument bereitgestellte Präfix wird auf den als Argument bereitgestellten Namespace URI-Wert gemappt. Wenn das als Argument angegebene Präfix leer ist, erstellt die

Name	Beschreibung

	Methode die Standard-Namespace-Deklaration im Element oder setzt diese außer Kraft.
	
	Angenommen, das XML-Dokument hat ein XML-Element namens "purchase". Bei Aufruf von
	
	`purchase.DeclareNamespace(_T("ord"), _T("http://OrderTypes"));`
	
	wird das XML-Dokument zu
	
	`<purchase xmlns:ord="http://OrderTypes" />`
	
	Ein weiteres Beispiel: Bei Aufruf von
	
	`purchase.DeclareNamespace(_T(""), _T("http://OrderTypes"));`
	
	wird das XML-Dokument zu
	
	`<purchase xmlns="http://OrderTypes" />`
	
	Anmerkung: Der deklarierte Namespace wird beim Anhängen danach folgender Child-Elemente oder Attribute nach den folgenden Regeln verwendet:
	
	1. Wenn der Child-Namespace der Standard-Namespace ist, wird ein leeres Präfix verwendet.
	2. Wenn der Child-Namespace der gleiche ist wie der Parent Namespace ist, wird das Parent-Präfix verwendet.
	3. Anderfalls wird unter Verwendung des in Abschnitt "B.2: Namespace Prefix Lookup" unter https://www.w3.org/TR/2002/WD-DOM-Level-3-Core-20021022/namespaces-algorithms.html beschriebenen Lookup-Algorithmus nach dem nächstgelegenen Präfix vom Parent bis nach oben gesucht .
	4. Wenn kein Präfix für einen Element-Namespace gefunden wird, wird ein leeres Präfix verwendet.

27.3.11 [YourSchema]::MemberAttribute

Wenn anhand eines XML-Schemas Code generiert wird, wird für jedes Member-Attribut eines Typs eine Klasse wie diese erstellt.

© 2016-2022 Altova GmbH

Methoden

Name	Beschreibung
`bool exists()`	Gibt den Wert "true" zurück, wenn das Attribut vorhanden ist
`int GetEnumerationValue()`	Wird nur für Enumeration Types generiert. Gibt eine der für die möglichen Werte generierten Konstanten oder "Invalid" zurück, wenn der Wert mit keinem der Enumerationswerte im Schema übereinstimmt.
`altova::meta::Attribute info()`	Gibt ein Objekt zum Abfragen von Schemainformationen zurück (siehe `altova::meta::Attribute`1212).
`void remove()`	Entfernt das Attribut aus dem übergeordneten Element.
`void SetEnumerationValue(int)`	Wird nur für Enumeration Types generiert. Übergibt eine der für die möglichen Werte generierten Konstanten an diese Methode um den Wert zu definieren.

27.3.12 [YourSchema]::MemberElement

Wenn anhand eines XML-Schemas Code generiert wird, wird für jedes Member-Element eines Typs eine Klasse wie diese erstellt. In den Beschreibungen unten steht "MemberType" für den Namen des Member-Elements selbst.

Methoden

Name	Beschreibung
`Iterator<MemberType> all()`	Gibt ein Objekt für iterierende Instanzen des Member Elements zurück.
`MemberType append()`	Erstellt ein neues Element und hängt es an das übergeordnete Element an.
`MemberType appendWithPrefix(string_type prefix)`	Erstellt ein neues Element mit dem als Argument angegebenen Präfix und hängt es an seinen Parent an. Ein Beispiel dazu finden Sie unter Beispiel: Bestellung1197.
`unsigned int count()`	Gibt die Anzahl der Elemente zurück.
`int GetEnumerationValue()`	Wird nur für Enumeration Types generiert. Gibt eine der für die möglichen Werte generierten Konstanten zurück oder **Invalid**, wenn der Wert mit keinem der Enumerationswerte im Schema übereinstimmt.
`bool exists()`	Gibt den Wert "true" zurück, wenn mindestens ein Element vorhanden ist.
`MemberType first()`	Gibt die erste Instanz des Member Elements zurück.

Name	Beschreibung
`MemberType operator[](unsigned int index)`	Gibt das vom Index definierte Member Element zurück.
`altova::meta::Element info()`	Gibt ein Objekt zum Abfragen von Schemainformationen zurück (siehe `altova::meta::Element`[1213]).
`MemberType last()`	Gibt die letzte Instanz des Member Elements zurück.
`void remove()`	Löscht alle Instanzen des Elements aus dem übergeordneten Element.
`void removeAt(unsigned int index)`	Löscht die Instanz des durch den Index definierten Elements.
`void SetEnumerationValue(int)`	Wird nur für Enumeration Types generiert. Übergeben Sie eine der für die möglichen Werte generierten Konstanten an diese Methode, um den Wert zu definieren.

© 2016-2022 Altova GmbH

27.4 Referenz zu generierten Klassen (C#)

Dieses Kapitel enthält eine Beschreibung der mit XMLSpy anhand einer DTD oder eines XML-Schemas (siehe Generieren von Code anhand von XML-Schemas oder DTDs[1160]) generierten C#-Klassen. Sie können diese Klassen in Ihren Code integrieren, um XML-Dokumente zu lesen, zu bearbeiten und zu schreiben.

Anmerkung: Der generierte Code enthält andere unterstützende Klassen, die hier nicht aufgelistet sind und Änderungen unterworfen sind.

27.4.1 Altova.Types.DateTime

Mit Hilfe dieser Klasse können XML-Attribute oder -Elemente, die Datums- und Uhrzeittypen wie z.B. `xs:dateTime` haben, verarbeitet werden.

Konstruktoren

	Name	Beschreibung
	`DateTime(DateTime obj)`	Initialisiert eine neue Instanz der `DateTime`-Klasse auf das als Argument bereitgestellte `DateTime`-Objekt.
	`DateTime(System.DateTime newvalue)`	Initialisiert eine neue Instanz der `DateTime`-Klasse auf das als Argument bereitgestellte `System.DateTime`.
	`DateTime(int year, int month, int day, int hour, int minute, double second, int offsetTZ)`	Initialisiert eine neue Instanz der `DateTime`-Klasse auf die als Argumente bereitgestellten Werte für Jahr, Monat, Tag, Stunde, Minute, Sekunde und Zeitzone.
	`DateTime(int year, int month, int day, int hour, int minute, double second)`	Initialisiert eine neue Instanz der `DateTime`-Klasse auf die als Argumente bereitgestellten Werte für Jahr, Monat, Tag, Stunde, Minute und Sekunde.
	`DateTime(int year, int month, int day)`	Initialisiert eine neue Instanz der `DateTime`-Klasse auf die als Argumente bereitgestellten Werte für Jahr, Monat und Tag.

Eigenschaften

	Name	Beschreibung
	`bool HasTimezone`	Ruft einen Booleschen Wert ab, der angibt, ob `DateTime` eine Zeitzone hat.
	`static DateTime Now`	Ruft ein `DateTime` Objekt ab, das auf das aktuelle Datum und die aktuelle Uhrzeit dieses Computers gesetzt ist.
	`short TimezoneOffset`	Ruft den Zeitzonenunterschied des `DateTime`-Objekts in Minuten ab bzw. definiert ihn.

	Name	Beschreibung
	System.DateTime Value	Ruft den Wert des DateTime -Objekts als System.DateTime-Wert ab bzw. definiert ihn als solchen.

Methoden

	Name	Beschreibung
	int CompareTo(object obj)	Die DateTime-Klasse implementiert die IComparable-Schnittstelle. Diese Methode vergleicht die aktuelle Instanz von DateTime mit einem anderen Objekt und gibt eine Ganzzahl zurück, die angibt, ob die aktuelle Instanz in der Sortierreihenfolge vor, nach oder an derselben Position wie das andere Objekt vorkommt. Siehe auch https://msdn.microsoft.com/en-us/library/system.icomparable.compareto(v=vs.110).aspx
	override bool Equals(object obj)	Gibt **true** zurück, wenn das angegebene Objekt gleich dem aktuellen Objekt ist; gibt andernfalls **false** zurück.
	System.DateTime GetDateTime(bool correctTZ)	Gibt ein System.DateTime-Objekt anhand der aktuellen Altova.Types.DateTime-Instanz zurück. Das Boolesche Argument correctTZ gibt an, ob die Uhrzeit des zurückgegebenen Objekts entsprechend der Zeitzone der aktuellen Altova.Types.DateTime-Instanz angepasst werden muss.
	override int GetHashCode()	Gibt den Hash-Code der aktuellen Instanz zurück.
	int GetWeekOfMonth()	Gibt die Nummer der Woche im Monat als Ganzzahl zurück.
	static DateTime Parse(string s)	Erstellt anhand des als Argument bereitgestellten String ein DateTime-Objekt. So würden die folgenden Beispiel-Stringwerte etwa erfolgreich in ein DateTime-Objekt konvertiert: 2015-01-01T23:23:23 2015-01-01 2015-11 23:23:23 Wenn der String nicht in ein DateTime-Objekt konvertiert werden kann, wird eine Ausnahme ausgegeben. Beachten Sie, dass es sich hierbei um eine statische Methode handelt, die nur in der Altova.Types.DateTime-Klasse selbst und nicht in einer Instanz der Klasse aufgerufen werden kann.
	static DateTime Parse(string s, DateTimeFormat format)	Erstellt anhand des als Argument bereitgestellten Formats ein DateTime-Objekt. Ein Liste der möglichen

© 2016-2022 Altova GmbH

	Name	Beschreibung
		Formate finden Sie unter Altova.Types.DateTimeFormat 1224. Wenn der String nicht in ein `DateTime`-Objekt konvertiert werden kann, wird eine Ausnahme ausgegeben. Beachten Sie, dass es sich hierbei um eine statische Methode handelt, die nur in der `Altova.Types.DateTime`-Klasse selbst und nicht in einer Instanz der Klasse aufgerufen werden kann.
	`override string ToString()`	Konvertiert das `DateTime`-Objekt in einen String.
	`string ToString(DateTimeFormat format)`	Konvertiert das `DateTime`-Objekt anhand des als Argument bereitgestellten Formats in einen String. Ein Liste der möglichen Formate finden Sie unter Altova.Types.DateTimeFormat 1224.

Operatoren

Name	Beschreibung
`!=`	Ermittelt, ob `DateTime` a ungleich `DateTime` b ist.
`<`	Ermittelt, ob `DateTime` a kleiner als `DateTime` b ist.
`<=`	Ermittelt, ob `DateTime` a kleiner oder gleich `DateTime` b ist.
`==`	Ermittelt, ob `DateTime` a gleich `DateTime` b ist.
`>`	Ermittelt, ob `DateTime` a größer als `DateTime` b ist.
`>=`	Ermittelt, ob `DateTime` a größer oder gleich `DateTime` b ist.

Beispiele

Stellen Sie sicher, dass die Altova-Typen importiert wurden, bevor Sie die folgenden Codefragmente in Ihrem Programm verwenden.

```
using Altova.Types;
```

In den folgenden Codefragmenten werden verschiedene Möglichkeiten zur Erstellung von `DateTime`-Objekten gezeigt:

```
protected static void DateTimeExample1()
{
	// Create a DateTime object from the current system time
	Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);
	Console.WriteLine("The current time is: " + dt.ToString());
```

```
      // Create an Altova DateTime object from parts (no timezone)
      Altova.Types.DateTime dt1 = new Altova.Types.DateTime(2015, 10, 12, 10, 50, 33);
      Console.WriteLine("My custom time is : " + dt1.ToString());

      // Create an Altova DateTime object from parts (with UTC+60 minutes timezone)
      Altova.Types.DateTime dt2 = new Altova.Types.DateTime(2015, 10, 12, 10, 50, 33,
60);
      Console.WriteLine("My custom time with timezone is : " + dt2.ToString());

      // Create an Altova DateTime object by parsing a string
      Altova.Types.DateTime dt3 = Altova.Types.DateTime.Parse("2015-01-01T23:23:23");
      Console.WriteLine("Time created from string: " + dt3.ToString());

      // Create an Altova DateTime object by parsing a string formatted as schema date
      Altova.Types.DateTime dt4 = Altova.Types.DateTime.Parse("2015-01-01",
DateTimeFormat.W3_date);
      Console.WriteLine("Time created from string formatted as schema date: " +
dt4.ToString());
}
```

In den folgenden Codefragmenten werden verschiedene Möglichkeiten zur Formatierung von `DateTime`-Objekten gezeigt:

```
protected static void DateTimeExample2()
{
      // Create a DateTime object from the current system time
      Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);

      // Output the unformatted DateTime
      Console.WriteLine("Unformatted time: " + dt.ToString());

      // Output this DateTime formatted using various formats
      Console.WriteLine("S_DateTime:       " + dt.ToString(DateTimeFormat.S_DateTime));
      Console.WriteLine("S_Days:           " + dt.ToString(DateTimeFormat.S_Days));
      Console.WriteLine("S_Seconds:        " + dt.ToString(DateTimeFormat.S_Seconds));
      Console.WriteLine("W3_date:          " + dt.ToString(DateTimeFormat.W3_date));
      Console.WriteLine("W3_dateTime:      " + dt.ToString(DateTimeFormat.W3_dateTime));
      Console.WriteLine("W3_gDay:          " + dt.ToString(DateTimeFormat.W3_gDay));
      Console.WriteLine("W3_gMonth:        " + dt.ToString(DateTimeFormat.W3_gMonth));
      Console.WriteLine("W3_gMonthDay:     " +
dt.ToString(DateTimeFormat.W3_gMonthDay));
      Console.WriteLine("W3_gYear:         " + dt.ToString(DateTimeFormat.W3_gYear));
      Console.WriteLine("W3_gYearMonth:    " +
dt.ToString(DateTimeFormat.W3_gYearMonth));
      Console.WriteLine("W3_time:          " + dt.ToString(DateTimeFormat.W3_time));
}
```

27.4.2 Altova.Types.DateTimeFormat

Der `DateTimeFormat` enum-Typ hat die folgenden Konstantenwerte:

© 2016-2022 Altova GmbH

Wert	Beschreibung	Beispiel
S_DateTime	Formatiert den Wert mit einer Genauigkeit einer zehnmillionstel Sekunde einschließlich Zeitzone als Standard-DateTime-Wert.	`2015-11-12 12:19:03.9019132+01:00`
S_Days	Formatiert den Wert als Anzahl der seit der UNIX-Epoche verstrichenen Tage.	`735913.631897345108796296 2963`
S_Seconds	Formatiert den Wert mit einer Genauigkeit einer Zehnmillionstelsekunde als Anzahl der seit der UNIX-Epoche verstrichenen Sekunden.	`63582937678.0769062`
W3_date	Formatiert den Wert als Schema-Datum	`2015-11-12`
W3_dateTime	Formatiert den Wert als Schema-DateTime (Datum und Uhrzeit).	`2015-11-12T15:12:14.5194251`
W3_gDay	Formatiert den Wert als Schema-gDay.	`---12` (vorausgesetzt, das Datum ist der 12. des Monats)
W3_gMonth	Formatiert den Wert als Schema-gMonth.	`--11` (vorausgesetzt, der Monat ist November)
W3_gMonthDay	Formatiert den Wert als Schema-gMonthDay.	`--11-12` (vorausgesetzt, das Datum ist der 12. November)
W3_gYear	Formatiert den Wert als Schema-gYear.	`2015` (vorausgesetzt das Jahr ist 2015)
W3_gYearMonth	Formatiert den Wert als Schema-gYearMonth.	`2015-11` (vorausgesetzt das Jahr ist 2015 und der Monat ist November)
W3_time	Formatiert den Wert mit der Genauigkeit einer Zehnmillionstelsekunde als Schema-Uhrzeit.	`15:19:07.5582719`

27.4.3 Altova.Types.Duration

Die Klasse `Altova.Types.Duration` stellt die folgenden öffentlichen Mitglieder bereit.

Konstruktoren

	Name	Beschreibung
	`Duration(Duration obj)`	Initialisiert eine neue Instanz der `Duration`-Klasse auf das als Argument bereitgestellte `Duration`-Objekt.
	`Duration(System.TimeSpan newvalue)`	Initialisiert eine neue Instanz der `Duration`-Klasse auf das als Argument bereitgestellte `System.TimeSpan`-Objekt.
	`Duration(long ticks)`	Initialisiert eine neue Instanz der `Duration`-Klasse auf die Anzahl der als Argument bereitgestellten Zeiteinheiten.
	`Duration(int newyears, int newmonths, int days, int hours, int minutes, int seconds, double partseconds, bool bnegative)`	Initialisiert eine neue Instanz der `Duration`-Klasse auf eine Zeitdauer, die anhand von als Argumente bereitgestellten Bestandteilen erstellt wird.

Eigenschaften

	Name	Beschreibung
	`int Months`	Ruft die Anzahl der Monate der aktuellen Instanz von `Duration` ab oder definiert diese.
	`System.TimeSpan Value`	Ruft den Wert (als `System.TimeSpan`) der aktuellen Instanz von `Duration` ab oder definiert diesen.
	`int Years`	Ruft die Anzahl der Jahre der aktuellen Instanz von `Duration` ab oder definiert diese.

Methoden

	Name	Beschreibung
	`override bool Equals(object other)`	Gibt **true** zurück, wenn das angegebene Objekt gleich dem aktuellen Objekt ist; gibt andernfalls **false** zurück.
	`override int GetHashCode()`	Gibt den Hash-Code der aktuellen Instanz zurück.
	`bool IsNegative()`	Gibt **true** zurück, wenn die aktuelle Instanz von `Duration` eine negative Zeitdauer darstellt.

© 2016-2022 Altova GmbH

	Name	Beschreibung
	`static Duration Parse( string s, ParseType pt )`	Gibt anhand des als Argument bereitgestellten String ein `Altova.Types.Duration`-Objekt zurück, wobei der als Argument bereitgestellte Parse-Typ verwendet wird. Gültige Parse-Typwerte sind: **DURATION** Parse-Zeitdauer, wenn als Bestandteile der Zeitdauer Jahr, Monate, Tag sowie Uhrzeit vorhanden sind. **YEARMONTH** Parse-Zeitdauer, wenn nur die Teile Jahr und Monat vorhanden sind. **DAYTIME** Parse-Zeitdauer, wenn nur die Teile Tag und Uhrzeit vorhanden sind. Beachten Sie, dass es sich hierbei um eine statische Methode handelt, die nur in der Klasse selbst und nicht in einer Instanz der Klasse aufgerufen werden kann.
	`override string ToString()`	Konvertiert die aktuelle `Duration`-Instanz in einen String. So würde eine Zeitspanne von 3 Stunden, 4 Minuten und 5 Sekunden z.B. in "PT3H4M5S" konvertiert
	`string ToYearMonthString()`	Konvertiert die aktuelle `Duration`-Instanz anhand des Parse-Typs "Jahr und Monat" in einen String.

Operatoren

Name	Beschreibung
`!=`	Ermittelt, ob `Duration` a ungleich `Duration` b ist.
`==`	Ermittelt, ob `Duration` a gleich `Duration` b ist.

Beispiele

Stellen Sie sicher, dass die Altova-Typen importiert wurden, bevor Sie die folgenden Codefragmente in Ihrem Programm verwenden:

```
using Altova.Types;
```

In den folgenden Codefragmenten werden verschiedene Möglichkeiten zur Erstellung von `Duration`-Objekten gezeigt:

```
protected static void DurationExample1()
{
	// Create a new time span of 3 hours, 4 minutes, and 5 seconds
	System.TimeSpan ts = new TimeSpan(3, 4, 5);
	// Create a Duration from the time span
	Duration dr = new Duration(ts);
	// The output is: PT3H4M5S
```

```
       Console.WriteLine("Duration created from TimeSpan: " + dr.ToString());

       // Create a negative Altova.Types.Duration from 6 years, 5 months, 4 days, 3
hours,
       // 2 minutes, 1 second, and .33 of a second
       Duration dr1 = new Duration(6, 5, 4, 3, 2, 1, .33, true);
       // The output is: -P6Y5M4DT3H2M1.33S
       Console.WriteLine("Duration created from parts: " + dr1.ToString());

       // Create a Duration from a string using the DAYTIME parse type
       Duration dr2 = Altova.Types.Duration.Parse("-P4DT3H2M1S",
Duration.ParseType.DAYTIME);
       // The output is -P4DT3H2M1S
       Console.WriteLine("Duration created from string: " + dr2.ToString());

       // Create a duration from ticks
       Duration dr3 = new Duration(System.DateTime.UtcNow.Ticks);
       // Output the result
       Console.WriteLine("Duration created from ticks: " + dr3.ToString());
}
```

In den folgenden Codefragmenten wird gezeigt, wie Werte aus `Duration`-Objekten ermittelt werden können:

```
protected static void DurationExample2()
{
       // Create a negative Altova.Types.Duration from 6 years, 5 months, 4 days, 3
hours,
       // 2 minutes, 1 second, and .33 of a second
       Duration dr = new Duration(6, 5, 4, 3, 2, 1, .33, true);
       // The output is: -P6Y5M4DT3H2M1.33S
       Console.WriteLine("The complete duration is: " + dr.ToString());

       // Get only the year and month part as string
       string dr1 = dr.ToYearMonthString();
       Console.WriteLine("The YEARMONTH part is: " + dr1);

       // Get the number of years in duration
       Console.WriteLine("Years: " + dr.Years);

       // Get the number of months in duration
       Console.WriteLine("Months: " + dr.Months);
}
```

27.4.4 Altova.Xml.Meta.Attribute

Mit Hilfe dieser Klasse können Sie Schemainformationen über anhand von Attributen generierte Klassen aufrufen. Beachten Sie, dass diese Klasse nicht dazu gedacht ist, dynamische Informationen über bestimmte Instanzen eines Attributs in einem XML-Dokument bereitzustellen. Es dient stattdessen dazu, Informationen über ein bestimmtes im XML-Schema definiertes Attribut programmatisch abzurufen.

© 2016-2022 Altova GmbH

Eigenschaften

	Name	Beschreibung
	`SimpleType DataType`	Gibt den Typ des Attributinhalts zurück.
	`string LocalName`	Gibt den lokalen Namen des Attributs zurück.
	`string NamespaceURI`	Gibt die Namespace URI des Attributs zurück.
	`XmlQualifiedName QualifiedName`	Gibt den qualifizierten Namen des Attributs zurück.
	`bool Required()`	Gibt "true" zurück, wenn das Attribut obligatorisch ist.

27.4.5 Altova.Xml.Meta.ComplexType

Mit Hilfe dieser Klasse können Sie Schemainformationen über anhand von complexTypes generierte Klassen aufrufen. Beachten Sie, dass diese Klasse nicht dazu gedacht ist, dynamische Informationen über bestimmte Instanzen eines complexType in einem XML-Dokument bereitzustellen. Es dient stattdessen dazu, Informationen über einen bestimmten im XML-Schema definierten complexType programmatisch abzurufen.

Eigenschaften

	Name	Beschreibung
	`Attribute[] Attributes`	Gibt eine Liste aller Attribute zurück.
	`ComplexType BaseType`	Gibt den Basistyp dieses Typs zurück oder Null, wenn kein Basistyp vorhanden ist.
	`SimpleType ContentType`	Gibt dem simpleType des Inhalts zurück.
	`Element[] Elements`	Gibt eine Liste aller Elemente zurück.
	`string LocalName`	Gibt den lokalen Namen des Typs zurück.
	`string NamespaceURI`	Gibt die Namespace URI des Typs zurück.
	`XmlQualifiedName QualifiedName`	Gibt den qualifizierten Namen dieses Typs zurück.

Methoden

	Name	Beschreibung
	`ComplexType BaseType`	Gibt den Basistyp dieses Typs zurück.

	Name	Beschreibung
	`bool Equals(obj)`	Überprüft auf Basis eines Vergleichs des qualifizierten Namens, ob zwei Info-Objekte denselben Typ referenzieren. Gibt "true" zurück, wenn der Typ denselben qualifizierten Namen hat.
	`Attribute FindAttribute(string localName, string namespaceURI)`	Sucht das Attribut mit dem angegebenen lokalen Namen und der angegebenen Namespace URI.
	`Element FindElement(string localName, string namespaceURI)`	Sucht das Element mit dem angegebenen lokalen Namen und der angegebenen Namespace URI.

27.4.6 Altova.Xml.Meta.Element

Mit Hilfe dieser Klasse können Sie Informationen über anhand von Schema-Elementen generierte Klassen aufrufen. Beachten Sie, dass diese Klasse nicht dazu gedacht ist, dynamische Informationen über bestimmte Instanzen eines Elements in einem XML-Dokument bereitzustellen. Es dient stattdessen dazu, Informationen über ein bestimmtes im XML-Schema definiertes Element programmatisch abzurufen.

Eigenschaften

	Name	Description
	`ComplexType DataType`	Gibt den Typ des Elements zurück. Beachten Sie, dass dies immer ein complexType ist, selbst wenn er im Originalschema als simpleType deklariert wurde. Verwenden Sie die Eigenschaft `ContentType` des zurückgegebenen Objekts zum den simple content type abzurufen.
	`string` `LocalName`	Gibt den lokalen Namen des Elements zurück.
	`int` `MaxOccurs`	Gibt den im Schema definierten `maxOccurs`-Wert zurück.
	`int` `MinOccurs`	Gibt den im Schema definierten `minOccurs`-Wert zurück.
	`string` `NamespaceURI`	Gibt die Namespace URI des Elements zurück.
	`XmlQualifiedName QualifiedName`	Gibt den qualifizierten Namen des Elements zurück.

27.4.7 Altova.Xml.Meta.SimpleType

Mit Hilfe dieser Klasse können Sie Schemainformationen über anhand von simpleTypes generierte Klassen aufrufen. Beachten Sie, dass diese Klasse nicht dazu gedacht ist, dynamische Informationen über bestimmte Instanzen eines simpleType in einem XML-Dokument bereitzustellen. Es dient stattdessen dazu, Informationen über einen bestimmten im XML-Schema definierten simpleTypes programmatisch abzurufen.

© 2016-2022 Altova GmbH

Eigenschaften

	Name	Beschreibung
	`SimpleType BaseType`	Gibt den Basistyp dieses Typs zurück.
	`string``[] Enumerations`	Gibt eine Liste aller Enumeration Facets zurück.
	`int` `FractionDigits`	Gibt den Wert dieses Facet zurück.
	`int` `Length`	Gibt den Wert dieses Facet zurück.
	`string` `LocalName`	Gibt den lokalen Namen des Tys zurück.
	`string` `MaxExclusive`	Gibt den Wert dieses Facet zurück.
	`string` `MaxInclusive`	Gibt den Wert dieses Facet zurück.
	`int` `MaxLength`	Gibt den Wert dieses Facet zurück.
	`string` `MinExclusive`	Gibt den Wert dieses Facet zurück.
	`string` `MinInclusive`	Gibt den Wert dieses Facet zurück.
	`int` `MinLength`	Gibt den Wert dieses Facet zurück.
	`string` `NamespaceURI`	Gibt die Namespace URI des Typs zurück.
	`string``[] Patterns`	Gibt die Pattern Facets oder Null zurück, wenn keine Patterns definiert sind.
	`XmlQualifiedName QualifiedName`	Gibt den qualifizierten Namens dieses Typs zurück.
	`int` `TotalDigits`	Gibt den Wert dieses Facet zurück.
	`WhitespaceType Whitespace`	Gibt das Whitespace Normalization Facet zurück.

27.4.8 [YourSchema].[Doc]

Wenn anhand eines XML-Schemas Code generiert wird, stellt der generierte Code eine Dokumentklasse mit demselben Namen wie dem des Schemas bereit. Diese Klasse enthält alle möglichen Root-Elemente als Members sowie die folgenden Members. Beachten Sie, dass "Doc" in den unten aufgelisteten Methodennamen für den Namen der generierten Dokumentklasse selbst steht.

Methoden

	Name	Beschreibung
	`static` `Doc CreateDocument()`	Erstellt ein neues leeres XML-Dokument.

	Name	Beschreibung
	`static Doc CreateDocument(string encoding)`	Erstellt ein neues leeres XML-Dokument mit Kodierung des Typs "encoding".
	`static void DeclareAllNamespacesFromSchema(Altova.Xml.ElementType node)`	Deklariert alle Namespaces aus dem XML-Schema an dem als Argument angegebenen Element (normalerweise dem XML-Root-Element). Diese Methode eignet sich dann, wenn Ihr Schema mehrere Namespace-Deklarationen, von denen jedes auf ein Präfix gemappt ist, hat, und Sie alle davon für das als Argument angegebene Element deklarieren möchten.
	`static Doc LoadFromBinary(byte[] binary)`	Lädt ein XML-Dokument aus einem Byte Array.
	`static Doc LoadFromFile(string filename)`	Lädt ein XML-Dokument aus einer Datei.
	`static Doc LoadFromString(string xmlstring)`	Lädt ein XML-Dokument aus einem String.
	`byte[] SaveToBinary(bool prettyPrint)`	Speichert ein XML-Dokument mit optionaler Pretty-Print-Formatierung in einem Byte Array.
	`byte[] SaveToBinary(bool prettyPrint, string encoding)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung in einem Byte-Array.
	`byte[] SaveToBinary(bool prettyPrint, string encoding, bool bBigEndian, bool bBOM)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung, Bytefolge und BOM (Byte Order Mark) in einem Byte-Array.
	`void SaveToFile(string fileName, bool prettyPrint)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung in einer Datei.
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung in einer Datei. Wenn `omitXmlDecl` auf "true" gesetzt ist, wird die XML-Deklaration nicht geschrieben.
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl, string encoding)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung in einer Datei. Wenn `omitXmlDecl` auf "true" gesetzt ist, wird die XML-Deklaration nicht geschrieben.
	`void SaveToFile(string fileName, bool prettyPrint, string encoding, string lineend)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung und dem angegebenen Zeilenende in einer Datei.
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl, string encoding, string lineend)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung und dem angegebenen Zeilenende in einer Datei. Wenn `omitXmlDecl` auf "true" gesetzt ist, wird die XML-Deklaration nicht geschrieben.

© 2016-2022 Altova GmbH

	Name	Beschreibung
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl, string encoding, bool bBigEndian, bool bBOM, string lineend)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung, Bytefolge, BOM (Byte Order Mark) und dem definierten Zeilenende in einer Datei. Wenn `omitXmlDecl` auf "true" gesetzt ist, wird die XML-Deklaration nicht geschrieben.
	`void SaveToFileWithLineEnd(string fileName, bool prettyPrint, bool omitXmlDecl, string lineend)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung und dem definierten Zeilenende in einer Datei. Wenn `omitXmlDecl` auf "true" gesetzt ist, wird die XML-Deklaration nicht geschrieben.
	`string SaveToString(bool prettyPrint)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung in einer Datei.
	`string SaveToString(bool prettyPrint, bool omitXmlDecl)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung in einer Datei. Wenn `omitXmlDecl` auf "true" gesetzt ist, wird die XML-Deklaration nicht geschrieben.
	`void SetDTDLocation(string dtdLocation)`	Fügt eine DOCTYPE-Deklaration mit der angegebenen System-ID hinzu. Es muss bereits ein Root-Element vorhanden sein.
	`void SetSchemaLocation(string schemaLocation)`	Fügt ein xsi:schemaLocation- oder xsi:noNamespaceSchemaLocation-Attribut zum Root-Element hinzu. Es muss bereits ein Root-Element vorhanden sein.

27.4.9 [YourSchema].[ElementType]

Diese Klasse enthält Methoden zur Bearbeitung von XML-Elementen aus Ihrem Schema. Die Methoden dieser Klasse können für Elemente, nicht aber für das XML-Dokument selbst, aufgerufen werden. Beachten Sie, dass Sie die Klasse nicht direkt instantiieren müssen, um Methoden dieser Klasse aufzurufen. Jedes mit den Methoden **`Append()`** oder **`AppendWithPrefix()`** erstelltes Element hat den Typ `[ElementType]`.

Methoden

	Name	Beschreibung
	`void DeclareNamespace(string prefix, string nsURI)`	Diese Methode erhält zwei Argumente, beide vom Typ "String": das gewünschte Präfix und die gewünschte Namespace URI. Das als Argument bereitgestellte Präfix wird auf den als Argument bereitgestellten Namespace URI-Wert gemappt. Wenn das als Argument angegebene Präfix leer ist, erstellt die Methode die Standard-Namespace-Deklaration im Element oder setzt diese außer Kraft. Angenommen, das XML-Dokument hat ein XML-Element namens "purchase". Bei Aufruf von

	Name	Beschreibung

		`purchase.DeclareNamespace("ord", "http://OrderTypes");`
		
		wird das XML-Dokument zu
		
		`<purchase xmlns:ord="http://OrderTypes" />`
		
		Ein weiteres Beispiel: Bei Aufruf von
		
		`purchase.DeclareNamespace("", "http://OrderTypes");`
		
		wird das XML-Dokument zu
		
		`<purchase xmlns="http://OrderTypes" />`
		
		Anmerkung: Der deklarierte Namespace wird beim Anhängen danach folgender Child-Elemente oder Attribute nach den folgenden Regeln verwendet:
		
		1. Wenn der Child-Namespace der Standard-Namespace ist, wird ein leeres Präfix verwendet.
		2. Wenn der Child-Namespace der gleiche ist wie der Parent Namespace ist, wird das Parent-Präfix verwendet.
		3. Anderfalls wird unter Verwendung des in Abschnitt "B.2: Namespace Prefix Lookup" unter https://www.w3.org/TR/2002/WD-DOM-Level-3-Core-20021022/namespaces-algorithms.html beschriebenen Lookup-Algorithmus nach dem nächstgelegenen Präfix vom Parent bis nach oben gesucht .
		4. Wenn kein Präfix für einen Element-Namespace gefunden wird, wird ein leeres Präfix verwendet.

27.4.10 [YourSchemaType].MemberAttribute

Wenn anhand eines XML-Schemas Code generiert wird, wird für jedes Member-Attribut eines Typs eine Klasse wie diese erstellt. In der Beschreibung unten steht "AttributeType" für den Typ des Member-Attributs selbst.

Methoden

	Name	Beschreibung
	`bool Exists()`	Gibt den Wert "true" zurück, wenn das Attribut vorhanden ist.

© 2016-2022 Altova GmbH

	Name	Beschreibung
	`void Remove()`	Entfernt das Attribut aus dem übergeordneten Element.

Properties

	Name	Beschreibung
	`int` `EnumerationValue`	Wird nur für Enumeration Types generiert. Definiert den Attributwert mittels einer der für die möglichen Werte generierten Konstanten bzw. definiert diesen. Gibt **Invalid** zurück, wenn der Wert mit keinem der enumerierten Werte im Schema übereinstimmt.
	`Altova.Xml.Meta.Attribute Info`	Gibt ein Objekt zum Abfragen von Schemainformationen zurück (siehe `Altova.Xml.Meta.Attribute` 1228).
	`AttributeType Value`	Definiert den Attributwert bzw. ruft ihn ab.

27.4.11 [YourSchemaType].MemberElement

Wenn anhand eines XML-Schemas Code generiert wird, wird für jedes Member-Element eines Typs eine Klasse mit den folgenden Menbers erstellt. Die Klasse implementiert die `System.Collections.IEnumerable`-Standardschnittstelle, daher kann sie mit der `foreach`-Anweisung verwendet werden.

In den Beschreibungen unten steht "MemberType" für den Typ des Member-Elements selbst.

Methoden

	Name	Beschreibung
	`MemberType Append()`	Erstellt ein neues Element und hängt es an das übergeordnete Element an.
	`MemberType AppendWithPrefix(string prefix)`	Erstellt ein neues Element mit dem als Argument angegebenen Präfix und hängt es an seinen Parent an. Ein Beispiel dazu finden Sie unter Beispiel: Bestellung 1197.
	`MemberType At(int index)`	Gibt das vom Index definierte Member Element zurück.
	`System.Collections.IEnumerator GetEnumerator()`	Gibt ein Objekt für iterierende Instanzen des Member Elements zurück.
	`void` `Remove()`	Löscht alle Instanzen des Elements aus dem übergeordneten Element.
	`void` `RemoveAt(int index)`	Löscht die Instanz des durch den Index definierten Elements.

Eigenschaften

	Name	Beschreibung
	`int Count`	Gibt die Anzahl der Elemente zurück.
	`int EnumerationValue`	Wird nur für Enumeration Types generiert. Definiert den Elementwert mittels einer der für die möglichen Werte generierten Konstanten bzw. definiert diesen. Gibt **Invalid** zurück, wenn der Wert mit keinem der enumerierten Werte im Schema übereinstimmt.
	`bool Exists`	Gibt den Wert "true" zurück, wenn mindestens ein Element vorhanden ist.
	`MemberType First`	Gibt die erste Instanz des Member Elements zurück.
	`Altova.Xml.Meta.Element Info`	Gibt ein Objekt zum Abfragen von Schemainformationen zurück (siehe `Altova.Xml.Meta.Element` 1230).
	`MemberType Last`	Gibt die letzte Instanz des Member Elements zurück.
	`MemberType this[int index]`	Gibt das vom Index definierte Member Element zurück.
	`MemberType Value`	Definiert den Elementinhalt bzw. ruft ihn ab (wird nur generiert, wenn das Element mixed oder simple content haben kann).

© 2016-2022 Altova GmbH

27.5 Referenz zu generierten Klassen (Java)

Dieses Kapitel enthält eine Beschreibung der mit XMLSpy anhand einer DTD oder eines XML-Schemas (siehe Generieren von Code anhand von XML-Schemas oder DTDs[1160]) generierten Java-Klassen. Sie können diese Klassen in Ihren Code integrieren, um XML-Dokumente zu lesen, zu bearbeiten und zu schreiben.

Anmerkung: Der generierte Code enthält andere unterstützende Klassen, die hier nicht aufgelistet sind und Änderungen unterworfen sind.

27.5.1 com.altova.types.DateTime

Mit Hilfe dieser Klasse können XML-Attribute oder -Elemente, die Datums- und Uhrzeittypen wie z.B. `xs:dateTime` haben, verarbeitet werden.

Konstruktoren

	Name	Beschreibung
C	`public DateTime()`	Initialisiert eine neue Instanz der `DateTime`-Klasse auf einen leeren Wert.
C	`public DateTime(DateTime newvalue)`	Initialisiert eine neue Instanz der `DateTime`-Klasse auf den als Argument bereitgestellten `DateTime`-Wert.
C	`public DateTime(int newyear, int newmonth, int newday, int newhour, int newminute, int newsecond, double newpartsecond, int newoffsetTZ)`	Initialisiert eine neue Instanz der `DateTime`-Klasse auf das Jahr, den Monat, den Tag, die Stunde, Minute, Sekunde, den Sekundenbruchteil und die Zeitzone, die als Argumente bereitgestellt wurden. Der Sekundenbruchteil newpartsecond muss zwischen 0 und 1 sein. Die Zeitzonendifferenz newoffsetTZ kann entweder positiv oder negativ sein und wird in Minuten ausgedrückt.
C	`public DateTime(int newyear, int newmonth, int newday, int newhour, int newminute, int newsecond, double newpartsecond)`	Initialisiert eine neue Instanz der `DateTime`-Klasse auf das Jahr, den Monat, den Tag, die Stunde, Minute, Sekunde und den Sekundenbruchteil, die als Argumente bereitgestellt wurden.
C	`public DateTime(int newyear, int newmonth, int newday)`	Initialisiert eine neue Instanz der `DateTime`-Klasse auf das Jahr, den Monat, den Tag, die als Argumente bereitgestellt wurden.
C	`public DateTime(Calendar newvalue)`	Initialisiert eine neue Instanz der `DateTime`-Klasse auf den als Argument bereitgestellten `java.util.Calendar`-Wert.

Methoden

	Name	Beschreibung
S	`static DateTime now()`	Gibt die aktuelle Uhrzeit als `DateTime`-Objekt zurück.

	Name	Beschreibung
S	**static** DateTime parse(String s)	Gibt ein DateTime-Objekt zurück, das aus dem als Argument bereitgestellten String-Werts geparst wird. Der folgende Beispiel-String würde erfolgreich in ein DateTime-Objekt konvertiert werden. 2015-11-24T12:54:47.969+01:00 2015-11-24T12:54:47 2015-11-24
S	**int** getDay()	Gibt den Tag der aktuellen DateTime-Instanz zurück.
S	**int** getHour()	Gibt die Stunde der aktuellen DateTime-Instanz zurück.
S	**int** getMillisecond()	Gibt die Millisekunde der aktuellen DateTime-Instanz als Ganzzahlwert zurück.
S	**int** getMinute()	Gibt die Minute der aktuellen DateTime-Instanz zurück.
S	**int** getMonth()	Gibt den Monat der aktuellen DateTime-Instanz zurück.
S	**double** getPartSecond()	Gibt die Nachkommastellen der Sekunde der aktuellen DateTime-Instanz als **double** -Wert zurück. Der Rückgabewert ist größer als Null und kleiner als eins, z.B.: 0,313
S	**int** getSecond()	Gibt die Sekunde der aktuellen DateTime-Instanz zurück.
S	**int** getTimezoneOffset()	Gibt den Zeitzonenunterschied zur aktuellen DateTime-Instanz in Minuten zurück. Die Zeitzone "UTC-01:00" würde z.B. als folgender Wert zurückgegeben: -60
S	Calendar getValue()	Gibt die aktuelle DateTime Instanz als java.util.Calendar-Wert zurück.
S	**int** getWeekday()	Gibt den Wochentag der aktuellen DateTime-Instanz zurück. Die Werte liegen im Bereich von 0 bis 6, wobei 0 Montag ist (ISO-8601).
S	**int** getYear()	Gibt das Jahr der aktuellen DateTime-Instanz zurück.
S	**int** hasTimezone()	Gibt Informationen über die Zeitzone der aktuellen DateTime-Instanz zurück. Mögliche Rückgabewerte sind: CalendarBase.TZ_MISSING — Es ist kein Zeitzonenunterschied definiert. CalendarBase.TZ_UTC — Die Zeitzone ist UTC.

© 2016-2022 Altova GmbH

	Name	Beschreibung
		`CalendarBase.TZ_OFFSET` Es ist ein Zeitzonenunterschied definiert.
S	**void** `setDay( int nDay )`	Setzt den Tag der aktuellen `DateTime`-Instanz auf den als Argument bereitgestellten Wert.
S	**void** `setHasTimezone( int nHasTZ )`	Setzt die Zeitzoneninformation der aktuellen `DateTime`-Instanz auf den als Argument bereitgestellten Wert. Mit Hilfe dieser Methode können die Zeitzoneninformationen entfernt werden oder die Zeitzone kann auf UTC (Coordinated Universal Time) gesetzt werden. Gültige Werte für das nHasTZ-Argument: `CalendarBase.TZ_MISSING` Setzt die Zeitzone auf nicht definiert. `CalendarBase.TZ_UTC` Setzt die Zeitzone auf UTC. `CalendarBase.TZ_OFFSET` Wenn das aktuelle Objekt einen Zeitzonenunterschied hat, so bleibt dieser unverändert.
S	**void** `setHour( int nHour )`	Setzt die Stunde der aktuellen `DateTime`-Instanz auf den als Argument bereitgestellten Wert.
S	**void** `setMinute( int nMinute )`	Setzt die Minute der aktuellen `DateTime`-Instanz auf den als Argument bereitgestellten Wert.
S	**void** `setMonth( int nMonth )`	Setzt den Monat der aktuellen `DateTime`-Instanz auf den als Argument bereitgestellten Wert.
S	**void** `setPartSecond( double nPartSecond )`	Setzt die Sekundenbruchteile der aktuellen `DateTime`-Instanz auf den als Argument bereitgestellten Wert.
S	**void** `setSecond( int nSecond )`	Setzt die Sekunde der aktuellen `DateTime`-Instanz auf den als Argument bereitgestellten Wert.
S	**void** `setTimezoneOffset( int nOffsetTZ )`	Setzt den Zeitzonenunterschied der aktuellen `DateTime`-Instanz auf den als Argument bereitgestellten Wert. Der Wert nOffsetTZ muss eine (positive oder negative) Ganzzahl sein und in Minuten ausgedrückt werden.
S	**void** `setYear( int nYear )`	Setzt das Jahr der aktuellen `DateTime`-Instanz auf den als Argument bereitgestellten Wert.
S	`String toString()`	Gibt die String-Darstellung der aktuellen `DateTime`-Instanz zurück, z.B.: `2015-11-24T15:50:56.968+01:00`

Beispiele

Stellen Sie sicher, dass die Altova-Typen importiert wurden, bevor Sie die folgenden Codefragmente in Ihrem Programm verwenden.

```
import com.altova.types.*;
```

In den folgenden Codefragmenten werden verschiedene Möglichkeiten zur Erstellung von `DateTime`-Objekten gezeigt:

```
protected static void DateTimeExample1()
{
   // Initialize a new instance of the DateTime class to the current time
   DateTime dt = new DateTime(DateTime.now());
   System.out.println("DateTime created from current date and time: " + dt.toString());

   // Initialize a new instance of the DateTime class by supplying the parts
   DateTime dt1 = new DateTime(2015, 11, 23, 14, 30, 24, .459);
   System.out.println("DateTime from parts (no timezone): " + dt1.toString());

   // Initialize a new instance of the DateTime class by supplying the parts
   DateTime dt2 = new DateTime(2015, 11, 24, 14, 30, 24, .459, -60);
   System.out.println("DateTime from parts (with negative timezone): " + dt2.toString());

   // Initialize a new instance of the DateTime class by parsing a string value
   DateTime dt3 = DateTime.parse("2015-11-24T12:54:47.969+01:00");
   System.out.println("DateTime parsed from string: " + dt3.toString());
}
```

Im folgenden Codefragment wird gezeigt, wie Werte aus `DateTime`-Objekte ermittelt werden:

```
protected static void DateTimeExample2()
   {
      // Initialize a new instance of the DateTime class to the current time
      DateTime dt = new DateTime(DateTime.now());

      // Output the formatted year, month, and day of this DateTime instance
      String str1 = String.format("Year: %d; Month: %d; Day: %d;", dt.getYear(),
dt.getMonth(), dt.getDay());
      System.out.println(str1);

      // Output the formatted hour, minute, and second of this DateTime instance
      String str2 = String.format("Hour: %d; Minute: %d; Second: %d;", dt.getHour(),
dt.getMinute(), dt.getSecond());
      System.out.println(str2);

      // Return the timezone (in minutes) of this DateTime instance
      System.out.println("Timezone: " + dt.getTimezoneOffset());

      // Get the DateTime as a java.util.Calendar value
```


© 2016-2022 Altova GmbH

```
      java.util.Calendar dt_java = dt.getValue();
      System.out.println("" + dt_java.toString());

      // Return the day of week of this DateTime instance
      System.out.println("Weekday: " + dt.getWeekday());

      // Check whether the DateTime instance has a timezone defined
      switch(dt.hasTimezone())
      {
         case CalendarBase.TZ_MISSING:
            System.out.println("No timezone.");
            break;
         case CalendarBase.TZ_UTC:
            System.out.println("The timezone is UTC.");
            break;
         case CalendarBase.TZ_OFFSET:
            System.out.println("This object has a timezone.");
            break;
         default:
            System.out.println("Unable to determine whether a timezone is defined.");
            break;
      }
   }
```

Im folgenden Codefragment wird gezeigt, wie der Zeitzonenunterschied eines `DateTime`-Objekts geändert wird:

```
protected static void DateTimeExample3()
{
   // Create a new DateTime object with timezone -0100 UTC
   DateTime dt = new DateTime(2015, 11, 24, 14, 30, 24, .459, -60);
   // Output the value before the change
   System.out.println("Before: " + dt.toString());
   // Change the offset to +0100 UTC
   dt.setTimezoneOffset(60);
   // Output the value after the change
   System.out.println("After:  " + dt.toString());
}
```

27.5.2 com.altova.types.Duration

Mit dieser Klasse können Sie XML-Attribute oder -Elemente vom Typ `xs:duration` verarbeiten.

Konstruktoren

	Name	Beschreibung
	`Duration(Duration newvalue)`	Initialisiert eine neue Instanz der `Duration`-Klasse auf das als Argument bereitgestellte `Duration`-Objekt.

	Name	Beschreibung
C	`Duration(int newyear, int newmonth, int newday, int newhour, int newminute, int newsecond, double newpartsecond, boolean newisnegative)`	Initialisiert eine neue Instanz der `Duration`-Klasse auf eine Zeitdauer, die anhand der als Argumente bereitgestellten Bestandteile konstruiert wird.

Methoden

	Name	Beschreibung
S	`static Duration getFromDayTime( int newday, int newhour, int newminute, int newsecond, double newpartsecond )`	Gibt ein `Duration`-Objekt zurück, das anhand der als Argument bereitgestellten Anzahl von Tagen, Stunden, Minuten, Sekunden und Sekundenbruchteile erstellt wird.
S	`static Duration getFromYearMonth( int newyear, int newmonth )`	Gibt ein `Duration`-Objekt zurück, das anhand der als Argument bereitgestellten Anzahl von Jahren und Monaten erstellt wird.
S	`static Duration parse( String s )`	Gibt ein `Duration`-Objekt zurück, das anhand des als Argument bereitgestellten String erstellt wurde. So kann z.B. anhand des String -P1Y1M1DT1H1M1.333S eine negative Zeitdauer von einem Jahr, einem Monat, einem Tag, einer Stunde, einer Minute, einer Sekunde und 0,333 Zehntelsekunden erstellt werden. Um eine negative Zeitdauer zu erstellen, hängen Sie das Minuszeichen (-) an den String an.
S	`static Duration parse( String s, ParseType pt )`	Gibt ein `Duration`-Objekt zurück, das unter Verwendung eines bestimmten Parse-Formats anhand des als Argument bereitgestellten String erstellt wird. Als Parse-Format kann eines der folgenden verwendet werden: **ParseType.DAYTIME** — Kann verwendet werden, wenn der String **s** aus einem der folgenden Bestandteile besteht: Tage, Stunden, Minuten, Sekunden, Sekundenbruchteile, z.B. -P4DT4H4M4.774S. **ParseType.DURATION** — Kann verwendet werden, wenn der String **s** aus einem der folgenden Bestandteile besteht: Jahre, Monate, Tage, Stunden, Minuten, Sekunden, Sekundenbruchteile, z.B. P1Y1M1DT1H1M1.333S. **ParseType.YEARMONTH** — Kann verwendet werden, wenn der String **s** aus einem der folgenden

© 2016-2022 Altova GmbH

	Name	Beschreibung
		Bestandteile besteht: Zum Beispiel: P3Y2M.
	int getDay()	Gibt die Anzahl der Tage in der aktuellen Duration-Instanz zurück.
	long getDayTimeValue()	Gibt den Tag- und Uhrzeitwert (in Millisekunden) der aktuellen Duration-Instanz zurück. Jahre und Monate werden ignoriert.
	int getHour()	Gibt die Anzahl der Stunden in der aktuellen Duration-Instanz zurück.
	int getMillisecond()	Gibt die Anzahl der Millisekunden in der aktuellen Duration-Instanz zurück.
	int getMinute()	Gibt die Anzahl der Minuten in der aktuellen Duration-Instanz zurück.
	int getMonth()	Gibt die Anzahl der Monate in der aktuellen Duration-Instanz zurück.
	double getPartSecond()	Gibt die Anzahl der Sekundenbruchteile in der aktuellen Duration-Instanz zurück.
	int getSecond()	Gibt die Anzahl der Sekunden in der aktuellen Duration-Instanz zurück.
	int getYear()	Gibt die Anzahl der Jahre in der aktuellen Duration-Instanz zurück.
	int getYearMonthValue()	Gibt das Jahr und den Monatswert (in Monaten) der aktuellen Duration-Instanz zurück. Tage, Stunden, Sekunden und Millisekunden werden ignoriert.
	boolean isNegative()	Gibt den Booleschen Wert **true** zurück, wenn die aktuelle Duration-Instanz negativ ist.
	void setDayTimeValue(**long** l)	Setzt die Zeitdauer auf die als Argument angegebene Anzahl der Millisekunden, wobei sich dies nur auf den Tag- und Uhrzeitbestandteil der Zeitdauer auswirkt.
	void setNegative(**boolean** isnegative)	Konvertiert die aktuelle Duration-Instanz in eine negative Zeitdauer.
	void setYearMonthValue(**int** l)	Setzt die Zeitdauer auf die als Argument angegebene Anzahl der Monate. Nur der Jahr- und Monatsteil der Zeitdauer ist davon betroffen.
	String toString()	Gibt die String-Darstellung der aktuellen Duration-Instanz zurück, z.B.: -P4DT4H4M4.774S

	Name	Beschreibung
	`String toYearMonthString()`	Gibt die String-Darstellung des YearMonth-Teils der aktuellen `Duration`-Instanz zurück, z.B.: `P1Y2M`

Beispiele

Stellen Sie sicher, dass die Altova-Typen importiert wurden, bevor Sie die folgenden Codefragmente in Ihrem Programm verwenden:

```
import com.altova.types.*;
import com.altova.types.Duration.ParseType;
```

In den folgenden Codefragmenten werden verschiedene Möglichkeiten zur Erstellung von `Duration`-Objekten gezeigt:

```
protected static void ExampleDuration()
{
   // Create a negative duration of 1 year, 1 month, 1 day, 1 hour, 1 minute, 1 second,
   // and 0.333 fractional second parts
   Duration dr = new Duration(1, 1, 1, 1, 1, 1, .333, true);

   // Create a duration from an existing Duration object
   Duration dr1 = new Duration(dr);

   // Create a duration of 4 days, 4 hours, 4 minutes, 4 seconds, .774 fractional second
parts
   Duration dr2 = Duration.getFromDayTime(4, 4, 4, 4, .774);

   // Create a duration of 3 years and 2 months
   Duration dr3 = Duration.getFromYearMonth(3, 2);

   // Create a duration from a string
   Duration dr4 = Duration.parse("-P4DT4H4M4.774S");

   // Create a duration from a string, using specific parse formats
   Duration dr5 = Duration.parse("-P1Y1M1DT1H1M1.333S", ParseType.DURATION);
   Duration dr6 = Duration.parse("P3Y2M", ParseType.YEARMONTH);
   Duration dr7 = Duration.parse("-P4DT4H4M4.774S", ParseType.DAYTIME);
}
```

Im folgenden Codefragment wird gezeigt, wie Sie den Wert von `Duration`-Objekten abrufen und definieren:

```
protected static void DurationExample2()
{
   // Create a duration of 1 year, 2 month, 3 days, 4 hours, 5 minutes, 6 seconds,
       // and 333 milliseconds
   Duration dr = new Duration(1, 2, 3, 4, 5, 6, .333, false);
```


© 2016-2022 Altova GmbH

```
   // Output the number of days in this duration
   System.out.println(dr.getDay());

   // Create a positive duration of one year and 333 milliseconds
   Duration dr1 = new Duration(1, 0, 0, 0, 0, 0, .333, false);
   // Output the day and time value in milliseconds
   System.out.println(dr1.getDayTimeValue());

   // Create a positive duration of 1 year, 1 month, 1 day, 1 hour, 1 minute, 1 second,
       // and 333 milliseconds
   Duration dr2 = new Duration(1, 1, 1, 1, 1, 1, .333, false);
   // Output the year and month value in months
   System.out.println(dr2.getYearMonthValue());

   // Create a positive duration of 1 year and 1 month
   Duration dr3 = new Duration(1, 1, 0, 0, 0, 0, 0, false);
   // Output the value
   System.out.println("The duration is now: " + dr3.toString());
   // Set the DayTime part of duration to 1000 milliseconds
   dr3.setDayTimeValue(1000);
   // Output the value
   System.out.println("The duration is now: " + dr3.toString());
   // Set the YearMonth part of duration to 1 month
   dr3.setYearMonthValue(1);
   // Output the value
   System.out.println("The duration is now: " + dr3.toString());
   // Output the year and month part of the duration
   System.out.println("The YearMonth part of the duration is: " +
dr3.toYearMonthString());
}
```

27.5.3 com.altova.xml.meta.Attribute

Mit Hilfe dieser Klasse können Sie Schemainformationen über anhand von Attributen generierte Klassen aufrufen. Beachten Sie, dass diese Klasse nicht dazu gedacht ist, dynamische Informationen über bestimmte Instanzen eines Attributs in einem XML-Dokument bereitzustellen. Es dient stattdessen dazu, Informationen über ein bestimmtes im XML-Schema definiertes Attribut programmatisch abzurufen.

Methoden

	Name	Beschreibung
●	`SimpleType getDataType()`	Gibt den Typ des Attributinhalts zurück.
●	`String getLocalName()`	Gibt den lokalen Namen des Attributs zurück.
●	`String getNamespaceURI()`	Gibt die Namespace URI des Attributs zurück.
●	**`boolean`** `isRequired()`	Gibt "true" zurück, wenn das Attribut obligatorisch ist.

27.5.4 com.altova.xml.meta.ComplexType

Mit Hilfe dieser Klasse können Sie Schemainformationen über anhand von complexTypes generierte Klassen aufrufen. Beachten Sie, dass diese Klasse nicht dazu gedacht ist, dynamische Informationen über bestimmte Instanzen eines complexType in einem XML-Dokument bereitzustellen. Es dient stattdessen dazu, Informationen über einen bestimmten im XML-Schema definierten complexType programmatisch abzurufen.

Methoden

	Name	Beschreibung
	`Attribute findAttribute(String localName, String namespaceURI)`	Sucht das Attribut mit dem definierten lokalen Namen und der Namespace URI
	`Element findElement(String localName, String namespaceURI)`	Sucht das Element mit dem definierten lokalen Namen und der Namespace URI.
	`Attribute[] GetAttributes()`	Gibt eine Liste aller Attribute zurück.
	`ComplexType getBaseType()`	Gibt den Basistyp dieses Typs zurück.
	`SimpleType getContentType()`	Gibt dem simpleType des Inhalts zurück.
	`Element[] GetElements()`	Gibt eine Liste aller Elemente zurück.
	`String getLocalName()`	Gibt den lokalen Namen des Typs zurück.
	`String getNamespaceURI()`	Gibt die Namespace URI des Typs zurück.

27.5.5 com.altova.xml.meta.Element

Mit Hilfe dieser Klasse können Sie Informationen über anhand von Schema-Elementen generierte Klassen aufrufen. Beachten Sie, dass diese Klasse nicht dazu gedacht ist, dynamische Informationen über bestimmte Instanzen eines Elements in einem XML-Dokument bereitzustellen. Es dient stattdessen dazu, Informationen über ein bestimmtes im XML-Schema definiertes Element programmatisch abzurufen.

Methoden

	Name	Beschreibung
	`ComplexType getDataType()`	Gibt den Typ des Elements zurück. Beachten Sie, dass dies immer ein complexType ist, selbst wenn er im Originalschema als simpleType deklariert wurde. Verwenden Sie `getContentType()` des zurückgegebenen Objekts zum den simple content type abzurufen.
	`String getLocalName()`	Gibt den lokalen Namen des Elements zurück.

© 2016-2022 Altova GmbH

	Name	Beschreibung
	`int getMaxOccurs()`	Gibt den im Schema definierten maxOccurs Wert zurück.
	`int getMinOccurs()`	Gibt den im Schema definierten minOccurs Wert zurück.
	`String getNamespaceURI()`	Gibt die Namespace URI des Elements zurück.

27.5.6 com.altova.xml.meta.SimpleType

Mit Hilfe dieser Klasse können Sie Schemainformationen über anhand von simpleTypes generierte Klassen aufrufen. Beachten Sie, dass diese Klasse nicht dazu gedacht ist, dynamische Informationen über bestimmte Instanzen eines simpleType in einem XML-Dokument bereitzustellen. Es dient stattdessen dazu, Informationen über einen bestimmten im XML-Schema definierten simpleTypes programmatisch abzurufen.

Methoden

	Name	Beschreibung
	`SimpleType getBaseType()`	Gibt den Basistyp dieses Typs zurück.
	`String[] getEnumerations()`	Gibt ein Array aller Enumeration Facets zurück.
	`int getFractionDigits()`	Gibt den Wert dieses Facet zurück.
	`int getLength()`	Gibt den Wert dieses Facet zurück.
	`String getLocalName()`	Gibt den lokalen Namen des Typs zurück.
	`String getMaxExclusive()`	Gibt den Wert dieses Facet zurück.
	`String getMaxInclusive()`	Gibt den Wert dieses Facet zurück.
	`int getMaxLength()`	Gibt den Wert dieses Facet zurück.
	`String getMinExclusive()`	Gibt den Wert dieses Facet zurück.
	`String getMinInclusive()`	Gibt den Wert dieses Facet zurück.
	`int getMinLength()`	Gibt den Wert dieses Facet zurück.
	`String getNamespaceURI()`	Gibt die Namespace URI des Typs zurück.
	`String[] getPatterns()`	Gibt ein Array aller Pattern Facets zurück.
	`int getTotalDigits()`	Gibt den Wert dieses Facet zurück.

	`int getWhitespace()`	Gibt den Wert des Whitespace Facet zurück, der einer der folgenden ist:
		`com.altova.typeinfo.WhitespaceType.Whitespace_Unknown`
		`com.altova.typeinfo.WhitespaceType.Whitespace_Preserve`
		`com.altova.typeinfo.WhitespaceType.Whitespace_Replace`

	Name	Beschreibung
		com.altova.typeinfo.WhitespaceType.Whitespace_Collapse

27.5.7 com.[YourSchema].[Doc]

Wenn anhand eines XML-Schemas Code generiert wird, stellt der generierte Code eine Dokumentklasse mit demselben Namen wie dem des Schemas bereit. Diese Klasse enthält alle möglichen Root-Elemente als Members sowie die folgenden Members. Beachten Sie, dass "Doc" in den unten aufgelisteten Methodennamen für den Namen der generierten Dokumentklasse selbst steht.

Methoden

	Name	Beschreibung
S	static Doc createDocument()	Erstellt ein neues leeres XML-Dokument.
S	static void declareAllNamespacesFromSchema(com.altova.xml.ElementType node)	Deklariert alle Namespaces aus dem XML-Schema an dem als Argument angegebenen Element (normalerweise dem XML-Root-Element). Diese Methode eignet sich dann, wenn Ihr Schema mehrere Namespace-Deklarationen, von denen jedes auf ein Präfix gemappt ist, hat, und Sie alle davon für das als Argument angegebene Element deklarieren möchten.
S	static Doc loadFromBinary(byte[] xml)	Lädt ein XML-Dokument aus einem Byte Array.
S	static Doc loadFromFile(String fileName)	Lädt ein XML-Dokument aus einer Datei.
S	static Doc loadFromString(String xml)	Lädt ein XML-Dokument aus einem String.
	byte[] saveToBinary(boolean prettyPrint)	Speichert ein XML-Dokument mit optionaler Pretty-Print-Formatierung in einem Byte Array.
	byte[] saveToBinary(boolean prettyPrint, String encoding)	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung in einem Byte-Array.
	byte[] saveToBinary(boolean prettyPrint, String encoding, boolean bigEndian, boolean writeBOM)	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung in einem Byte-Array. Für Unicode-Kodierungen können die Bytefolge und Unicode-Bytefolge-Markierung definiert werden.
	void saveToFile(String fileName, boolean prettyPrint)	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung in einer Datei.
	void saveToFile(String fileName, boolean prettyPrint, boolean omitXmlDecl)	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit UTF-8-Kodierung in einer Datei. Wenn

© 2016-2022 Altova GmbH

	Name	Beschreibung
		`omitXmlDecl` auf "true" gesetzt ist, wird die XML-Deklaration nicht geschrieben.
	`void saveToFile(String fileName, boolean prettyPrint, boolean omitXmlDecl, String encoding)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung in einer Datei. Wenn `omitXmlDecl` auf "true" gesetzt ist, wird die XML-Deklaration nicht geschrieben.
	`void saveToFile(String fileName, boolean prettyPrint, boolean omitXmlDecl, String encoding, boolean bBigEndian, boolean bBOM)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung in einer Datei. Wenn `omitXmlDecl` auf "true" gesetzt ist, wird die XML-Deklaration nicht geschrieben. Für Unicode-Kodierungen können die Bytefolge und Unicode-Bytefolge-Markierung definiert werden.
	`void saveToFile(String fileName, boolean prettyPrint, String encoding)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung in einer Datei.
	`void saveToFile(String fileName, boolean prettyPrint, String encoding, boolean bBigEndian, boolean bBOM)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung mit der definierten Kodierung in einer Datei. Für Unicode-Kodierungen können die Bytefolge und Unicode-Bytefolge-Markierung definiert werden.
	`String saveToString(boolean prettyPrint)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung in einem String.
	`String saveToString(boolean prettyPrint, boolean omitXmlDecl)`	Speichert ein XML-Dokument mit optionaler "pretty-print"-Formatierung in einem String. Wenn `omitXmlDecl` auf "true" gesetzt ist, wird die XML-Deklaration nicht geschrieben.
	`void setSchemaLocation(String schemaLocation)`	Fügt ein `xsi:schemaLocation`- oder `xsi:noNamespaceSchemaLocation`-Attribut zum Root-Element hinzu. Es muss bereits ein Root-Element vorhanden sein.

27.5.8 com.[YourSchema].[ElementType]

Diese Klasse enthält Methoden zur Bearbeitung von XML-Elementen aus Ihrem Schema. Die Methoden dieser Klasse können für Elemente, nicht aber für das XML-Dokument selbst, aufgerufen werden. Beachten Sie, dass Sie die Klasse nicht direkt instantiieren müssen, um Methoden dieser Klasse aufzurufen. Jedes mit den Methoden **`append()`** oder **`appendWithPrefix()`** erstelltes Element hat den Typ [`ElementType`].

Methoden

	Name	Beschreibung
	`void declareNamespace(String prefix, String nsURI)`	Diese Methode erhält zwei Argumente, beide vom Typ "String": das gewünschte Präfix und die gewünschte Namespace URI. Das als Argument bereitgestellte Präfix

	Name	Beschreibung

		wird auf den als Argument bereitgestellten Namespace URI-Wert gemappt. Wenn das als Argument angegebene Präfix leer ist, erstellt die Methode die Standard-Namespace-Deklaration im Element oder setzt diese außer Kraft.
		
		Angenommen, das XML-Dokument hat ein XML-Element namens "purchase". Bei Aufruf von
		
		`purchase.declareNamespace("ord", "http://OrderTypes");`
		
		wird das XML-Dokument zu
		
		`<purchase xmlns:ord="http://OrderTypes" />`
		
		Ein weiteres Beispiel: Bei Aufruf von
		
		`purchase.declareNamespace("", "http://OrderTypes");`
		
		wird das XML-Dokument zu
		
		`<purchase xmlns="http://OrderTypes" />`
		
		Anmerkung: Der deklarierte Namespace wird beim Anhängen danach folgender Child-Elemente oder Attribute nach den folgenden Regeln verwendet:
		
		1. Wenn der Child-Namespace der Standard-Namespace ist, wird ein leeres Präfix verwendet.
		2. Wenn der Child-Namespace der gleiche ist wie der Parent Namespace ist, wird das Parent-Präfix verwendet.
		3. Anderfalls wird unter Verwendung des in Abschnitt "B.2: Namespace Prefix Lookup" unter https://www.w3.org/TR/2002/WD-DOM-Level-3-Core-20021022/namespaces-algorithms.html beschriebenen Lookup-Algorithmus nach dem nächstgelegenen Präfix vom Parent bis nach oben gesucht .
		4. Wenn kein Präfix für einen Element-Namespace gefunden wird, wird ein leeres Präfix verwendet.

© 2016-2022 Altova GmbH

27.5.9 com.[YourSchema].[YourSchemaType].MemberAttribute

Wenn anhand eines XML-Schemas Code generiert wird, wird für jedes Member-Attribut eines Typs eine Klasse erstellt. In der Beschreibung unten steht "AttributeType" für den Typ des Member-Attributs selbst.

Methoden

	Name	Beschreibung
	`boolean exists()`	Gibt den Wert "true" zurück, wenn das Attribut vorhanden ist.
	`int getEnumerationValue()`	Wird nur für Enumeration Types generiert. Gibt eine der für die möglichen Werte generierten Konstanten zurück oder **Invalid,** wenn der Wert mit keinem der Enumerationswerte im Schema übereinstimmt.
	`com.altova.xml.meta.Attribute getInfo()`	Gibt ein Objekt zum Abfragen von Schemainformationen zurück (siehe `com.altova.xml.meta.Attribute` 1245).
	`AttributeType getValue()`	Ruft den Attributwert ab.
	`void remove()`	Entfernt das Attribut aus dem übergeordneten Element.
	`void setEnumerationValue(int)`	Wird nur für Enumeration Types generiert. Übergeben Sie eine der für die möglichen Werte generierten Konstanten an diese Methode, um den Wert zu definieren.
	`void setValue(AttributeType value)`	Definiert den Attributwert.

27.5.10 com.[YourSchema].[YourSchemaType].MemberElement

Wenn anhand eines XML-Schemas Code generiert wird, wird für jedes Member-Element eines Typs eine Klasse mit den folgenden Menbers erstellt. In den Beschreibungen unten steht "MemberType" für den Typ des Member-Elements selbst.

Methoden

	Name	Beschreibung
	`MemberType append()`	Erstellt ein neues Element und hängt es an das übergeordnete Element an.
	`MemberType appendWithPrefix(String prefix)`	Erstellt ein neues Element mit dem als Argument angegebenen Präfix und hängt es an seinen Parent an. Ein Beispiel dazu finden Sie unter Beispiel: Bestellung 1197.
	`MemberType at(int index)`	Gibt die Instanz des Member Elements am angegebenen Index zurück.

	Name	Beschreibung
	`int count()`	Gibt die Anzahl der Elemente zurück.
	`boolean exists()`	Gibt den Wert "true" zurück, wenn mindestens ein Element vorhanden ist.
	`MemberType first()`	Gibt die erste Instanz des Member Elements zurück.
	`int getEnumerationValue()`	Wird nur für Enumeration Types generiert. Gibt eine der für die möglichen Werte generierten Konstanten zurück oder **Invalid,** wenn der Wert mit keinem der Enumerationswerte im Schema übereinstimmt.
	`com.altova.xml.meta.Element getInfo()`	Gibt ein Objekt zum Abfragen von Schemainformationen zurück (siehe `com.altova.xml.meta.Element` 1246).
	`MemberType getValue()`	Ruft den Elementinhalt ab (wird nur generiert, wenn das Element simple oder mixed content haben kann).
	`java.util.Iterator iterator()`	Gibt ein Objekt für iterierende Instanzen des Member Elements zurück.
	`MemberType last()`	Gibt die letzte Instanz des Member Elements zurück.
	`void remove()`	Löscht alle Instanzen des Elements aus dem übergeordneten Element.
	`void removeAt(int index)`	Löscht die Instanz des durch den Index definierten Elements.
	`void setEnumerationValue(int index)`	Wird nur für Enumeration Types generiert. Übergeben Sie eine der für die möglichen Werte generierten Konstanten an diese Methode, um den Wert zu definieren.
	`void setValue(MemberType value)`	Definiert den Elementinhalt (wird nur generiert, wenn das Element simple oder mixed content haben kann).

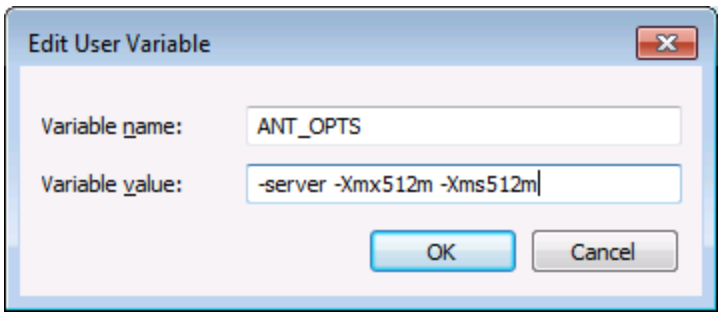
© 2016-2022 Altova GmbH

27.6 Codegenerierung Tipps

Behebung von Ausnahmeereignissen im Zusammenhang mit zu wenig Arbeitsspeicher bei der Java-Kompilierung

Es kommt vor, dass beikomplexen XML Schemas eine so große Menge von Code generiert wird, dass es bei der Kompilierung mit Ant zu einem `java.lang.OutofMemory`-Ausnahmeereignis kommt. Um dieses Problem zu beheben:

- Fügen Sie die Umgebungsvariable `ANT_OPTS` hinzu. Damit werden bestimmte Ant-Optionen festgelegt, wie z.B. wie viel Arbeitsspeicher dem Compiler zur Verfügung steht. Sie können die Werte wie unten gezeigt definieren.



- Um sicherzustellen, dass der Compiler und der generierte Code im selben Prozess wie Ant ausgeführt wird, ändern Sie das Attribut `fork` in der Datei **build.xml** in `false` um.

Je nach verfügbarem Arbeitsspeicher auf Ihrem Computer und je nach Projektgröße müssen Sie diese Werte eventuell anpassen. Nähere Informationen finden Sie in der Java VM-Dokumentation.

Bei der Ausführung des `ant jar`-Befehls wird eventuell eine Fehlermeldung angezeigt, dass "`[...] das Archiv mehr als 65535 Einheiten enthält`". Damit diese Fehlermeldung nicht auftritt, wird empfohlen, Versionen ab Ant 1.9 zu verwenden und in der Datei **build.xml** zum Element `<jar> zip64mode="as-needed"` hinzuzufügen.

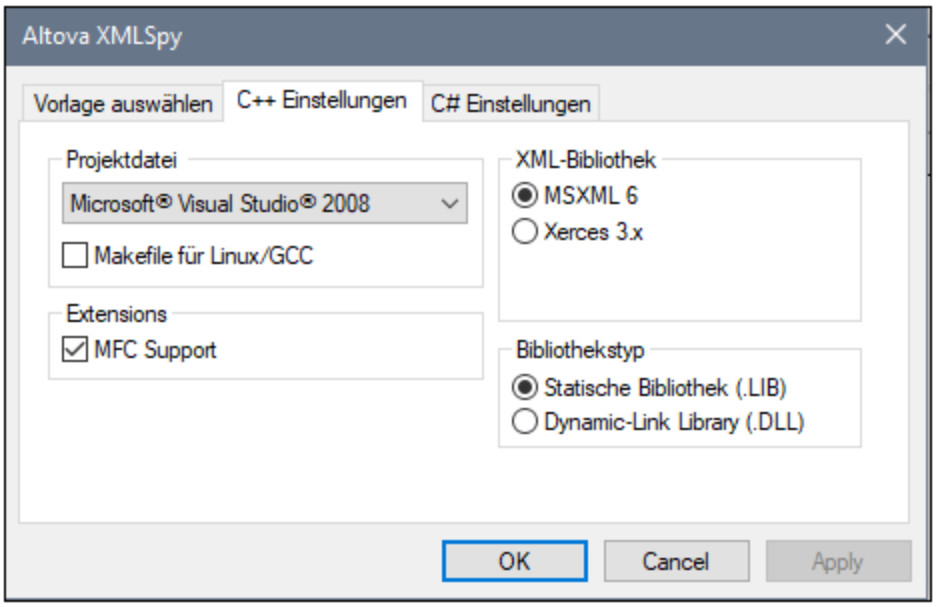
Reservieren von Methodennamen

Beim Anpassen der Codegenerierung mittels der vorgegebenen SPL-Dateien ist es unter Umständen nötig, Methodennamen zu reservieren. Gehen Sie dazu folgendermaßen vor:

1. Navigieren Sie zum Programminstallationsverzeichnis z.B. `C:\Programme\Altova\XMLSpy2023`.
2. Navigieren Sie im Unterverzeichnis **spl** zum Verzeichnis für die entsprechende Programmiersprache z.B. **..\spl\java**.
3. Öffnen Sie entweder die Datei **settings.spl** und fügen Sie in den Abschnitt "reserve" eine neue Zeile ein, z.B. reserve `"myReservedWord"`.
4. Generieren Sie den Programmcode neu.

27.7 Code Generator-Optionen

Über die Menüoption **DTD/Schema | Programmcode generieren** können Sie die Programmier-Zielsprache sowie die jeweiligen Einstellungen für die Codegenerierung definieren.



Es stehen die folgenden Einstellungen zur Verfügung.

C++-Einstellungen	Definiert die Compiler-Einstellungen für die C++-Umgebung: • die Visual Studio Version (2013, 2015, 2017, 2019, 2022) • ob ein makefile für Linux mit GCC Compiler generiert werden muss. • die XML-Bibliothek (MSXML, Xerces 3.x) • ob statische oder dynamische Bibliotheken generiert werden müssen. • ob Code mit oder ohne MFC-Unterstützung generiert werden muss. Mit der Option **Makefile für Linux/GCC** werden Makefiles zum generierten Code hinzugefügt. Es werden C++-Quelldateien generiert, damit diese mit Hilfe von `#ifdef`-Konstrukten übertragbar werden, damit unterschiedliche Compiler und Betriebssysteme unterstützt werden. Wenn Sie beabsichtigen, den generierten Code unter Linux mit GCC (GNU Compiler Collection) zu kompilieren, beachten Sie Folgendes: • Bei der Linux/GCC-Kompilierung wird **Xerces 3.x** als einzige XML-Bibliothek unterstützt. • Die Aktivierung der Option **MFC-Unterstützung** hat keine Auswirkung auf die Kompilierung mit Linux/GCC.
C#-Einstellungen	Wählen Sie die Option **Microsoft .NET Core .NET 3.1**, **Microsoft .NET 5.0** oder **Microsoft .NET 6.0** aus, um eine Visual Studio-Projektmappe für die

© 2016-2022 Altova GmbH

	entsprechende Plattform zu generieren. Wenn Sie die **.NET Framework**-Plattform für eine bestimmte Visual Studio-Version auswählen müssen, wählen Sie eine beliebige der **Microsoft Visual Studio 2010-2019**-Optionen aus - in diesem Fall ist das Ziel der generierten Projektmappe die .NET Framework-Version für die entsprechende Visual Studio-Version.

27.8 SPL-Referenz

In diesem Abschnitt erhalten Sie eine Übersicht über Spy Programming Language (SPL), die Template-Sprache von Code Generator.

Es wird vorausgesetzt, dass Sie bereits über Programmiererfahrung verfügen und mit Operatoren, Funktionen, Variablen und Klassen sowie den Grundzügen von in SPl häufig verwendeter objektorientierter Programmierung vertraut sind. Außerdem sollten Sie gute XML-Schema-Kenntnisse haben.

Im Ordner \XMLSpy\spl finden Sie die von XMLSpy verwendeten Vorlagen**.** Anhand dieser Dateien können Sie Ihre eigenen Vorlagen entwickeln.

Funktionsweise von Code Generator

Die Basis von Code Generator bilden die Vorlagendateien (.spl) und das von XMLSpy bereitgestellte Objektmodell. Die Vorlagendateien enthalten SPL-Anweisungen zum Erstellen von Dateien, Lesen von Informationen aus dem Objektmodell und Ausführen von Berechnungen, die in der Zielprogrammiersprache auch Literalcodefragmente enthalten.

Die Vorlagendatei wird von Code Generator interpretiert und je nach Vorlage wird eine **.cpp, .java, .cs-**Quellcodedatei Projektdatei oder eine andere Dateiart ausgegeben.
Der Quellcode wird anschließend in eine exe-Datei kompiliert. Nach Starten der exe-Datei greift diese auf die durch die Schema-Datei beschriebenen XML-Daten zu.

SPL-Dateien haben Zugriff auf die verschiedensten Informationen, die aus den Quellschemas geholt werden. Beachten Sie bitte, dass eine SPL-Datei nicht an ein bestimmtes Schema gebunden ist, sondern Zugriff auf alle Schemas erlaubt! Stellen Sie sicher, dass Sie Ihre SPL-Dateien generisch anlegen und Strukturen, die nur auf bestimmte Schemas zutreffen, vermeiden.

Beispiel: Erstellen einer neuen Datei in SPL

Dies ist ein sehr einfacher SPL-Code. Damit wird eine Datei names **test.cpp** erstellt. Das Include Statement wird innerhalb davon positioniert. Der Befehl "close" schließt und beendet die Vorlage.

```
[create "test.cpp"]
#include "stdafx.h"
[close]
```

27.8.1 Grundlegende SPL-Struktur

Eine SPL-Datei enthält Literaltext, der ausgegeben werden soll, der zwischendurch Code Generator-Anweisungen enthält.

Code Generator-Anweisungen sind in eckige Klammern eingeschlossen '**[**' und '**]**'. Innerhalb einer Klammer können mehrere Anweisungen stehen. Zusätzliche Anweisungen müssen durch eine neue Zeile oder einen Doppelpunkt ':' getrennt werden.

© 2016-2022 Altova GmbH

Gültige Beispiele sind:

```
[$x = 42
$x = $x + 1]
```

oder

```
[$x = 42: $x = $x + 1]
```

Hinzufügen von Text zu Dateien

Text, der nicht innerhalb von **[** und **]** steht, wird direkt in die Ausgabedatei geschrieben. Falls es keine aktuelle Ausgabedatei gibt, wird der Text ignoriert (eine Anleitung zum Erstellen einer Ausgabedatei finden Sie unter Verwendung von Dateien[1262]).
Um eckige Klammern als Literale auszugeben, setzen Sie davor als Excape-Zeichen einen umgekehrten Schrägstrich: **\[** und **\]**. Um einen umgekehrten Schrägstrich auszugeben, verwenden Sie \\.

Kommentare

Kommentare innerhalb eines Anweisungsblocks beginnen immer mit einem **'** Zeichen und enden an der nächsten Zeile oder an einem Zeichen, zum Beenden eines Blocks **]**.

27.8.2 Deklarationen

Die folgenden Anweisungen werden beim Parsen der SPL-Vorlagendatei ausgewertet. Flusssteuerungsanweisungen wie z.B. Bedingungen, Schleifen oder Subroutinen haben **keinen** Einfluss darauf und werden immer nur einmal ausgewertet.

Die Groß- und Kleinschreibung spielt wie bei allen Schlüsselwörtern in SPL keine Rolle.

Alle diese Deklarationen müssen unbedingt innerhalb von eckigen Klammern stehen.

map ... to ...

```
map mapname key to value [, key to value ]...
```

Diese Anweisung fügt Informationen zu einer Zuordnung hinzu. Beispiele für konkrete Anwendungen siehe unten.

```
map schemanativetype schematype to typespec
```

Der angegebene built-in XML-Schematyp wird unter Verwendung der angegebenen Formatierung auf den angegebenen Native Type oder die Klasse gemappt. Diese Einstellung gilt nur für die Codegenerierung für Version 2007r3 und höher. Typespec ist ein Native Type oder ein Klassenname, gefolgt von einem Komma, gefolgt vom der Klasseninstanz des Formatters..

Beispiel:

```
map schemanativetype "double" to "double,Altova::DoubleFormatter"
```

map type ... to ...

```
map type schematype to classname
```

Der angegebene vordefinierte XML-Schematyp wird auf die angegebene Klasse gemappt. Diese Einstellung gilt nur für die Codegenerierung für Version 2007 oder niedriger.

Beispiel:

```
map type "float" to "CSchemaFloat"
```

default ... is ...

```
default setting is value
```

Mit dieser Anweisung können Sie festlegen, wie Klassen- und Member-Namen vom XML-Schema abgeleitet werden.
Beachten Sie, dass die die Groß- und Kleinschreibung bei Einstellungsnamen relevant ist.

Beispiel:

```
default "InvalidCharReplacement" is "_"
```

Einstellungsname	Erklärung
ValidFirstCharSet	Zulässige Zeichen zum Starten eines Identifier
ValidCharSet	Zulässige Zeichen für andere Zeichen in einem Identifier
InvalidCharReplacement	Das Zeichen, durch das alle Zeichen in Namen ersetzt werden, die nicht im ValidCharSet enthalten sind
AnonTypePrefix	Präfix für Namen von anonymen Typen*
AnonTypeSuffix	Suffix für Namen von anonymen Typen*
ClassNamePrefix	Präfix für generierte Klassennamen
ClassNameSuffix	Suffix für generierte Klassennamen
EnumerationPrefix	Präfix für Symbolkonstanten, die für Enumerationswerte deklariert wurden

© 2016-2022 Altova GmbH

EnumerationUpperCase	"ein" für das Konvertieren der Namen der Enumerationskonstanten in Großbuchstaben
FallbackName	Verwende diesen Namen, wenn ein Name nur aus Zeichen besteht, die sich nicht im ValidCharSet befinden

* Namen anonymer Typen werden aus AnonTypePrefix + Elementname + AnonTypeSuffix erzeugt.

reserve

```
reserve word
```

Fügt das angegebene Wort zur Liste der reservierte Wörter hinzu. Damit stellen Sie sicher, dass es nie als Klasse oder Member-Name generiert wird.

Beispiel:

```
reserve "while"
```

include

Inkludiert die angegebene Datei als SPL-Quelldatei. Auf diese Art können Sie Ihre Vorlage zur einfacheren Bearbeitung und Verwaltung in mehrere Dateien aufteilen.

```
include filename
```

Beispiel:

```
include "Module.cpp"
```

27.8.3 Variablen

Jede etwas komplexere SPL-Datei enthält Variablen. Einige Variablen sind vom Code Generator vordefiniert[1260]. Sie können jederzeit neue Variablen erstellen, indem Sie diesen Werte zuweisen.

Das **$**-Zeichen wird zur **Deklarierung** oder **Verwendung** einer Variable verwendet. Vor der Variable muss immer ein **$** stehen.

Bei Variablennamen muss die **Groß- und Kleinschreibung beachtet werden.**

Variablentypen:

- Integer - wird auch als Boolescher Wert verwendet, wobei 0 false ist und alle anderen Werte true.
- String
- Objekt - wird von XMLSpy bereitgestellt
- Iterator (siehe foreach[1265]-Anweisung)

Variablentypen werden durch die erste Zuweisung deklariert.

```
[$x = 0]
```

x ist nun ein Integer.

```
[$x = "teststring"]
```

x wird nun als String behandelt.

Strings

String-Konstanten müssen wie im oben gezeigten Beispiel immer innerhalb von doppelten Anführungszeichen stehen. **\n** und **\t** innerhalb von doppelten Anführungszeichen werden als neue Zeile und Tabulator interpretiert, \" ist ein literales doppeltes Anführungszeichen und \\ ist ein umgekehrter Schrägstrich. String-Konstanten können auch mehrere Zeilen einnehmen.

Bei der Verkettung von Strings wird das **&** Zeichen verwendet.

```
[$BasePath = $outputpath & "/" & $JavaPackageDir]
```

Objekte

Objekte stehen für die Informationen, die im XML-Schema enthalten sind. Objekte haben **Eigenschaften**, die über den **.** Operator aufgerufen werden können. In SPL können keine neuen Objekte erstellt werden (sie werden durch den Code Generator vordefiniert, aus dem Eingabe-Schema abgeleitet). Sie können Objekte jedoch Variablen zuweisen.

Beispiel:

```
class [=$class.Name]
```

In diesem Beispiel wird das Wort "class" ausgegeben, gefolgt von einem Leerzeichen und dem Wert der Eigenschaft **Name** des Objekts **$class**.

27.8.4 Vordefinierte Variablen

Nachdem eine Schema-Datei von Code Generator analysiert wurde, sind die in der Tabelle aufgelisteten Objekte im Template-Prozessor vorhanden.

Name	Typ	Beschreibung
$schematype	integer	1 für DTD, 2 für XML-Schema
$TheLibrary	Library 1269	Die anhand des XML-Schemas oder der DTD abgeleitete Bibliothek

© 2016-2022 Altova GmbH

$module	string	Name des Quellschemas ohne Erweiterung
$outputpath	string	Der vom Benutzer definierte Ausgabepfad oder der Standardausgabepfad

Nur bei **C++** Generierung:

Name	Typ	Beschreibung
$domtype	Integer	1 bei MSXML, 2 bei Xerces
$libtype	Integer	1 bei statischer LIB, 2 bei DLL
$mfc	Boolean	True, wenn MFC Unterstützung aktiviert ist
$VSVersion	Integer	Definiert die Visual Studio Version. Gültige Werte: **0** No Visual Studio project **2010** Visual Studio 2010 **2013** Visual Studio 2013 **2015** Visual Studio 2015 **2017** Visual Studio 2017 **2019** Visual Studio 2019

Nur bei **C#** Generierung:

Name	Typ	Beschreibung
$VSVersion	Integer	Definiert die Visual Studio Version. Gültige Werte: **0** No Visual Studio project **2010** Visual Studio 2010 **2013** Visual Studio 2013 **2015** Visual Studio 2015 **2017** Visual Studio 2017 **2019** Visual Studio 2019

27.8.5 Erstellen von Ausgabedateien

Diese Anweisung dienen zum Erstellen von Ausgabedateien mit Hilfe der Codegenerierung.
Beachten Sie, dass all diese Anweisungen sich innerhalb eines Blocks befinden müssen, der innerhalb von eckigen Klammern stehen muss.

create

create *filename*

Erstellt eine neue Datei. Die Datei muss mit der **close**-Anweisung geschlossen werden. Die gesamte darauf folgende Ausgabe wird in die angegebene Datei geschrieben.

Beispiel:

```
[create $outputpath & "/" & $JavaPackageDir & "/" & $application.Name & ".java"]
package [=$JavaPackageName];

public class [=$application.Name]Application {
...
}
[close]
```

close

Schließt die aktuelle Ausgabedatei.

=$*variable*

Schreibt den Wert der angegebenen Variable in die aktuelle Ausgabedatei.

Beispiel:

```
[$x = 20+3]
The result of your calculation is [=$x] - so have a nice day!
```

Die Dateiausgabe lautet:

```
The result of your calculation is 23 - so have a nice day!
```

write

© 2016-2022 Altova GmbH

write *string*

Schreibt den String in die aktuelle Ausgabedatei.

Beispiel:

```
[write "C" & $name]
```

Dies kann auch geschrieben werden als:

```
C[=$name]
```

filecopy ... to ...

filecopy *source* **to** *target*

Kopiert die Quelldatei ohne jede Interpretation in die Zieldatei.

Beispiel:

```
filecopy "java/mapforce/mapforce.png" to $outputpath & "/" & $JavaPackageDir &
"/mapforce.png"
```

27.8.6 Operatoren

Operatoren in SPL funktionieren wie in den meisten anderen Programmiersprachen.

Liste von SPL-Operatoren absteigend nach Vorrangigkeit gereiht:

.	Objekteigenschaft aufrufen
( )	Gruppierung eines Ausdrucks
true	Boolesche Konstante "true"
false	Boolesche Konstante "false"
&	String-Verkettung
-	Zeichen für negative Zahl
not	Logische Negation
*	Multiplizieren
/	Dividieren
%	Modulo
+	Addieren
-	Subtrahieren

<= Kleiner oder gleich
< Kleiner als
>= Größer oder gleich
> Größer als

= Gleich
<> Nicht gleich

and Logische Konjunktion (mit short circuit-Auswertung)
or Logische Disjunktion (mit short circuit-Auswertung)

= Zuweisung

27.8.7 Bedingungen

SPL erlaubt die Verwendung von Standard-"if"-Anweisungen. Die Syntax lautet wie folgt:

```
if condition
        statements
else
        statements
endif
```

oder ohne else:

```
if condition
        statements
endif
```

Beachten Sie bitte, dass für die Bedingung keine runden Klammern verwendet werden!

Wenn in allen anderen Programmiersprachen werden Bedingungen mit logischen und Vergleichsoperatoren[1263] konstruiert.

Beispiel:

```
[if $namespace.ContainsPublicClasses and $namespace.Prefix <> ""]
       whatever you want ['inserts whatever you want, in the resulting file]
[endif]
```

Switch

SPL enthält auch eine Multiple Choice-Anweisung.

Syntax:

```
switch $variable
```


© 2016-2022 Altova GmbH

```
       case X:
              statements
       case Y:
       case Z:
              statements
       default:
              statements
endswitch
```

Die Case-Bezeichnungen müssen Konstanten oder Variablen sein.

Die switch-Anweisung in SPL fällt nicht durch die Cases (wie in C), daher wird auch keine "break"-Anweisung benötigt.

27.8.8 Collections und foreach

Collections und Iteratoren

Eine Collection enthält mehrere Objekte - wie ein gewöhnliches Array. Iteratoren lösen das Problem des Speicherns und Inkrementierens von Array Indizes beim Aufrufen von Objekten.

Syntax:

```
foreach iterator in collection
       statements
next
```

Beispiel:

```
[foreach $class in $classes
       if not $class.IsInternal
              ]      class [=$class.Name];
[      endif
next]
```

Beispiel 2:

```
[foreach $i in 1 To 3
      Write "// Step " & $i & "\n"
      ' Do some work
next]
```

Die erste Zeile:
$classes ist das **globale Objekt**[1260] von allen Typen. $classes ist eine Collection (Sammlung) von einzelnen Klassenobjekten.

Foreach iteriert durch alle Datenelemente in $classes und führt für die Anweisung den Code der auf die Anweisung folgt, bis zur **nächsten** Anweisung aus.

In jeder Iteration wird **$class** dem nächsten Klassenobjekt zugewiesen. Sie arbeiten einfach mit dem Klassenobjekt, anstelle classes[i]->class->Name(), zu verwenden, wie das in C++ der Fall wäre.

Alle Collection-Iteratoren haben die folgenden zusätzlichen Eigenschaften:

Index	der aktuelle Index beginnend mit 0
IsFirst	true, wenn das aktuelle Objekt das erste der Collection ist (Index ist 0)
IsLast	true, wenn das aktuelle Objekt das letzte einer Collection ist
Current	das aktuelle Objekt (dies ist implizit, wenn es nicht definiert ist und kann weggelassen werden)

Beispiel:

```
[foreach $enum in $facet.Enumeration
       if not $enum.IsFirst
              ], [
       endif
       ]"[=$enum.Value]"[
next]
```

27.8.9 Subroutinen

Code Generator unterstützt nun Subroutinen in der Form von Prozeduren oder Funktionen.

Funktionen:

- Übergabe von Werten nach Wert und nach Referenz
- Lokale/globale Parameter (lokal innerhalb von Subroutinen)
- Lokale Variablen
- Rekursiver Aufruf (Subroutinen können sich selbst aufrufen)

27.8.9.1 Deklaration einer Subroutine

Subroutinen

Syntax-Beispiel:

```
Sub SimpleSub()
       ... lines of code
 EndSub
```


© 2016-2022 Altova GmbH

- **Sub** ist das Schlüsselwort, das die Prozedur notiert.
- **SimpleSub** ist der Name, der der Subroutine zugewiesen wurde.
- Runde **Klammern** können eine Parameterliste enthalten.
- Der Code-Block einer Subroutine beginnt direkt nach der geschlossenen Klammer nach dem Parameter.
- **EndSub** notiert das Ende des Code-Blocks.

Bitte beachten Sie:
Eine rekursive oder kaskadierende Subroutine-**Deklaration** ist nicht zulässig, d.h. eine Subroutine darf keine weiteren Subroutinen enthalten.

Parameter

Parameter können auch durch Prozeduren unter Verwendung der folgenden Syntax übergeben werden:

- Alle Parameter müssen Variablen sein
- Variablen muss das **$-**Zeichen vorangestellt werden
- Lokale Variablen werden in einer Subroutine definiert
- Globale Variablen werden explizit außerhalb von Subroutinen deklariert
- Mehrere Parameter werden innerhalb runder Klammern durch Kommas "**,**" voneinander getrennt
- Parameter können Werte übergeben

Parameter - Übergabe von Werten

Parameter können auf zwei Arten übergeben werden: nach Wert oder nach Referenz, wobei die Schlüsselwörter ByVal bzw. ByRef verwendet werden.

Syntax:

```
' define sub CompleteSub()
[Sub CompleteSub( $param, ByVal $paramByValue, ByRef $paramByRef )
] ...
```

- **ByVal** definiert, dass der Parameter nach Wert übergeben wird. Beachten Sie, dass die meisten Objekte nur nach Referenz übergeben werden können.
- **ByRef** definiert, dass der Parameter nach Referenz übergeben wird. Dies ist die Standardeinstellung, wenn weder ByVal noch ByRef definiert ist.

Funktionsrückgabewerte

Für die Rückgabe eines Werts von einer Subroutine verwenden Sie ein **Return-**Statement. Eine solche Funktion kann von innerhalb eines Ausdrucks aufgerufen werden.

Beispiel:

```
' define a function
[Sub MakeQualifiedName( ByVal $namespacePrefix, ByVal $localName )
if $namespacePrefix = ""
  return $localName
else
  return $namespacePrefix & ":" & $localName
```

```
endif
EndSub
]
```

27.8.9.2 Subroutinenaufruf

Verwenden Sie zum Aufrufen einer Subroutine **call**, gefolgt vom Namen der Prozedur und ggf. den Parametern.

```
Call SimpleSub()
```

oder

```
Call CompleteSub( "FirstParameter", $ParamByValue, $ParamByRef )
```

Funktionsaufruf

Um eine Funktion (jede Subroutine, die ein **Return**-Statement enthält) aufzurufen, verwenden Sie einfach deren Namen in einem Ausdruck. Verwenden Sie zum Aufrufen von Funktionen nicht das **Call**-Statement.
Beispiel:

```
$QName = MakeQualifiedName($namespace, "entry")
```

27.8.9.3 Beispiel für eine Subroutine

Das folgende Beispiel enthält die Deklaration und den Aufruf einer Subroutine.

```
[create $outputpath & $module & "output.txt"

' define sub SimpleSub()
Sub SimpleSub()
]SimpleSub() called
[endsub

' execute sub SimpleSub()
Call SimpleSub()

$ParamByValue        = "Original Value"
]ParamByValue        = [=$ParamByValue]
[$ParamByRef  = "Original Value"
]ParamByRef   = [=$ParamByRef]

' define sub CompleteSub()
[Sub CompleteSub( $param, ByVal $paramByValue, ByRef $paramByRef )
]CompleteSub called.
       param = [=$param]
       paramByValue = [=$paramByValue]
```


© 2016-2022 Altova GmbH

```
      paramByRef = [=$paramByRef]
[$ParamByRef = "Local Variable"
$paramByValue = "new value"
$paramByRef = "new value"
]      Set values inside Sub
[$ParamByRef = "Local Variable"
$paramByValue = "new value"
$paramByRef = "new value"
]CompleteSub finished.
[endsub

' run sub CompleteSub()
Call CompleteSub( "FirstParameter", $ParamByValue, $ParamByRef )
]
ParamByValue=[=$ParamByValue]
ParamByRef=[=$ParamByRef]
[
Close
]
```

27.8.10 Vordefinierte Typen

In diesem Abschnitt werden die in vordefinierten Variablen[1260] verwendeten Eigenschaften von integrierten Typen beschrieben, die das geparste Schema beschreiben.

27.8.10.1 Bibliothek

Dieses Objekt steht für die gesamte anhand des XML-Schemas oder der DTD generierte Bibliothek.

Eigenschaft	Typ	Beschreibung
SchemaNamespaces	Namespace[1269] collection	Namespaces in dieser Bibliothek
SchemaFilename	string	Name der XSD- oder DTD-Datei, von der diese Bibliothek abgeleitet wurde
SchemaType	integer	1 für DTD, 2 für XML-Schema
Guid	string	Eine globale eindeutige ID
CodeName	string	Generierter Bibliotheksname (vom Schemadateinamen abgeleitet)

27.8.10.2 Namespace

Pro XML-Schema Namespace wird ein Namespace-Objekt generiert. Schema-Komponenten, die sich in keinem Namespace befinden, werden in ein spezielles Namespace-Objekt mit einer leeren NamespaceURI gelegt.

Beachten Sie: Bei DTD werden Namespaces auch von Attributen abgeleitet, deren Namen mit "xmlns" beginnen.

Eigenschaft	Typ	Beschreibung
CodeName	String	Name für den generierten Code (abgeleitet vom Präfix)
LocalName	String	Namespace-Präfix
NamespaceURI	String	Namespace URI
Types	Type 1270 Collection	Alle in diesem Namespace enthaltenen Typen
Library	Library 1269	Bibliothek, die diesen Namespace enthält

27.8.10.3 Typ

Dieses Objekt stellt einen complex oder simpleType dar. Es dient zum Generieren einer Klasse in der Zielsprache.
Es gibt keinen zusätzlichen Typ pro Bibliothek, der für das Dokument steht, der alle möglichen Root-Elemente als Members hat.
Anonyme Typen haben einen leeren LocalName.

Eigenschaft	Typ	Beschreibung
CodeName	string	Name für generierten Code (wird vom lokalen Namen oder der Parent-Deklaration abgeleitet)
LocalName	string	Ursprünglicher Name im Schema
Namespace	Namespace 1269	Namespace, der diesen Typ enthält
Attributes	Member 1271 collection	Attribute, die in diesem Typ enthalten sind
Elements	Member 1271 collection	Child-Elemente, die in diesem Typ enthalten sind
IsSimpleType	boolean	True bei simpleTypes, False bei complexTypes
IsDerived	boolean	True, wenn dieser Typ von einem anderen Typ abgeleitet wurde, der ebenfalls durch ein Typobjekt dargestellt wird
IsDerivedByExtension	boolean	"true", wenn dieser Typ mittels extension abgeleitet wurde
IsDerivedByRestriction	boolean	"true", wenn dieser Typ mittels restriction abgeleitet wurde

© 2016-2022 Altova GmbH

IsDerivedByUnion	boolean	"true", wenn dieser Typ mittels union abgeleitet wurde
IsDerivedByList	boolean	"true", wenn dieser Typ mittels list abgeleitet wurde
BaseType	Type	Der Basistyp dieses Typs (wenn IsDerived "true" ist)
IsDocumentRootType	boolean	True, wenn dieser Typ für das Dokument selbst steht
Library	Library 1269	Bibliothek, die diesen Typ enthält
IsFinal	boolean	True, wenn als "final" im Schema deklariert
IsMixed	boolean	True, wenn dieser Typ Mixed Content haben kann
IsAbstract	boolean	True, wenn dieser Typ als abstrakt deklariert ist
IsGlobal	boolean	True, wenn dieser Typ im Schema global deklariert ist
IsAnonymous	boolean	True, wenn dieser Typ lokal in einem Element deklariert ist

Nur bei simpleTypes:

Eigenschaft	**Typ**	**Beschreibung**
IsNativeBound	boolean	True, wenn Native Type Binding vorhanden ist
NativeBinding	NativeBinding 1272	Native Binding für diesen Typ
Facets	Facets 1273	Facets dieses Typs
Whitespace	string	Shortcut zur Whitespace Facet

* ComplexTypes mit Textinhalt (dabei handelt es sich um Typen mit Mixed Content und complexType mit simpleContent) haben ein zusätzliches nicht benanntes Attribut-Member, das den Textinhalt repräsentiert.

27.8.10.4 Member

Dieses Objekt steht für ein Attribut oder Element im XML-Schema. Es dient zum Erstellen von Klassen-Members von Typen.

Eigenschaft	**Typ**	**Beschreibung**

CodeName	string	Name für generierten Code (vom lokalen Namen oder der Parent-Deklaration abgeleitet)
LocalName	string	Ursprünglicher Name im Schema. Leer bei dem speziellen Member, das den Textinhalt von complexTypes darstellt.
NamespaceURI	string	Die Namespace URI dieses Elements/Attributs in XML-Instanzdokumenten / -streams.
DeclaringType	Type 1270	Typ, der ursprünglich das Member deklariert (entspricht ContainingType für nicht vererbte Members)
ContainingType	Type 1270	Typ, dessen Member dies ist
DataType	Type 1270	Datentyp des Inhalts dieses Members
Library	Library 1269	Bibliothek, die den Datentyp dieses Members enthält
IsAttribute	boolean	True bei Attributen, False bei Elementen
IsOptional	boolean	True, wenn minOccurs = 0 oder optionales Attribut
IsRequired	boolean	True, wenn minOccurs > 0 oder required Attribut
IsFixed	boolean	True bei fixed Attributen, Wert ist in der Eigenschaft "Default"
IsDefault	boolean	True bei Attributen mit Standardwert, Wert ist in der Eigenschaft "Default"
IsNillable	boolean	True bei nillable Elementen
IsUseQualified	boolean	True, wenn NamespaceURI nicht leer ist
MinOccurs	integer	minOccurs wie in Schema. 1 bei required Attributen
MaxOccurs	integer	maxOccurs, wie in Schema. 0 bei prohibited Attributen, -1 bei unbounded
Default	string	Standardwert

27.8.10.5 NativeBinding

Dieses Objekt steht für das Binding eines simpleType an einen native Type in der Zielprogrammiersprache, wie in der "schemanativetype" Zuordnung angegeben.

Eigenschaft	**Typ**	**Beschreibung**

© 2016-2022 Altova GmbH

ValueType	string	Native Type
ValueHandler	string	Formatierungsklasseninstanz

27.8.10.6 Facets

Dieses Objekt steht für alle Facets eines simpleType.
Vererbte Facets werden mit den explizit deklarierten Facets zusammengeführt. Wenn ein Length Facet gilt, werden MinLength und MaxLength auf denselben Wert gesetzt.

Eigenschaft	**Typ**	**Beschreibung**
DeclaringType	Type	Type Facets werden darauf deklariert
Whitespace	string	"preserve", "collapse" oder "replace"
MinLength	integer	Facet-Wert
MaxLength	integer	Facet-Wert
MinInclusive	integer	Facet-Wert
MinExclusive	integer	Facet-Wert
MaxInclusive	integer	Facet-Wert
MaxExclusive	integer	Facet-Wert
TotalDigits	integer	Facet-Wert
FractionDigits	integer	Facet-Wert
List	Facet collection	Alle Facets als Liste

Facet

Dieses Objekt steht für eine einzelne Facet, deren berechneter Wert für einen bestimmten Typ wirksam wird.

Eigenschaft	**Typ**	**Beschreibung**
LocalName	string	Facet-Name
NamespaceURI	string	Facet Namespace
FacetType	string	Einer der folgenden: "normalization", "lexicalspace", "valuespace-length", "valuespace-enum" oder "valuespace-range"
DeclaringType	Type 1270	Typ, auf den dieses Facet deklariert ist

FacetCheckerName	string	Name des Facet Checker (aus der schemafacet-Zuordnung)
FacetValue	string or integer	Tatsächlicher Wert dieses Facet

© 2016-2022 Altova GmbH

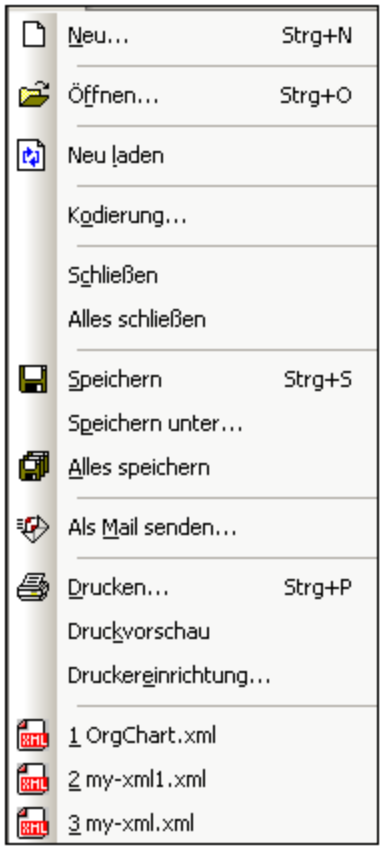
28 Menübefehle

In diesem Abschnitt finden Sie eine Beschreibung aller XMLSpy-Menübefehle. Neben einer allgemeinen Beschreibung eines Befehls, haben wir versucht, wo nötig, entsprechendes Hintergrundwissen dazu zu vermitteln. Falls Sie weitere Fragen haben, die in dieser Dokumentation nicht beantwortet wurden, lesen Sie bitte auf der Altova Website unter FAQs und im Diskussionsforum nach. Sollten Ihre Fragen damit noch immer nicht beantwortet sein, kontaktieren Sie uns bitte über unser Altova Support Center[1657].

Bitte beachten Sie, dass das Menü Datei[1276] und Bearbeiten[1297] alle Windows-Standardbefehle (wie z.B. **Öffnen, Speichern, Ausschneiden, Kopieren** und **Einfügen)** enthält. Diese Menüs enthalten zusätzlich dazu Befehle im Zusammenhang mit XML und Internet.

28.1 Menü "Datei"

Das Menü "Datei" enthält Befehle, die zum Arbeiten mit Dateien benötigt werden. Die Befehle sind in der aus Windows-Produkten bekannten Reihenfolge angeordnet. Zusätzlich zu den Standardbefehlen Neu[1276], Öffnen[1280], Speichern[1286], Drucken[1293], Druckereinrichtung[1295], und Beenden[1296] bietet XMLSpy eine Reihe von XML- und applikationsspezifischen Befehlen.



28.1.1 Neu

In diesem Abschnitt:

- Schaltfläche und Tastenkürzel[1277]
- Beschreibung[1277]
- Vorlagen für neue Dokumente[1277]
- Zuweisen einer DTD oder eines XML-Schemas zu einem neuen XML-Dokument[1278]
- Angabe des Root-Elements eines neuen XML-Dokuments[1279]
- Zuweisen eines SPS zu einem neuen XML-Dokument[1279]
- Erstellen neuer XBRL-Taxonomien mit dem XBRL-Taxonomieassistenten[1280]

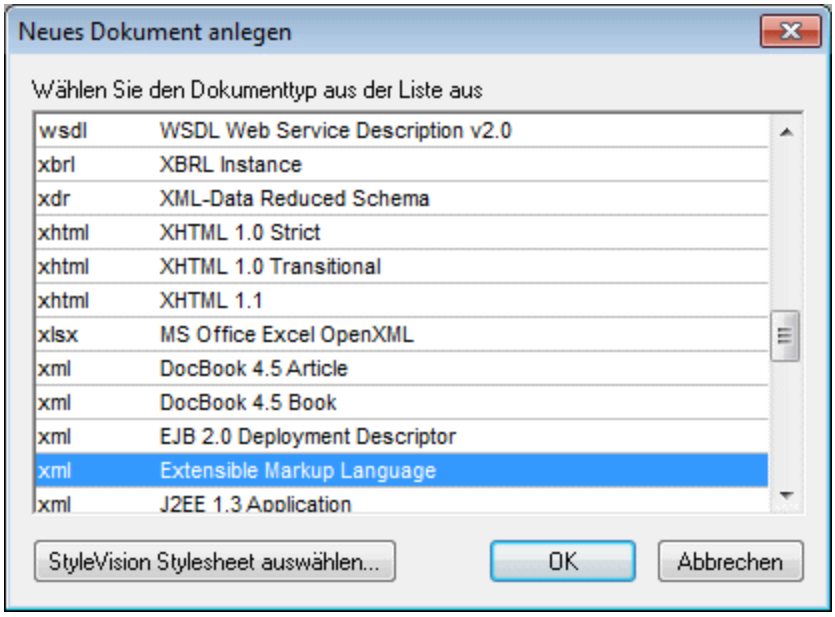
© 2016-2022 Altova GmbH

Schaltfläche und Tastenkürzel

Symbol:	
Tastenkürzel:	**Strg+N**

Beschreibung

Der Befehl **Neu** wird verwendet, um ein neues Dokument zu erstellen. Wenn Sie auf **Neu** klicken, wird das Dialogfeld "Neues Dokument anlegen" geöffnet, in dem Sie auswählen können, welche Art von Dokument erstellt werden soll. Falls das Dokument, das Sie erstellen wollen, nicht aufgelistet ist, wählen Sie "XML", und ändern Sie die Erweiterung des Dokuments beim Speichern. Beachten Sie, dass Sie neue Dateiarten in die Liste einfügen können, indem Sie diese unter Extras | Optionen im Abschnitt "Dateiarten" 1606 eintragen.



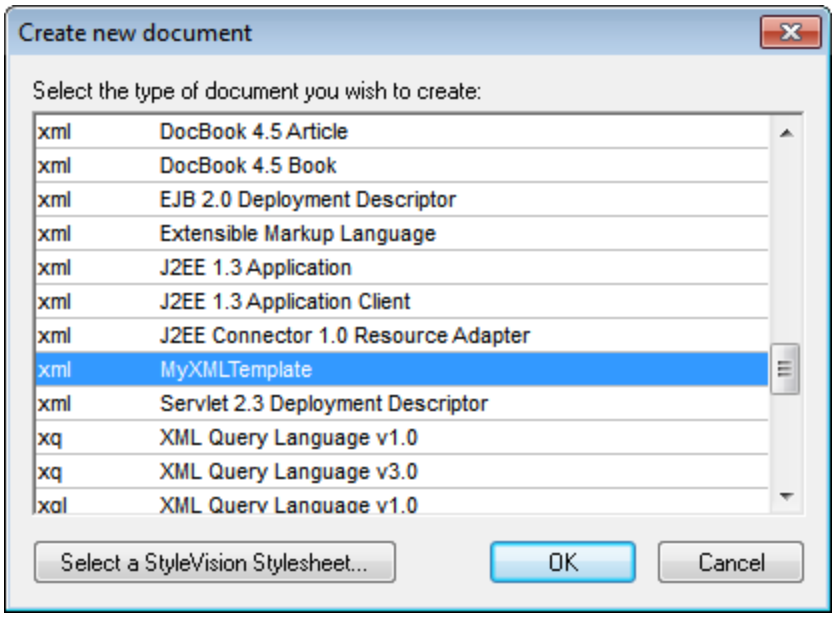
Vorlagen für neue Dokumente

Die Liste der Dokumentarten im Dialogfeld "Neues Dokument anlegen" kann auch Einträge für benutzerdefinierte Dokumentvorlagen jedes beliebigen Dokumenttyps enthalten. Diese Vorlagen können direkt über das Dialogfeld "Neues Dokument anlegen" geöffnet und bearbeitet werden. Um Ihre eigene Dokumentvorlage zu erstellen, sodass sie in diesem Dialogfeld in der Liste der Dokumenttypen angezeigt wird, müssen Sie zuerst ein Vorlagendokument erstellen und dieses dann in dem für die Dokumentvorlagen vorgesehenen Ordner speichern.

Um eine Dokumentvorlage zu erstellen, gehen Sie vor wie folgt:

1. Öffnen Sie den Ordner `XMLSpy\Template` aus dem Applikationsordner 34 über den Windows Explorer oder Ihr bevorzugtes Navigationstool und wählen Sie aus den mit **`new.xxx`** benannten Dateien (wobei `.xxx` eine Dateierweiterung wie z.B. `.xml` und `.xslt` ist) eine rudimentäre Vorlagendatei aus.

2. Öffnen Sie die Datei in XMLSpy und ändern Sie die Datei Ihren Wünschen entsprechend. Diese Datei wird dann als Vorlage verwendet.
3. Wählen Sie anschließend **Datei | Speichern unter** um die Datei wieder im Ordner `\Template` unter einem passenden Namen z.B. `MyXMLTemplate.xml` zu speichern. Sie haben nun eine Dokumentvorlage mit dem Namen `my-xml.xml`, die im Dialogfeld "Neues Dokument anlegen" in der Liste der Dokumenttypen angezeigt wird.



4. Zum Öffnen der Vorlage wählen Sie **Datei | Neu** und anschließend die Vorlage (in diesem Fall `my-xml`).

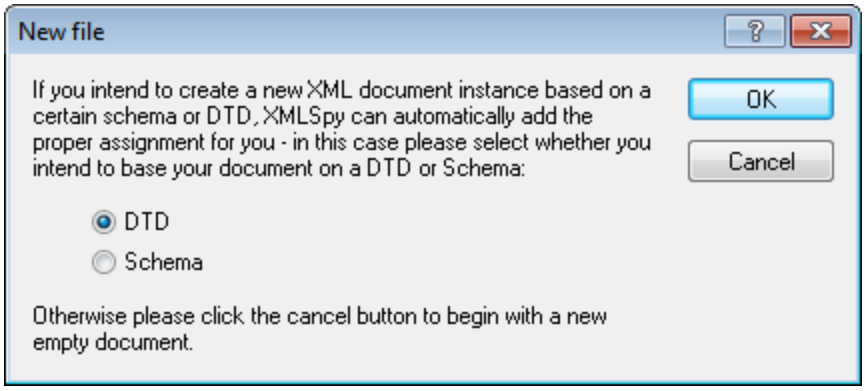
Um eine Dokumentvorlage aus der Liste der Dokumenttypen zu löschen, löschen Sie die Vorlagendatei aus dem Vorlagenordner (oder verschieben Sie diese in einen anderen Ordner).

Zuweisen einer DTD oder eines XML-Schemas zu einem neuen XML-Dokument

Wenn Sie über das Dialogfeld "Neues Dokument anlegen" ein neues Dokument einer bestimmten Art erstellen, wird das Dokument automatisch mit der korrekten DTD oder XML-Schema-Verknüpfung geöffnet. So wird z.B. eine XHTML-Datei mit der damit verknüpften DTD `http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd` geöffnet. Eine XML-Schema-Datei (`.xsd`) wird mit dem Schemadokument `http://www.w3.org/2001/XMLSchema` verknüpft.

Wenn Sie ein neues Dokument erstellen, für das das Schema nicht bekannt ist (z.B. eine XML-Datei), werden Sie aufgefordert, ein Schema (DTD oder XML-Schema) damit zu verknüpfen. (*Abbildung unten*)

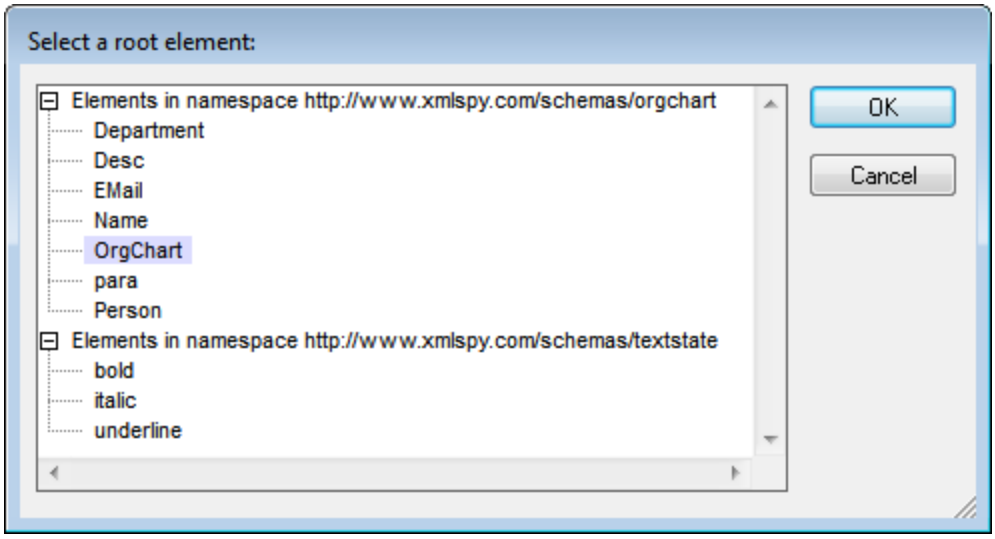
© 2016-2022 Altova GmbH



Wenn Sie im Dialogfeld eine DTD oder ein XML-Schema auswählen und auf **OK** klicken, können Sie zum gewünschten Schema navigieren. Wenn Sie in diesem Dialogfeld auf **Abbrechen** klicken, wird eine neue Datei erstellt, die mit keinem Schema verknüpft ist.

Angabe des Root-Elements eines neuen XML-Dokuments

Wenn Sie als das mit dem XML-Dokument verknüpfte Schema ein XML-Schema auswählen und wenn dieses Schema mehr als ein globales Element enthält, so kann potenziell jedes dieser globalen Elemente ein Root-Element sein. In diesem Fall wird das Dialogfeld "Wählen Sie das Root-Element" angezeigt. Hier können Sie auswählen, welches globale Element als Root-Element für das XML-Dokument verwendet werden soll. In der Abbildung unten wurde das globale Element `OrgChart` ausgewählt.

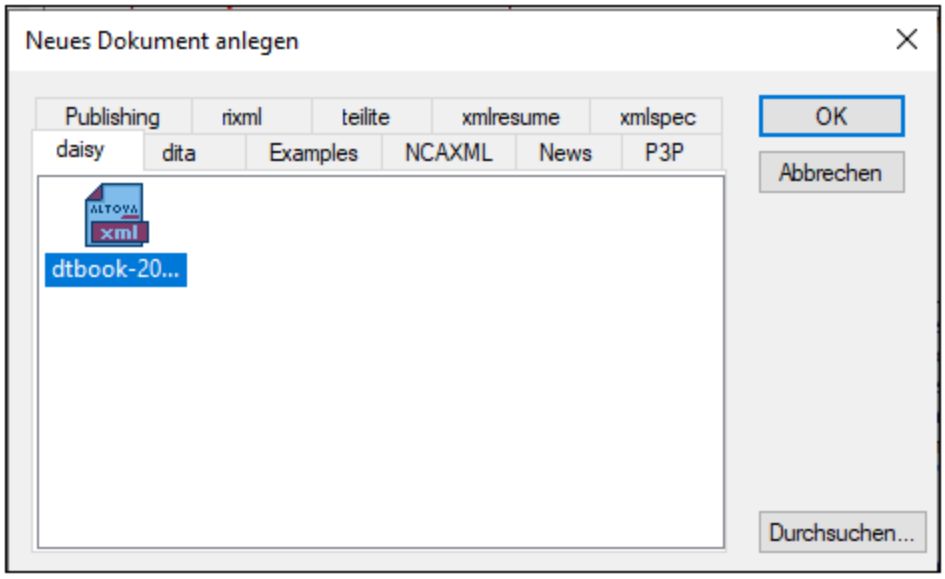


Wenn Sie auf OK klicken, wird ein neues XML-Dokument mit diesem Element (`OrgChart`) als Root-Element erstellt.

Zuweisen eines SPS zu einem neuen XML-Dokument

Bei Erstellung eines neuen XML-Dokuments können Sie es mit einem StyleVision Power Stylesheet (`.sps`-Datei) verknüpfen, um es in der Authentic-Ansicht anzeigen zu können. Wenn Sie im Dialogfeld "Neues

Dokument anlegen" (*siehe erste Abbildung in diesem Abschnitt*) auf die Schaltfläche **StyleVision Stylesheet auswählen** klicken, erscheint das Dialogfeld "Neues Dokument anlegen" (*siehe Abbildung unten*).



Auf den verschiedenen Ordner-Registern dieses Dialogfelds stehen Ihnen die gewünschten SPS-Dateien zur Verfügung. Alternativ dazu können Sie das gewünschte SPS auch über die Schaltfläche **Durchsuchen...** suchen.

Erstellen neuer XBRL-Taxonomien mit dem XBRL-Taxonomieassistenten

Wenn Sie im Dialogfeld "Neues Dokument anlegen" den Eintrag "XBRL Taxonomy Schema (`.xsd`)" auswählen, führt Sie ein Assistent Schritt für Schritt durch die Erstellung einer neuen XBRL-Taxonomie. Dieser Assistent wird in dieser Dokumentation im Abschnitt XBRL[851] beschrieben.

28.1.2 Öffnen

Schaltfläche und Tastenkürzel

Schaltfläche:	
Tastenkürzel:	**Strg+O**

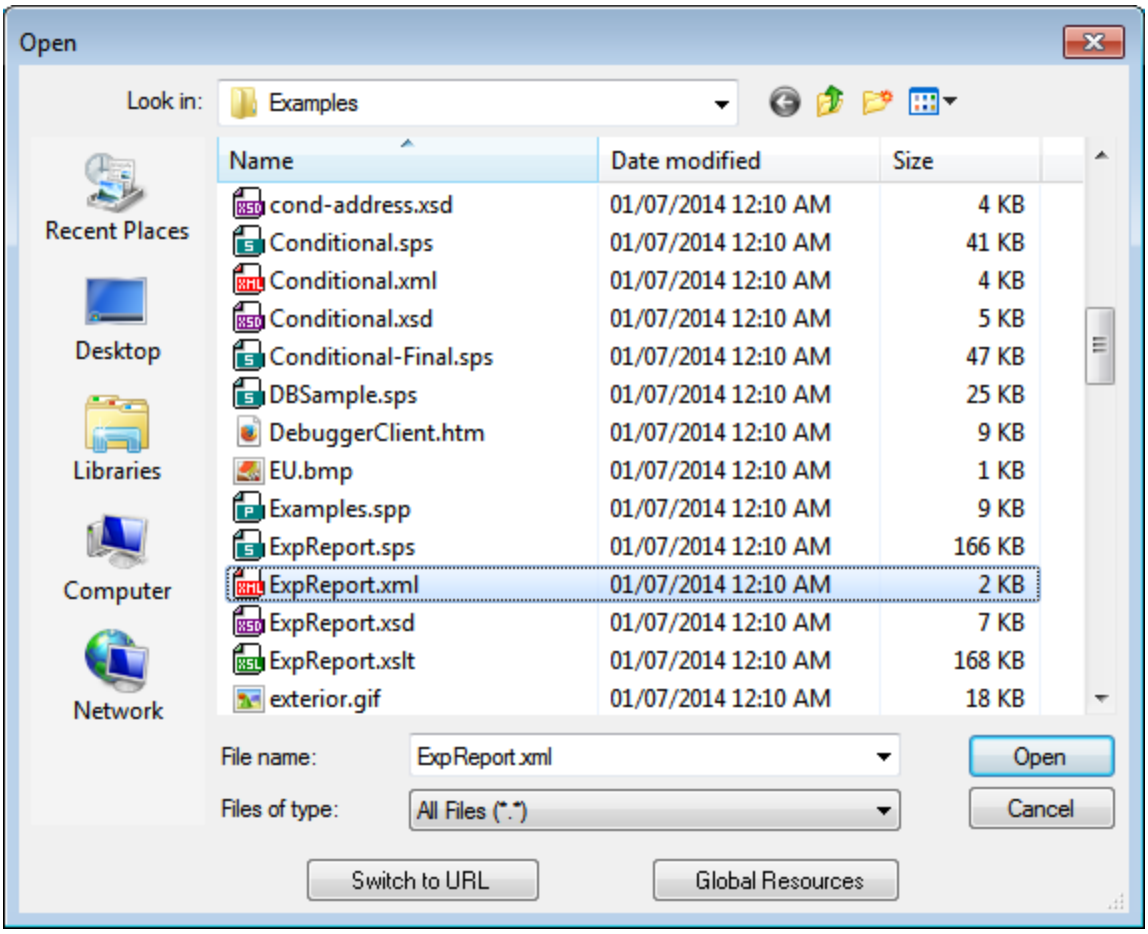
Beschreibung

Mit **Öffnen** rufen Sie das bekannte Windows-Dialogfeld "Öffnen" auf, über das Sie jedes XML-Dokument oder jede Textdatei öffnen können. Sie haben auch die Möglichkeit, mehrere Dateien auszuwählen. Im Listenfeld "Dateityp" können Sie auswählen, welche Dateien im Dialogfeld angezeigt werden sollen (die Liste der verfügbaren Dateiarten kann auf dem Register "Dateiarten" unter **Extras | Optionen**[1606] eingestellt werden). Beim Öffnen einer XML-Datei wird diese auf Wohlgeformtheit überprüft. Ist sie nicht wohlgeformt, wird eine Fehlermeldung angezeigt. Beheben Sie den Fehler und wählen Sie den Menübefehl **XML | Wohngeformtheit prüfen (F7)**[1351], um die Datei erneut zu überprüfen. Wenn Sie die Option Validierung beim Öffnen der Datei[1603]

aktiviert haben und die Datei ungültig ist, wird ebenfalls eine Fehlermeldung angezeigt. Beheben Sie den Fehler und klicken Sie auf **XML | XML validieren (F8)**[1353], um das Dokument erneut zu validieren.

▼ Auswählen und Speichern von Dateien über URLs und globale Ressourcen

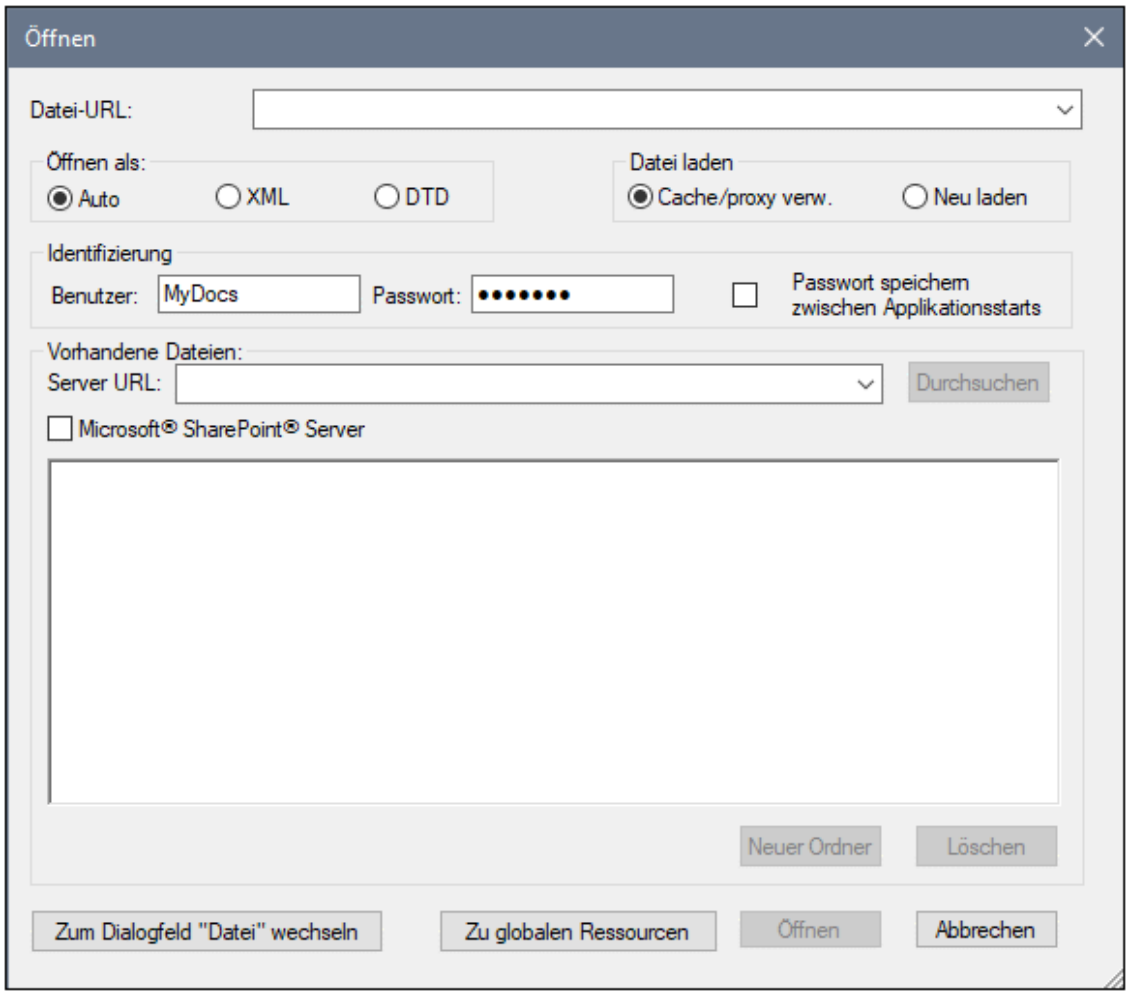
Es gibt mehrere Dialogfelder zum Öffnen und Speichern von Dateien, über die Sie die gewünschte Datei entweder über eine URL oder eine globale Ressource (*siehe Abbildung unten*) auswählen bzw. speichern können. Wählen Sie den Befehl **Zu URL wechseln** oder **Globale Ressourcen**, um das entsprechende Dialogfeld aufzurufen.



Auswählen von Dateien über URLs

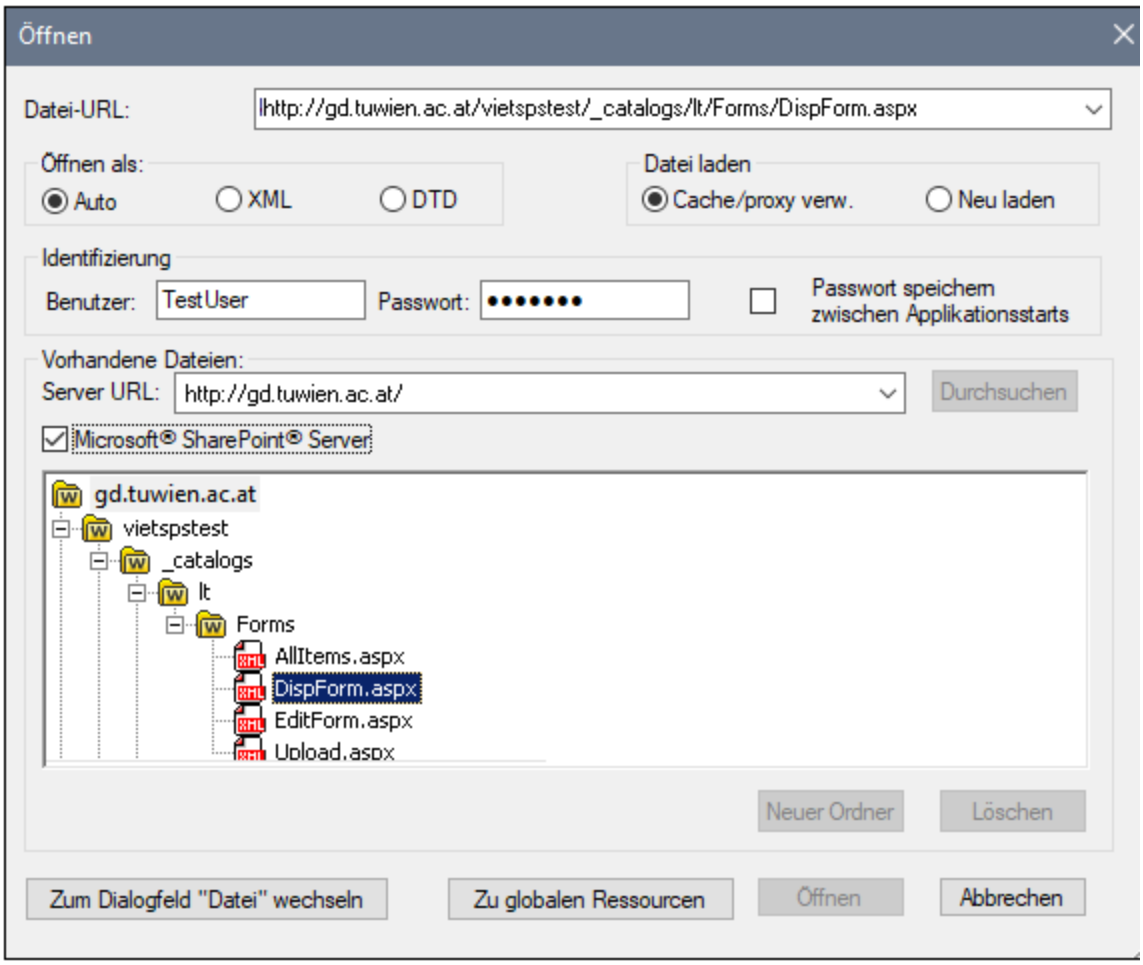
Um eine Datei über eine URL auszuwählen, gehen Sie folgendermaßen vor:

1. Klicken Sie auf den Befehl **Zu URL wechseln**. Daraufhin wechselt das Dialogfeld "Öffnen" bzw. "Speichern" in den URL-Modus (*In der Abbildung unten sehen Sie das Dialogfeld "Öffnen"*).



2. Geben Sie die gewünschte URL in das Feld *Server URL* ein (*Abbildung oben*). Wenn es sich beim Server um einen Microsoft® SharePoint® Server handelt, aktivieren Sie das Kontrollkästchen *Microsoft® SharePoint® Server.* Nähere Informationen zum Arbeiten mit Dateien dieses Servertyps finden Sie unter den Anmerkungen weiter unten zu Microsoft® SharePoint® Server.
3. Geben Sie Ihre Benutzer-ID und Ihr Passwort in die Felder *Benutzer* und *Passwort* ein, wenn der Server passwortgeschützt ist.
4. Klicken Sie auf **Durchsuchen**, um die Verzeichnisstruktur des Servers zu sehen und darin zu navigieren.
5. Suchen Sie in die Ordnerstruktur Sie die gewünschte Datei und klicken Sie darauf.

© 2016-2022 Altova GmbH



Die Datei-URL wird im Feld "Datei-URL" angezeigt (*Abbildung oben*). Die Schaltflächen **Öffnen** oder **Speichern** werden erst zu diesem Zeitpunkt aktiv.

6. Klicken Sie auf **Öffnen,** um die Datei zu laden bzw. auf **Speichern**, um sie zu speichern.

Beachten Sie bitte die folgenden Punkte:

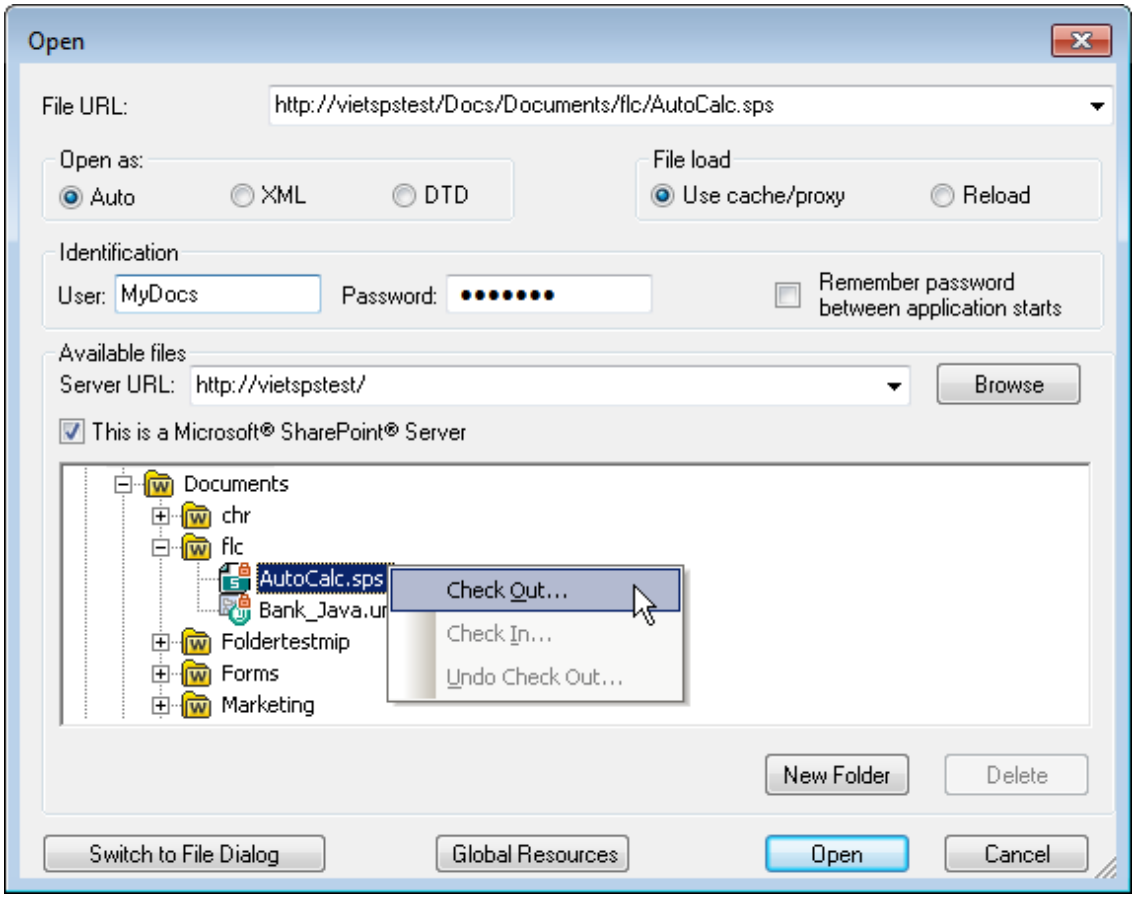
- Die Durchsuchen-Funktion steht nur auf Servern zur Verfügung, die WebDAV und Microsoft SharePoint Server unterstützen. Die unterstützten Protokolle sind FTP, HTTP und HTTPS.
- Damit Sie mehr Kontrolle über den Ladevorgang haben, können Sie auswählen, ob die Datei über den lokalen Cache-Speicher oder über einen Proxy-Server (wodurch der Ladevorgang beträchtlich schneller geht, wenn die Datei bereits einmal geladen war) geladen werden soll. Alternativ dazu müssen Sie die Datei eventuell neu laden, wenn Sie z.B. mit einem EPS oder einem Datenbanksystem arbeiten; wählen Sie in diesem Fall die Option **Neu laden**.

.

▼ Anmerkungen zu Microsoft® SharePoint® Server

Beachten Sie bei Dateien auf Microsoft® SharePoint® Servern die folgenden Punkte:

- In der Verzeichnisstruktur, die im Fenster der verfügbaren Dateien angezeigt wird (*Abbildung unten*), sind die Dateisymbole mit Symbolen versehen, die den Status der Datei (Eingecheckt/Ausgecheckt) kennzeichnen.



Wenn Sie mit der rechten Maustaste auf eine Datei klicken, erscheint ein Kontextmenü mit Befehlen, die für diese Datei zur Verfügung stehen (*Abbildung oben*).

- Im Folgenden sehen Sie die verschiedenen Dateisymbole:

	Eingecheckt. Die Datei kann ausgecheckt werden.
	Von einem anderen Benutzer ausgecheckt. Kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Kann bearbeitet und eingecheckt werden.

- Nachdem Sie eine Datei ausgecheckt haben, können Sie sie in Ihrer Altova-Applikation bearbeiten und mit **Datei | Speichern (Strg+S)** speichern.
- Sie können die bearbeitete Datei über das Kontextmenü im Dialogfeld "URL öffnen" (*siehe Abbildung oben*) oder über das Kontextmenü, das angezeigt wird, wenn Sie mit der rechten Maustaste auf das Dateiregister im Hauptfenster Ihrer Applikation klicken (*Abbildung unten*), einchecken.

© 2016-2022 Altova GmbH



- Wenn eine Datei von einem anderen Benutzer ausgecheckt wurde, kann sie nicht auscheckt werden.
- Wenn eine Datei lokal von Ihnen ausgecheckt wurde, können Sie dies mit dem Kontextmenübefehl "Auschecken rückgängig" rückgängig machen. In diesem Fall wird die Datei dem Server unverändert zurückgegeben.
- Wenn Sie eine Datei in einer Altova-Applikation auschecken, können Sie sie nicht in einer anderen Altova-Applikation auschecken. Die Datei gilt bereits als für Sie ausgecheckt. Daher stehen in diesem Fall in jeder Altova-Applikation, die Microsoft® SharePoint® Server unterstützt, nur die Befehle **Einchecken** und **Auschecken rückgängig** zur Verfügung.

▼ Öffnen und Speichern von Dateien über globale Ressourcen

Um eine Datei über eine globale Ressource zu öffnen oder zu speichern, klicken Sie auf **Globale Ressourcen**. Daraufhin wird ein Dialogfeld angezeigt, in dem Sie die globale Ressource auswählen können. Diese Dialogfelder werden im Abschnitt Verwenden globaler Ressourcen[1074] näher beschrieben. Eine allgemeine Beschreibung globaler Ressourcen finden Sie in der Dokumentation im Abschnitt Globale Ressourcen[1061].

28.1.3 Neu laden

Schaltfläche

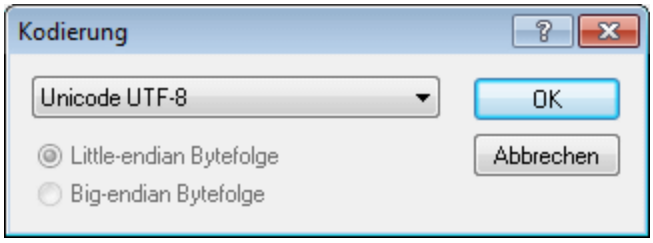
Schaltfläche:	

Beschreibung

Mit **Neu laden** können Sie geöffnete Dokumente, die außerhalb von XMLSpy geändert wurden, neu laden. Wenn ein oder mehrere Dokumente außerhalb von XMLSpy geändert wurden, werden Sie von XMLSpy gefragt, ob die Datei erneut geladen werden soll. Wenn Sie dies tun, gehen alle Änderungen, die Sie an der Datei seit der letzten Speicherung vorgenommen haben, verloren.

28.1.4 Kodierung

Mit **Kodierung...** können Sie (i) die Kodierung der in Verwendung befindlichen Datei (XML oder nicht XML) ansehen und (ii) eine andere Kodierung auswählen, mit der das aktive Dokument beim nächsten Mal gespeichert wird.



Wenn Sie eine andere Kodierung als die derzeit verwendete auswählen, wird das Kodierungsattribut der XML-Deklaration entsprechend geändert. Bei 2-Byte und 4-Byte-Zeichenkodierungen (UTF-16, UCS-2 und UCS-4) können Sie weiters die zu verwendende Bytefolge in der Datei angeben. Eine andere Methode, um die Kodierung eines XML-Dokuments zu ändern, ist, das Kodierungsattribut direkt in der XML-Deklaration des Dokuments zu ändern.

Die Standardkodierung für vorhandene und neue XML- und nicht-XML-Dokumente kann im Dialogfeld "Optionen" im Abschnitt "Kodierung" [1608] festgelegt werden.

Anmerkung: Beim Speichern eines Dokumentes prüft XMLSpy automatisch die Kodierungsspezifikation. Über das Dialogfeld "Kodierung" können Sie die gewünschte Kodierung auswählen. Sollte Ihr Dokument Zeichen enthalten, die in der gewählten Kodierung nicht angezeigt werden können, erhalten Sie eine Warnmeldung beim Speichern der Datei.

28.1.5 Schließen, Alle schließen, Alle inaktiven schließen

Schließen

Mit **Schließen** beenden Sie das aktive Dokument. Sollten Sie die Datei verändert haben (in der Titelleiste wird der Dateiname mit einem Sternchen "*" angezeigt), werden Sie gefragt, ob Sie die Datei vor dem Schließen speichern möchten.

Alle schließen

Mit **Alle schließen** beenden Sie alle Dokumente. Sollten Sie Dateien verändert haben (in der Titelleiste wird der Dateiname mit einem Sternchen "*" angezeigt), werden Sie gefragt, ob Sie die Dateien vor dem Schließen speichern möchten.

Alle inaktiven schließen

Mit **Alle inaktiven schließen** beenden Sie alle Dokumente außer das aktive Dokument. Sollten Sie Dateien verändert haben (in der Titelleiste wird der Dateiname mit einem Sternchen "*" angezeigt), werden Sie gefragt, ob Sie die Dateien vor dem Schließen speichern möchten.

28.1.6 Speichern, Speichern unter, Alles speichern

Schaltflächen und Tastenkürzel

© 2016-2022 Altova GmbH

Befehl	*Schaltfläche*	*Tastenkürzel*
Speichern		**Strg+S**
Alles speichern		

Speichern

Mit dem Befehl **Speichern** (**Strg+S)** wird der Inhalt des aktiven Dokuments in der geöffneten Datei gespeichert. Beim Speichern wird das Dokument automatisch auf Wohlgeformtheit geprüft[1351]. Außerdem wird das Dokument auch automatisch validiert, wenn diese Option im Dialogfeld "Optionen" (**Extras | Optionen**[1603]) im Abschnitt "Datei" aktiviert wurde. Die XML Deklaration wird auch auf Kodierung[1608]sspezifikationen geprüft. Diese Kodierung wird beim Speichern des Dokuments verwendet.

Speichern unter

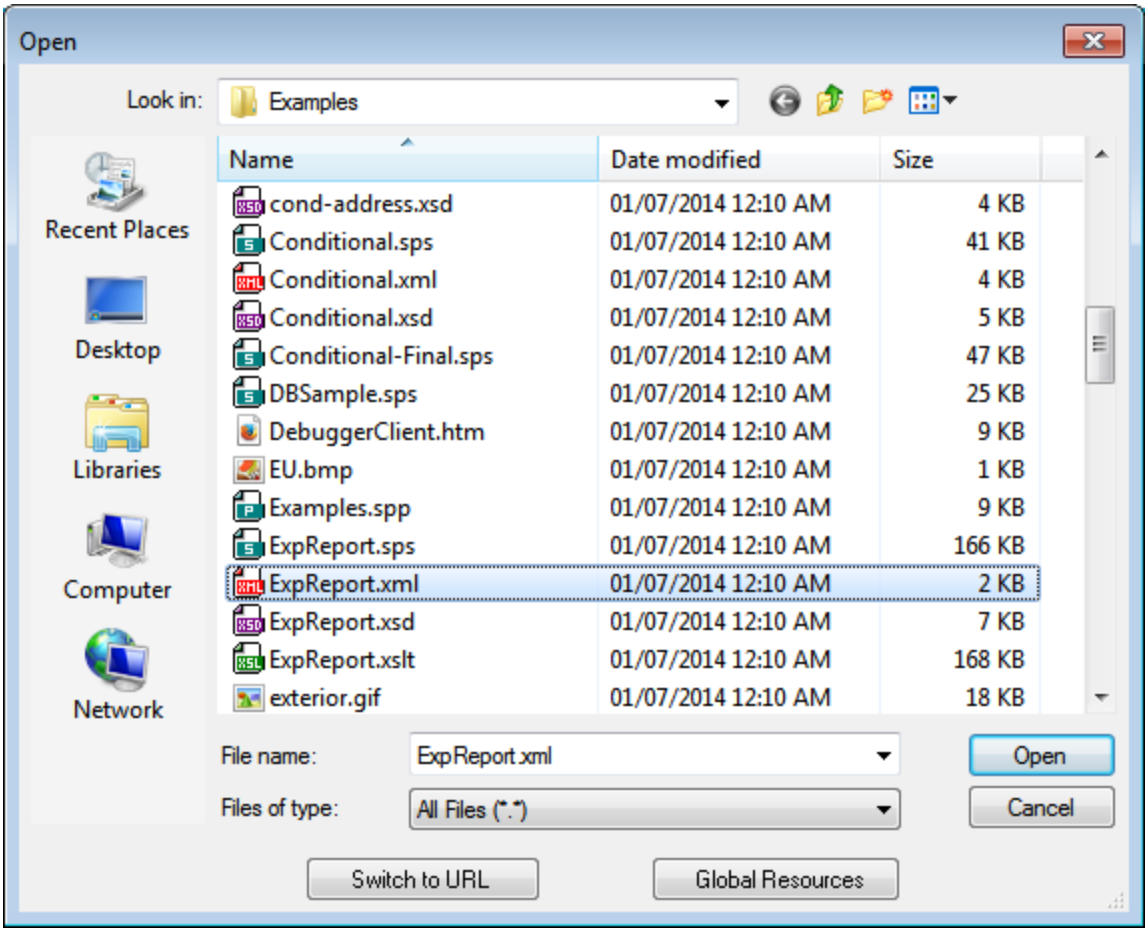
Mit dem Befehl **Speichern unter** gelangen Sie zu dem aus Windows bekannten Dialogfeld "Speichern unter...". Hier geben Sie den Namen und den Ordner an, in dem die Datei gespeichert werden soll. Bei Verwendung des Befehls **Speichern unter** werden dieselben Validierungen und Prüfungen durchgeführt, wie bei dem Befehl **Speichern.**

Alles speichern

Mit dem Befehl **Alles speichern** werden alle Änderungen gespeichert, die Sie an offenen Dokumenten durchgeführt haben. Dieser Befehl ist hilfreich, wenn Sie gleichzeitig an mehreren Dokumenten arbeiten. Falls ein zu speicherndes Dokument bisher noch nie gespeichert wurde (z.B. nachdem es neu erstellt wurde), wird das Dialogfeld "Speichern unter..." für dieses Dokument angezeigt.

▼ Auswählen und Speichern von Dateien über URLs und globale Ressourcen

Es gibt mehrere Dialogfelder zum Öffnen und Speichern von Dateien, über die Sie die gewünschte Datei entweder über eine URL oder eine globale Ressource (*siehe Abbildung unten*) auswählen bzw. speichern können. Wählen Sie den Befehl **Zu URL wechseln** oder **Globale Ressourcen**, um das entsprechende Dialogfeld aufzurufen.

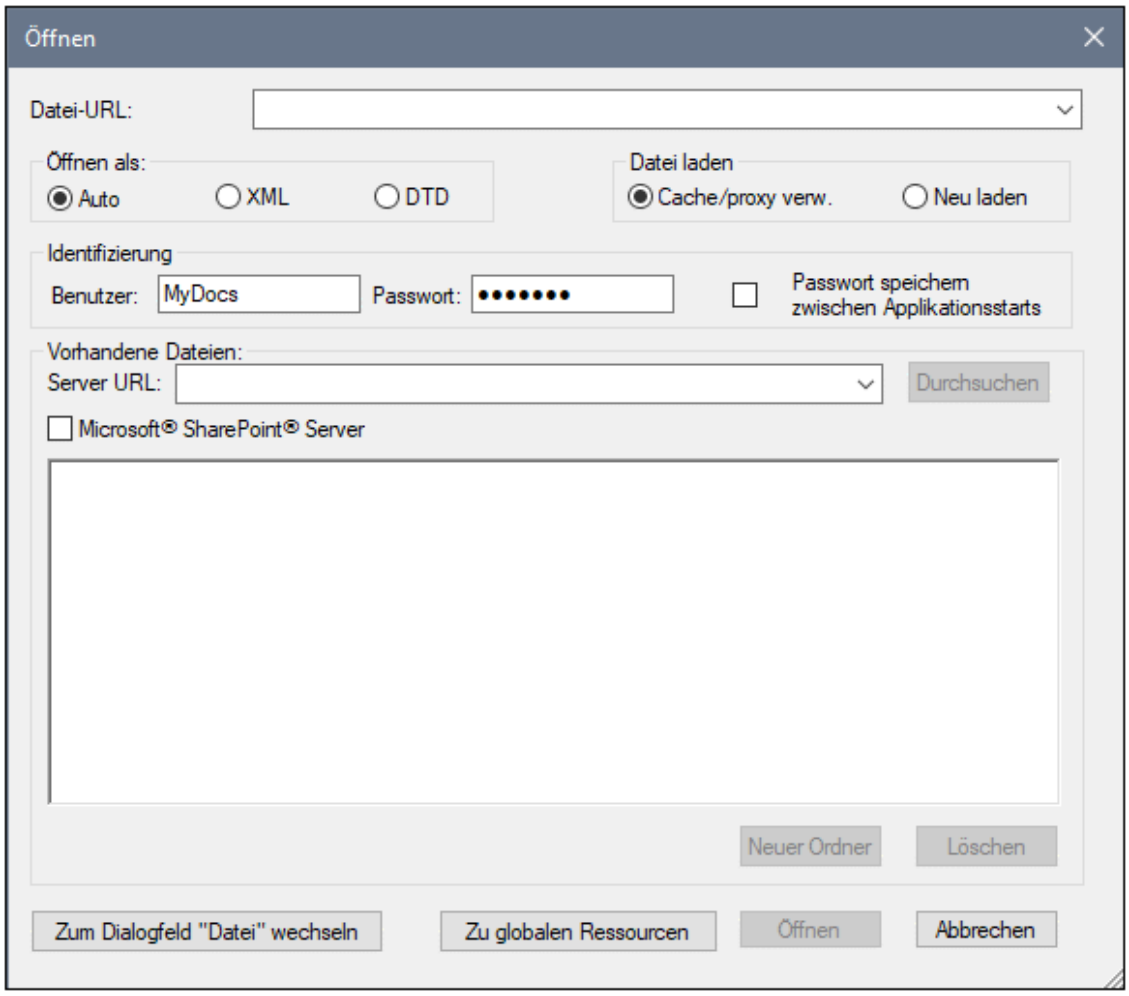


Auswählen von Dateien über URLs

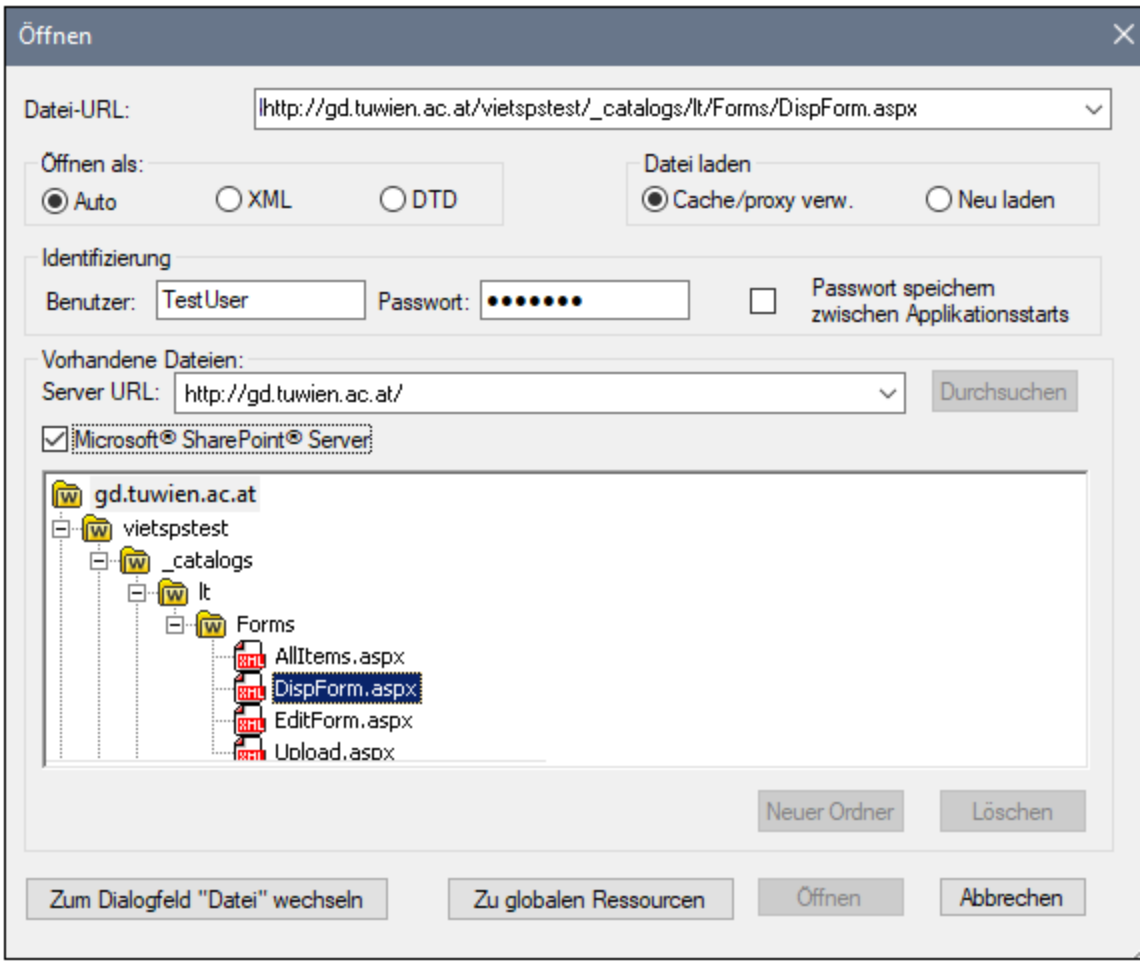
Um eine Datei über eine URL auszuwählen, gehen Sie folgendermaßen vor:

1. Klicken Sie auf den Befehl **Zu URL wechseln**. Daraufhin wechselt das Dialogfeld "Öffnen" bzw. "Speichern" in den URL-Modus (*In der Abbildung unten sehen Sie das Dialogfeld "Öffnen"*).

© 2016-2022 Altova GmbH



2. Geben Sie die gewünschte URL in das Feld *Server URL* ein (*Abbildung oben*). Wenn es sich beim Server um einen Microsoft® SharePoint® Server handelt, aktivieren Sie das Kontrollkästchen *Microsoft® SharePoint® Server.* Nähere Informationen zum Arbeiten mit Dateien dieses Servertyps finden Sie unter den Anmerkungen weiter unten zu Microsoft® SharePoint® Server.
3. Geben Sie Ihre Benutzer-ID und Ihr Passwort in die Felder *Benutzer* und *Passwort* ein, wenn der Server passwortgeschützt ist.
4. Klicken Sie auf **Durchsuchen**, um die Verzeichnisstruktur des Servers zu sehen und darin zu navigieren.
5. Suchen Sie in die Ordnerstruktur Sie die gewünschte Datei und klicken Sie darauf.



Die Datei-URL wird im Feld "Datei-URL" angezeigt (*Abbildung oben*). Die Schaltflächen **Öffnen** oder **Speichern** werden erst zu diesem Zeitpunkt aktiv.

6. Klicken Sie auf **Öffnen,** um die Datei zu laden bzw. auf **Speichern**, um sie zu speichern.

Beachten Sie bitte die folgenden Punkte:

- Die Durchsuchen-Funktion steht nur auf Servern zur Verfügung, die WebDAV und Microsoft SharePoint Server unterstützen. Die unterstützten Protokolle sind FTP, HTTP und HTTPS.
- Damit Sie mehr Kontrolle über den Ladevorgang haben, können Sie auswählen, ob die Datei über den lokalen Cache-Speicher oder über einen Proxy-Server (wodurch der Ladevorgang beträchtlich schneller geht, wenn die Datei bereits einmal geladen war) geladen werden soll. Alternativ dazu müssen Sie die Datei eventuell neu laden, wenn Sie z.B. mit einem EPS oder einem Datenbanksystem arbeiten; wählen Sie in diesem Fall die Option **Neu laden**.

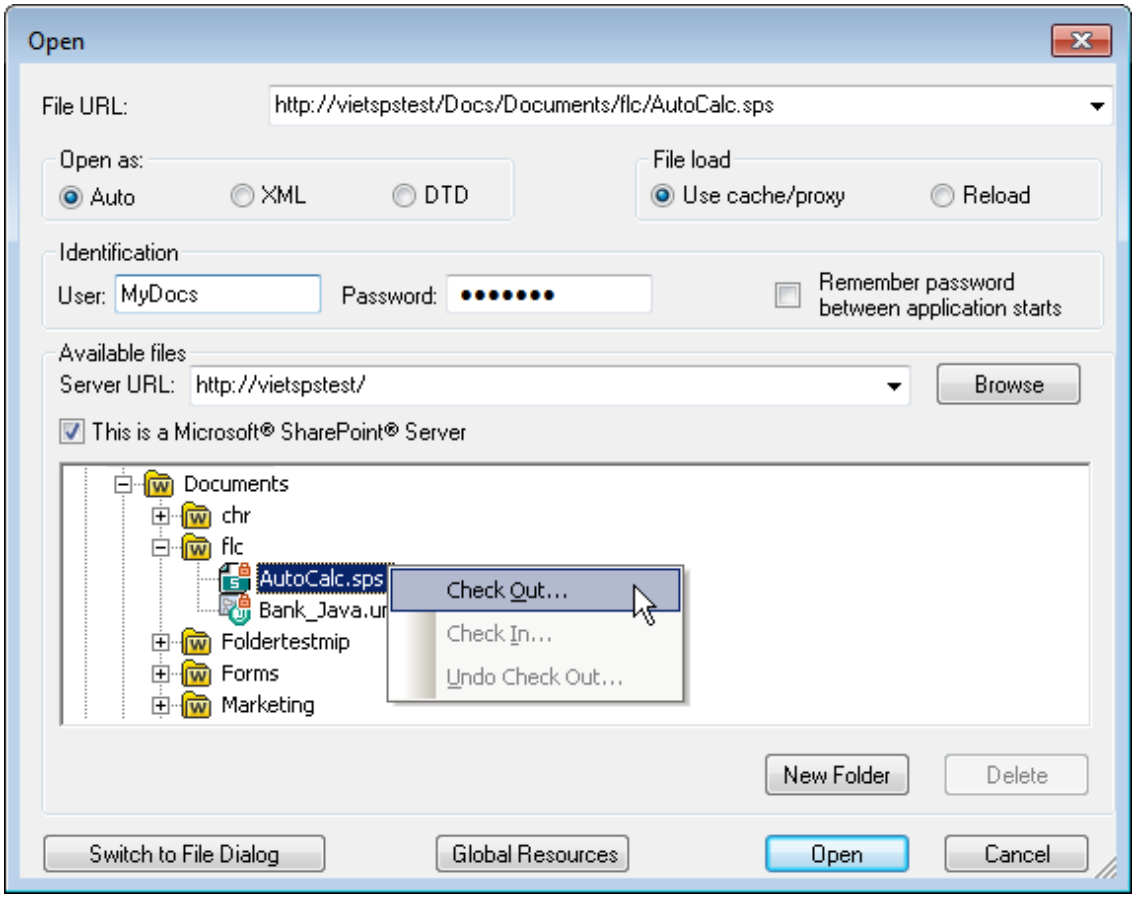
.

▼ Anmerkungen zu Microsoft® SharePoint® Server

Beachten Sie bei Dateien auf Microsoft® SharePoint® Servern die folgenden Punkte:

© 2016-2022 Altova GmbH

- In der Verzeichnisstruktur, die im Fenster der verfügbaren Dateien angezeigt wird (*Abbildung unten*), sind die Dateisymbole mit Symbolen versehen, die den Status der Datei (Eingecheckt/Ausgecheckt) kennzeichnen.



Wenn Sie mit der rechten Maustaste auf eine Datei klicken, erscheint ein Kontextmenü mit Befehlen, die für diese Datei zur Verfügung stehen (*Abbildung oben*).

- Im Folgenden sehen Sie die verschiedenen Dateisymbole:

	Eingecheckt. Die Datei kann ausgecheckt werden.
	Von einem anderen Benutzer ausgecheckt. Kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Kann bearbeitet und eingecheckt werden.

- Nachdem Sie eine Datei ausgecheckt haben, können Sie sie in Ihrer Altova-Applikation bearbeiten und mit **Datei | Speichern (Strg+S)** speichern.
- Sie können die bearbeitete Datei über das Kontextmenü im Dialogfeld "URL öffnen" (*siehe Abbildung oben*) oder über das Kontextmenü, das angezeigt wird, wenn Sie mit der rechten Maustaste auf das Dateiregister im Hauptfenster Ihrer Applikation klicken (*Abbildung unten*), einchecken.



- Wenn eine Datei von einem anderen Benutzer ausgecheckt wurde, kann sie nicht auscheckt werden.
- Wenn eine Datei lokal von Ihnen ausgecheckt wurde, können Sie dies mit dem Kontextmenübefehl "Auschecken rückgängig" rückgängig machen. In diesem Fall wird die Datei dem Server unverändert zurückgegeben.
- Wenn Sie eine Datei in einer Altova-Applikation auschecken, können Sie sie nicht in einer anderen Altova-Applikation auschecken. Die Datei gilt bereits als für Sie ausgecheckt. Daher stehen in diesem Fall in jeder Altova-Applikation, die Microsoft® SharePoint® Server unterstützt, nur die Befehle **Einchecken** und **Auschecken rückgängig** zur Verfügung.

▼ Öffnen und Speichern von Dateien über globale Ressourcen

Um eine Datei über eine globale Ressource zu öffnen oder zu speichern, klicken Sie auf **Globale Ressourcen**. Daraufhin wird ein Dialogfeld angezeigt, in dem Sie die globale Ressource auswählen können. Diese Dialogfelder werden im Abschnitt Verwenden globaler Ressourcen[1074] näher beschrieben. Eine allgemeine Beschreibung globaler Ressourcen finden Sie in der Dokumentation im Abschnitt Globale Ressourcen[1061].

28.1.7 Als Mail senden

Schaltfläche

Schaltfläche:	

Beschreibung

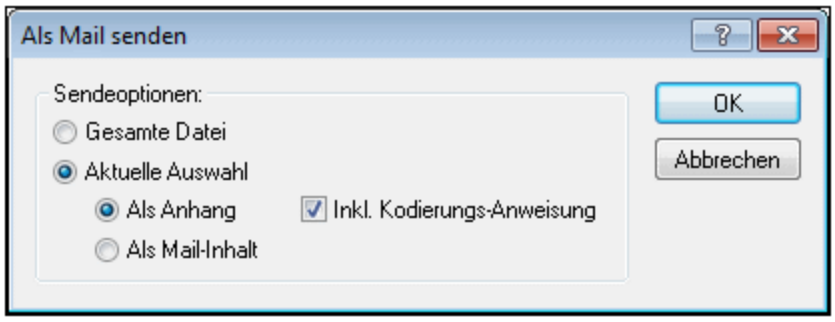
Mit dem Befehl **Als Mail senden** können Sie jede XML-Datei oder Teile einer XML-Datei als E-Mail versenden. Je nachdem ob es sich um ein Dokument oder einen Teil eines Dokuments handelt, können Sie die Informationen als Anhang, Inhalt oder Link senden (siehe Tabelle unten).

Was kann gesendet werden	Wie kann es versendet werden
Aktives XML-Dokument	Als E-Mail-Anhang
Auswahl im aktiven XML-Dokument	Als E-Mail-Anhang oder E-Mail-Inhalt
Eine oder mehrere Dateien im Projektfenster	Als E-Mail-Anhang

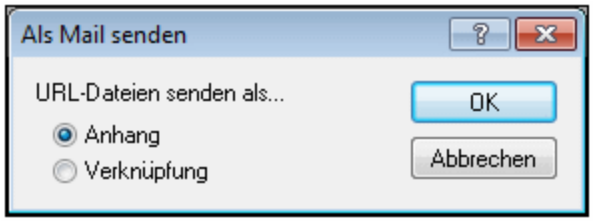
© 2016-2022 Altova GmbH

Eine oder mehrere URLs im Projektfenster	Als E-Mail-Anhang oder Link

Wenn der Befehl **Als Mail senden** auf einen ausgewählten Bereich im aktiven XML-Dokument angewendet wird, wird das Dialogfeld "Als Mail senden" (*Abbildung unten*) aufgerufen. Sie können darin die in der Abbildung unten gezeigten Optionen auswählen. Wenn der Befehl **Als Mail senden** ausgewählt wird, während in der aktiven Datei kein Text ausgewählt ist, ist nur das Optionsfeld *Gesamte Datei* (*siehe Abbildung oben*) aktiv; die anderen Optionen sind deaktiviert.



Da über das Fenster "Projekt" gesendete Dateien immer nur als E-Mail-Anhänge versendet werden, wird das Dialogfeld "Als Mail senden" in diesem Fall übersprungen. Statt dessen wird eine E-Mail geöffnet, die die ausgewählte(n) Datei(en) als Anhang enthält. URLs im Fenster "Projekt" können als Anhang oder Verknüpfung (Link) versendet werden (*siehe Abbildung unten*). Wählen Sie, wie die URL versendet werden soll und klicken Sie auf **OK**.



28.1.8 Drucken

Schaltfläche und Tastenkürzel

Schaltfläche:	
Tastenkürzel:	**Strg+P**

Beschreibung

Der Befehl **Drucken** öffnet das Dialogfeld "Drucken". Hier können Sie Optionen zum Drucken des gerade aktiven Dokuments festlegen.

Wenn Sie in der Grid-Ansicht auf **Drucken** klicken, wird ein Dialogfeld mit Druckoptionen geöffnet (*siehe Abbildung unten*), in dem Sie Druckoptionen für das XML-Dokument einstellen können. Durch Klicken auf die Schaltfläche "Drucken" wird das Dialogfeld für die Druckeroptionen geöffnet.



Im Folgenden sind die Optionen für das Drucken in der Grid-Ansicht beschrieben:

- Im Bereich **Typ** haben Sie die Möglichkeit festzulegen, was in der Druckausgabe enthalten sein soll.
- Unter **Drucken** können Sie auswählen, ob die gesamte Datei oder ein ausgewählter Bereich gedruckt werden soll.
- Im Bereich **Erweitern** geben Sie an, ob das Dokument so gedruckt werden soll, wie es ist, oder ob Sie alle Child-Elemente ausgeklappt sehen möchten.
- Im Bereich **Inhalt** können Sie entscheiden, ob Sie den Inhalt aller Nodes oder nur die Node-Namen drucken möchten.
- Im Bereich **Wenn Inhalt breiter als Seite** können Sie entscheiden, was geschehen soll, wenn das Dokument größer als eine Papierseite ist. Die Option *aufteilen* druckt das gesamte Dokument in normaler Größe, und der Inhalt wird auf mehrere Seiten aufgeteilt (sowohl senkrecht als auch waagerecht). Sie können die Seiten dann zu einem Poster zusammenfügen. Die Option *Erste Seite* ermöglicht es Ihnen, nur die erste linke Seite des Druckbereichs zu drucken. Der restliche Bereich, der über den Seitenrand ragt, wird nicht gedruckt. Diese Option eignet sich v.a. dann, wenn sich die meisten wichtigen Informationen in der Grid-Ansicht des Dokuments auf der linken Seite befinden. Die Option *Horizontal verkleinern* verkleinert die Druckausgabe bis sie horizontal auf der Papierseite Platz hat. Die Druckausgabe kann mehrere Seiten lang sein. Die Option *Beide verkleinern* reduziert die Größe der Druckausgabe bis sie ganz auf einer Papierseite Platz hat.
- Mit der Schaltfläche **Drucken** wird das Dokument nach Ihren Vorgaben ausgedruckt.
- Mit der Schaltfläche **Vorschau** erhalten Sie eine Vorschau auf die Druckausgabe, bevor diese auf Papier gedruckt wird.
- Mit der Schaltfläche **Drucker einrichten** wird das Dialogfeld "Druck einrichten" geöffnet, wo Sie Papierformat, Ausrichtung, und weitere Druckeinstellungen für diesen Ausdruck vornehmen können. Siehe auch Befehl "Drucker einrichten[1295]".

Bitte beachten Sie: Sie können die Spaltenbreite in der Grid-Ansicht anpassen, um die Druckausgabe zu optimieren.

© 2016-2022 Altova GmbH

Programmlogo

Wenn Sie eine Lizenz erworben haben, können Sie das Programmlogo, die Copyright-Informationen und die Registrierungsinformationen deaktivieren, wenn Sie ein Dokument von XMLSpy aus drucken. Diese Option steht im Dialogfeld "Optionen" im Abschnitt "Ansicht" 1617 zur Verfügung.

WSDL- und XBRL-Design

Grafische Ansichten des WSDL- und des XBRL-Dokuments, wie sie am Bildschirm angezeigt werden, können auch mit dem Befehl **Drucken** gedruckt werden.

28.1.9 Druckvorschau, Druckereinrichtung

Druckvorschau

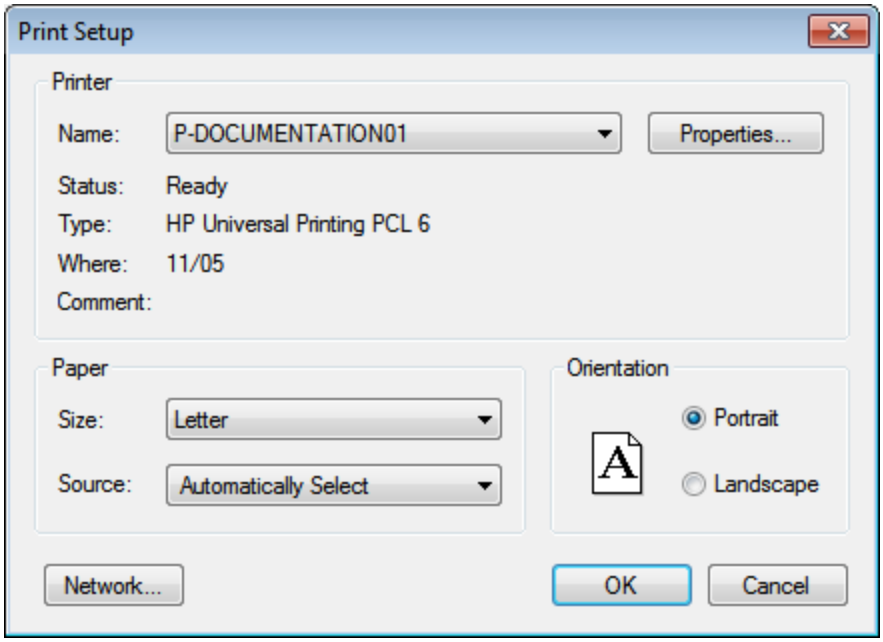
Mit dem Befehl **Druckvorschau** öffnen Sie in der Text-Ansicht, der Authentic-Ansicht und der Browser-Ansicht eine Druckvorschau des aktiven Dokuments. In der Grid-Ansicht, der Schema-Ansicht, der WSDL-Ansicht und der XBRL-Ansicht wird das Dialogfeld "Drucken" geöffnet, in dem Sie die Druckoptionen auswählen und anschließend auf die Schaltfläche "Druckvorschau" klicken können, um die Druckvorschau anzuzeigen.

Im Modus "Druckvorschau" finden Sie in der Durckvorschau-Symbolleiste links oben im Vorschaufenster Druck- und Druckvorschauoptionen. Die Vorschau kann mit Hilfe der Schaltflächen **Vergrößern** und **Verkleinern** vergrößert bzw. verkleinert werden. Wenn eine ganze Seitenlänge in das Vorschaufenster passt, können Sie mit Hilfe der Schaltflächen **Eine Seite** und **Zwei Seiten** eine Ansicht von einer bzw. zwei Seiten anzeigen. Mit Hilfe der Schaltflächen **Nächste Seite** und **Vorherige Seite** können Sie zwischen den Seiten navigieren. Die Symbolleiste enthält auch Schaltflächen, um alle Seiten zu drucken und das Vorschaufenster wieder zu schließen.

Anmerkung: Um die Anzeige von Hintergrundfarben und Bildern in der Druckvorschau zu aktivieren, gehen Sie folgendermaßen vor: (i) Klicken Sie in Internet Explorer im Menü **Extras** auf **Internetoptionen** und anschließend auf das Register "Erweitert"; (ii) aktivieren Sie im Kästchen "Einstellungen" unter "Drucken" das Kontrollkästchen *Hintergrundfarben und -bilder drucken* und (iii) klicken Sie anschließend auf **OK**.

Druckereinrichtung

Mit dem Befehl **Druckereinrichtung...** gelangen Sie in das druckerspezifische Dialogfeld, wo Sie Papierformat, Ausrichtung und weitere Einstellungen tätigen können, die für alle zukünftigen Druckaufträge gültig sind.



28.1.10 Zuletzt verwendete Dateien, Beenden

Zuletzt verwendete Dateien

Im unteren Bereich des Menüs **Datei** sehen Sie eine Liste der neun zuletzt verwendeten Dateien, wobei die zuletzt verwendete Datei zuoberst angezeigt wird. Durch Anklicken einer Datei können Sie diese öffnen. Über die Tastatur können Sie mit der Funktionstaste **ALT+D** das Menü **Datei** aufrufen, und dann die der Datei entsprechende Nummer eintippen.

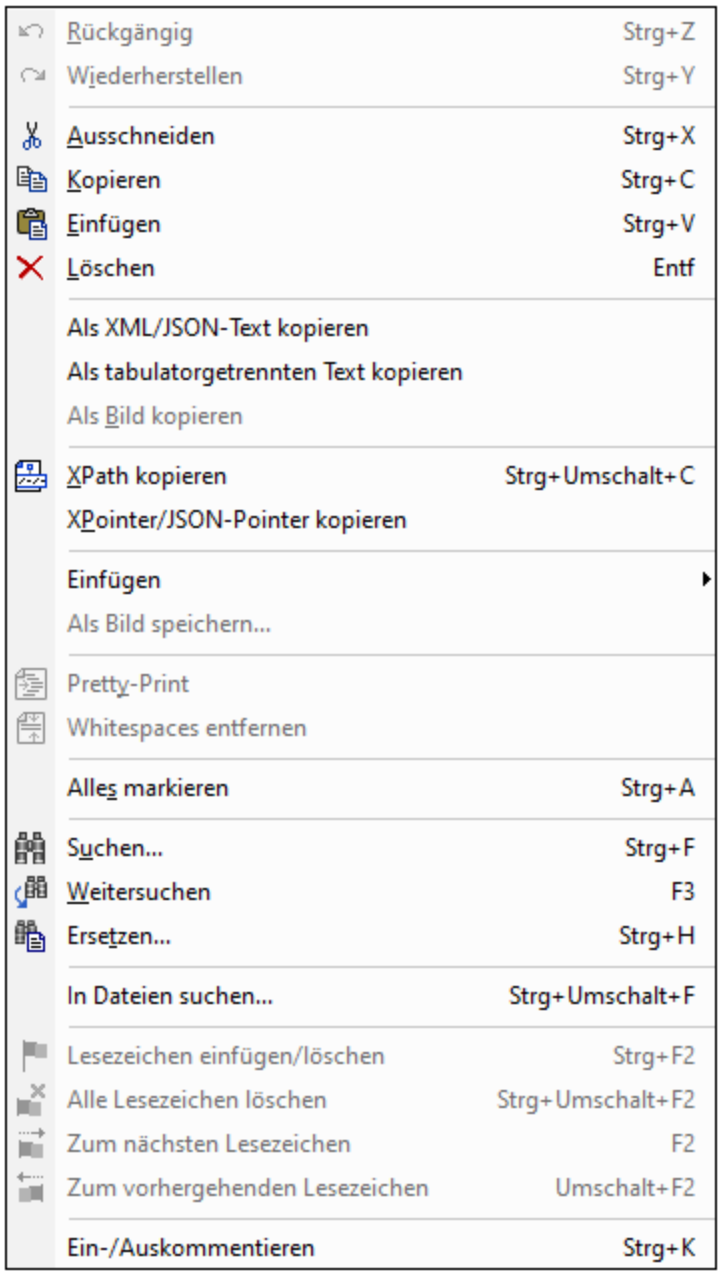
Beenden

Mit dem Befehl **Beenden** beenden Sie XMLSpy. Wenn noch nicht gespeicherte Dokumente offen sind, werden Sie gefragt, ob Sie die Änderungen speichern möchten. XMLSpy speichert weiters Programmeinstellungen, sowie Informationen über die zuletzt verwendeten Dateien.

© 2016-2022 Altova GmbH

28.2 Menü "Bearbeiten"

In diesem Menü befinden sich Befehle, die beim Editieren mit XMLSpy verwendet werden. Darunter befinden sich die Standardbefehle Rückgängig[1298], Wiederherstellen[1298], Ausschneiden[1298], Kopieren[1298], Löschen[1298], Alles markieren[1306], Suchen[1307], Weitersuchen[1307] und Ersetzen[1310].



Außerdem bietet XMLSpy Sonderbefehle für folgende Funktionen:

- Auswahl in Zwischenablage als XML-Text kopieren[1299],
- Als tabulatorgetrennten Text kopieren[1300], oder

- XPath Selector für das ausgewählte Objekt in die Zwischenablage kopieren[1301].
- Einfügen und Entfernen von Lesezeichen und Gehe zu Lesezeichen

28.2.1 Rückgängig, Wiederherstellen

Schaltflächen und Tastenkürzel

Befehl	*Schaltfläche*	*Tastenkürzel*
Rückgängig		**Strg+Z**
Wiederherstellen		**Strg+Y**

Rückgängig

Der Befehl **Rückgängig** unterstützt unbegrenztes Rückgängigmachen aller Aktionen. Jede durchgeführte Aktion kann in Einzelschritten rückgängig gemacht werden. Der Verlauf der durchgeführten Aktionen wird beim Schließen des Dokuments gespeichert, sodass Aktionen auch zu einem späteren Zeitpunkt zurückverfolgt werden können. Mit Hilfe der Befehle **Rückgängig** und **Wiederherstellen** (*siehe Befehl **Wiederherstellen** unten*) können Sie sich Schritt für Schritt vorwärts und rückwärts durch den Verlauf bewegen.

Wiederherstellen

Mit **Wiederherstellen** (in Visual Studio **Wiederholen**) können Sie zuvor rückgängig gemachte Aktionen wiederherstellen. Mlt **Rückgängig** und **Wiederherstellen** können Sie den Verlauf des Dokuments sowohl zurück als auch nach vorne verfolgen.

28.2.2 Ausschneiden, Kopieren, Einfügen, Löschen

Schaltflächen und Tastenkürzel

Befehl	*Schaltfläche*	*Tastenkürzel*
Ausschneiden		**Strg+X** oder **Umschalt+Entf**
Kopieren		**Strg+C**
Einfügen		**Strg+V**
Löschen		**Entf**

Ausschneiden

Mit dem Befehl **Ausschneiden** wird der markierte Text bzw. das markierte Objekt in die Zwischenablage kopiert und aus der aktuellen Position im Dokument gelöscht.

© 2016-2022 Altova GmbH

Kopieren

Mit dem Befehl **Kopieren** kopieren Sie markierten Text oder Objekte in die Zwischenablage. So können Sie Daten in XMLSpy duplizieren, oder in eine andere Applikation kopieren.

Bitte beachten Sie: Es gibt zwei verschiedene Methoden für das Kopieren aus der Grid-Ansicht: **Als XML-Text kopieren**[1299] und **Als strukturierten Text kopieren**[1300]. Mit dem ersten Befehl wird die Auswahl als XML-Text kopiert. Mit dem zweiten wird die Auswahl als Tabelle kopiert. Auf dem Register "Bearbeiten" unter Extras | Optionen[1609] können Sie festlegen, welche dieser beiden Operationen bei Auswahl des Befehls **Kopieren** durchgeführt werden soll.

Einfügen

Mit dem Befehl **Einfügen** fügen Sie den Inhalt der Zwischenablage an der aktuellen Cursorposition ein.

Löschen

Mit dem Befehl **Löschen** wird der markierte Text/das markierte Objekt gelöscht, ohne dass es in die Zwischenablage kopiert wird.

28.2.3 Als XML/JSON-Text kopieren

Mit dem Befehl **Als XML/JSON-Text kopieren** werden XML- oder JSON-Daten aus der Grid-Ansicht als XML- (*siehe Codefragment unten*) oder JSON-Text kopiert. Der Text wird mit seinem Markup-Code in die Zwischenablage kopiert und kann an anderen Stellen in einem Dokument eingefügt werden. Beachten Sie, dass dieser Befehl nur in der Grid-Ansicht zur Verfügung steht.

```
Zeile
   <para align="left">
      <bold>Check the FAQ</bold>
   </para>
   <para>
      <link mode="internal">
         <link_section>support</link_section>
         <link_subsection>faq2020</link_subsection>
         <link_text>XMLSPY 2020 FAQ</link_text>
      </link>
      <link mode="internal">
         <link_section>support</link_section>
         <link_subsection>faq2021</link_subsection>
         <link_text>XMLSPY 2021 FAQ</link_text>
      </link>
   </para>
</row>
```

Die Formatierung des Texts entspricht den gerade aktiven Pretty-Print-Einstellungen, die im Dialogfeld "Optionen"[1611] (**Extras | Optionen**) definiert sind. Dasselbe Ergebnis erzielen Sie, wenn Sie in die Textansicht wechseln und ein XML-Textfragment mit **Strg+C** (**Bearbeiten | Kopieren**) kopieren.

28.2.4 Als tabulatorgetrennten Text kopieren

Der Befehl **Als tabulatorgetrennten Text kopieren** ist nur dann aktiviert, wenn in der Tabellenansicht der XML-Grid-Ansicht[312] und der JSON-Grid-Ansicht[704] ein Bereich von Zellen ausgewählt ist. Sie können damit tabellenartige Daten in Tabellenform kopieren, und zwar auch in Tabellenkalkulationsprogramme.

In den Abbildungen unten sehen Sie, wie zwei Zeilen in der Tabellenansicht als tabulatorgetrennter Text kopiert werden.

expense-item (4)		type	expto	Date	expense
	1	Lodging	Sales	2003-01-01	122.11
	2	Lodging	Development	2003-01-02	122.12

In den nächsten beiden Abbildungen wird gezeigt, wie die Daten als tabulatorgetrennter Text in ein Microsoft Excel- bzw. ein Notepad-Dokument eingefügt werden.

	A	B	C	D	E
1	Lodging	Sales	1/1/2003	122.11	
2	Lodging	Development	1/2/2003	122.12	
3					

Beachten Sie, dass Excel die einzelnen Zellen automatisch auf Basis der lexikalischen Form des Texts formatiert (*Abbildung oben*), während Notepad den gesamten Zellentext als Strings einfügt (*Abbildung unten*).

*Untitled - Notepad

File Edit Format View Help

```
Lodging Sales          2003-01-01      122.11
Lodging Development    2003-01-02      122.12
```

Nähere Informationen dazu finden Sie in den Abschnitten Tabellenansicht (XML)[312] und Tabellenansicht (JSON)[704].

28.2.5 Als Bild kopieren

Der Befehl **Als Bild kopieren** ist nur dann aktiviert, wenn die ausgewählte Zelle in der Tabellenansicht der XML-Grid-Ansicht[312] und der JSON-Grid-Ansicht[704] ein Bild enthält. Der Befehl kopiert den als Base64-kodierten String des ausgewählten Bilds. Wenn der String in ein Dokument eingefügt wird, in dem der Base-64-kodierte String als Bild dargestellt werden kann (z.B. in eine andere Tabellenzelle in der Tabellenansicht), so wird er als Bild dargestellt. Anderfalls wird er als String eingefügt.

Nähere Informationen dazu finden Sie in den Abschnitten Kontextmenü in der XML-Grid-Ansicht[326] und Kontextmenü in der JSON-Grid-Ansicht[718].

© 2016-2022 Altova GmbH

28.2.6 XPath kopieren

Der Befehl **XPath kopieren** steht in der Text- und der Grid-Ansicht zur Verfügung und erstellt einen XPath-Ausdruck, der den/die aktuell ausgewählten Node(s) im Dokument aussucht und den Ausdruck in die Zwischenablage kopiert, sodass Sie ihn dann in ein Dokument (z.B. ein XSLT-Dokument) einfügen können. Alle Ausdrücke beginnen bei der Dokument-Root. Wenn z.B. ein Element des dritten `Person`-Elements des zweiten `Company`-Elements ausgewählt wird, das den Namen `LastName` hat, würde der kopierte XPath-Ausdruck folgendermaßen lauten: `/Companies/Company[2]/Person[3]/LastName`

Anmerkung: In der Grid-Ansicht kann der Befehl **XPath kopieren** auch über das Kontextmenü aufgerufen werden.

28.2.7 XPointer/JSON-Pointer kopieren

Der Befehl **XPointer/JSON-Pointer kopieren** steht in der Grid-Ansicht von XML- und JSON-Dokumenten und in der Textansicht von XML-Dokumenten zur Verfügung. Der Befehl kopiert einen XPointer/JSON-Pointer zum Aufrufen des ausgewählten Nodes in die Zwischenablage.

- *XML-Dokumente:* Der Befehl erstellt einen **`element()`** Schema XPointer auf den/die aktuell ausgewählten Node(s) des XML-Dokuments und kopiert den XPointer in die Zwischenablage. So wird etwa mit dem XPointer **`element(/1/3)`** das dritte Child des Dokument-Elements (oder Root-Elements) ausgewählt.
- *JSON-Dokumente:* Der Befehl erstellt einen JSON-Pointer auf den/die aktuell ausgewählten Node(s) des JSON-Dokuments und kopiert den JSON-Pointer in die Zwischenablage. So wird z.B. mit dem JSON-Pointer **`/Artists/1/Albums/2/Tracks/3/Title`** ein JSON-Node folgendermaßen ausgewählt: Suche nach dem ersten Objekt des **`Artists`** Array auf der obersten Ebene; suche in diesem das zweite Objekt des **`Albums`** Array; suche in diesem das dritte Objekt des **`Tracks`** Array; wähle in diesem das Objekt **`Title`** aus.

Beachten Sie die folgenden Punkte:

- XML-Attribute können mit Hilfe des **`element()`** Schemas nicht dargestellt werden. Wenn ein Attribut ausgewählt wird, wird der XPointer des Parent-Elements des Attributs generiert.
- Wenn mehrere XML-Elemente ausgewählt sind, wird der XPointer des ersten dieser Elemente generiert.
- Wenn ein JSON-Wert ausgewählt ist, wird der JSON-Pointer des Schlüssels des Werts generiert.
- Wenn mehrere JSON-Elemente ausgewählt sind, wird der JSON-Pointer des ersten dieser Elemente generiert.

Anmerkung: Der Befehl **XPointer/JSON-Pointer kopieren** auch über das Kontextmenü aufgerufen werden.

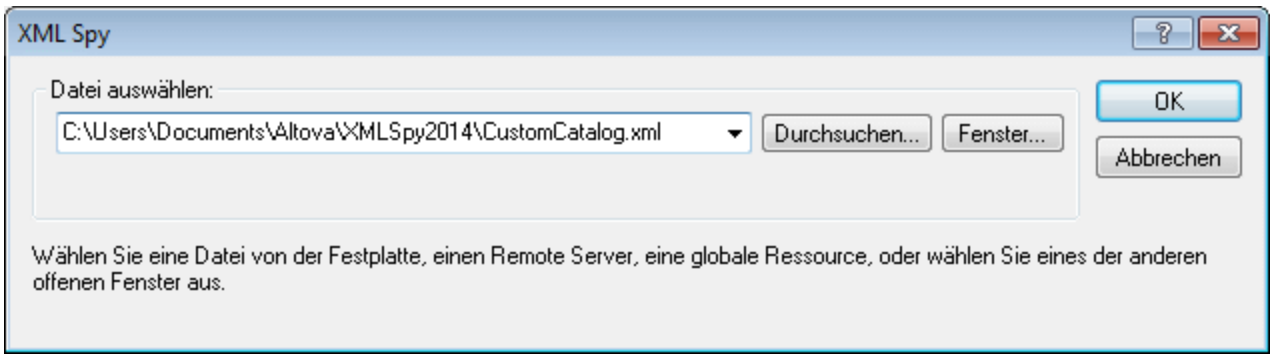
28.2.8 Einfügen

Wenn Sie den Befehl Einfügen auswählen oder den Mauszeiger darüber positionieren, wird ein Untermenü mit drei Befehlen ausgeklappt, die weiter unten beschrieben werden:

- Dateipfad einfügen 1302
- XInclude einfügen 1302
- Externe kodierte Datei einfügen 1305

Dateipfad einfügen

Der Befehl **Dateipfad einfügen** ist in der Text- und der Grid-Ansicht von Dokumenten jedes Dateityps aktiv. Sie können damit den Pfad zu einer Datei am Cursorpunkt einfügen. Wenn Sie auf den Befehl klicken, erscheint ein Dialogfeld (*Abbildung unten*), in dem Sie die gewünschte Datei auswählen können.

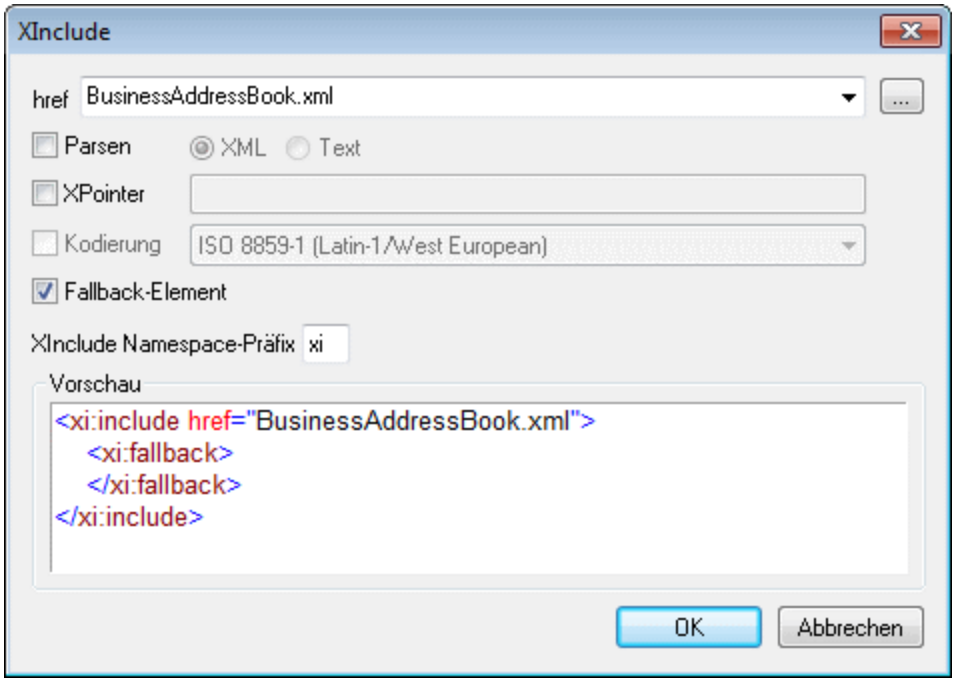


Die gewünschte Datei kann auf eine der folgenden Arten ausgewählt werden: (i) durch Suchen nach der Datei, der URL oder der globalen Ressource (mit Hilfe der Schaltfläche **Durchsuchen** 1280*)*; (ii) durch Auswählen des Fensters, in dem die Datei geöffnet ist (Schaltfläche **Fenster**). Klicken Sie anschließend auf OK. Der Pfad zur ausgewählten Datei wird im aktiven Dokument an der Cursorposition eingefügt.

XInclude einfügen

Der Befehl **XInclude** steht in der Text- und der Grid-Ansicht zur Verfügung und fügt in der Textansicht ein neues XInclude-Element an der Cursorposition ein bzw. fügt er sowohl in der Text- als auch in der Grid-Ansicht vor dem ausgewählten Objekt ein neues XInclude-Element ein. Wenn in der Grid-Ansicht gerade ein Attribut ausgewählt ist, wird das XInclude-Element nach dem Attribut und vor dem ersten Child-Element des Parent-Elements des Attributs eingefügt. Bei Auswahl dieses Befehl wird das Dialogfeld "XInclude" aufgerufen (*Abbildung unten*).

© 2016-2022 Altova GmbH



Die zu inkludierende XML-Datei wird in das Textfeld `href` eingegeben (alternativ dazu können Sie die Datei auch über die Durchsuchen-Schaltfläche (...) rechts vom Textfeld eingeben). Der Dateiname wird als Wert des `href`-Attributs eingegeben. Auch die Attribute `parse`, `xpointer` und `encoding` des Elements XInclude (`xi:include`) und das Child-Element `fallback` von `xi:include` können über das Dialogfeld eingegeben werden. Aktivieren Sie dazu zuerst das entsprechende Kontrollkästchen und wählen Sie anschließend die benötigten Werte aus, bzw. geben Sie sie ein. Im Fall des Elements `fallback` wird durch das Aktivieren des Kontrollkästchens nur das leere Element eingefügt. Der Inhalt des Elements `fallback` muss später in einer der Bearbeitungsansichten hinzugefügt werden.

Das Attribut `parse` bestimmt ob das inkludierte Element als XML oder als Text geparst wird. (XML ist die Standardeinstellung und muss daher nicht extra angegeben werden.) Das Attribut `xpointer` identifiziert ein bestimmtes Fragment des Dokuments, das mit dem Attribute `href` lokalisiert wird; dieses Element wird inkludiert. Das Attribute `encoding` definiert die Kodierung des inkludierten Dokuments, damit dieses Dokument (oder der Teil davon, der inkludiert wird) von XMLSpy in die Kodierung des inkludierenden Dokuments transkodiert werden kann. Der Inhalt des Child-Elements `fallback` ersetzt das Element `xi:include` wenn das zu inkludierende Dokument nicht auffindbar ist.

Hier sehen Sie ein Beispiel für ein XML-Dokument, in dem zum Inkludieren zweier XML-Dokumente XInclude verwendet wird:

```
<?xml version="1.0" encoding="UTF-16"?>
<AddressBook xsi:schemaLocation="http://www.altova.com/sv/myaddresses AddressBook.xsd"
       xmlns="http://www.altova.com/stylevision/tutorials/myaddresses"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:xi="http://www.w3.org/2001/XInclude">
       <xi:include href="BusinessAddressBook.xml"/>
       <xi:include href="PersonalAddressBook.xml"/>
</AddressBook>
```

Beim Parsen dieses Dokuments werden die beiden XInclude-Elemente durch die in den entsprechenden `href`

Attributen definierten Dateien ersetzt.

xml:base
Wenn der XML-Validator von XMLSpy ein XML-Dokument liest und im XInclude-Namespace auf das Element `include` trifft (im Folgenden `xi:include` genannt), ersetzt er dieses Element (`xi:include`) durch das im `href`-Attribut des `xi:include`-Elements definierte XML-Dokument. Das Dokument-Element (Root-Element) des inkludierten XML-Dokuments (bzw. das durch einen XPointer gekennzeichnete Element) wird mit einem Attribut von `xml:base` inkludiert, um die Basis-URIs des inkludierten Elements beizubehalten. Wenn das XML-Ergebnisdokument (das das/die inkludierte(n) XML-Dokument(e) oder Baumstruktur-Fragment(e) enthält) dem Schema entsprechend gültig sein muss, so muss das Dokument-Element des inkludierten Dokuments (oder das Element auf oberster Ebene des Baumstrukturfragments) mit einem Content Model erstellt werden, das ein Attribut `xml:base` zulässt. Wenn das `xml:base` Attribut laut Schema in diesem Element nicht zulässig ist, ist das Ergebnisdokument ungültig. Eine Anleitung zum Definieren eines `xml:base` Attributs im Content Model eines Elements in der Schema-Ansicht von XMLSpy finden Sie in dieser Dokumentation im Abschnitt zu Attributen mit dem Präfix xml: 249 des Abschnitts Schema-Ansicht.

XPointers

XMLSpy unterstützt XPointers in XInclude. Die relevanten W3C Recommendations dazu sind die XPointer Framework und die XPointer element() Schema Recommendation. Durch die Verwendung eines XPointers in einem XInclude-Element kann ein bestimmter Teil des XML-Dokuments anstatt des gesamten XML-Dokuments inkludiert werden. XPointers werden folgendermaßen in einem XInclude-Element verwendet:

```
    <xi:include href="PersonalAddressBook.xml" xpointer="element(usa)"/>
<xi:include href="BusinessAddressBook.xml" xpointer="element(/1/1)"/>
<xi:include href="BobsAddressBook.xml" xpointer="element(usa/3/1)"/>
<xi:include href="PatsAddressBook.xml" xpointer="element(usa)element(/1/1)"/>
```

Im element() Schema von XPointer kann ein NCName oder eine von Ganzzahlen geleitete Child-Sequenz verwendet werden.

- Im ersten oben angeführten `xi:include` Element verwendet das `xpointer` Attribut das Elementschema mit dem NCName `usa`. Laut XPointer Framework identifiziert dieser NCName das Element, das die ID `usa` hat.
- Im zweiten oben angeführten `xi:include` Element identifiziert das `xpointer` Attribut mit dem Wert `element(/1/1)` im ersten Schritt das erste Child-Element der Dokument-Root (wobei es sich um das Dokument (oder Root)-Element handelt, wenn das Dokument wohlgeformt ist. Im zweiten Schritt wird das erste Child-Element des im vorherigen Schritt gesuchten Elements gesucht; in unserem Beispiel wäre das das erste Child-Element des Dokument-Elements.
- Das `xpointer` Attribut des dritten oben angeführten `xi:include` Elements verwendet eine Kombination von NCName und Child-Sequenz. Dieses XPointer-Element sucht das erste Child-Element des dritten Child-Elements des Elements mit der ID `usa`.
- Wenn Sie nicht sicher sind, ob Ihr erstes XPointer-Element funktioniert, können Sie zur Sicherheit ein zweites verwenden, wie im vierten oben angeführten `xi:include` Element aufgelistet: `xpointer="element(usa)element(/1/1)"`. Wenn hier kein Element mit der ID `usa` gefunden wird, definiert der Backup-XPointer, dass das erste Child-Element des Dokument-Elements ausgewählt werden soll. Auch zusätzliche Backups sind zulässig. Einzelne XPointer-Elemente müssen nicht getrennt werden, können aber auch durch ein Leerzeichen getrennt werden wie z.B. in `xpointer="element(usa)element(addresses/1) element(/1/1)"`.

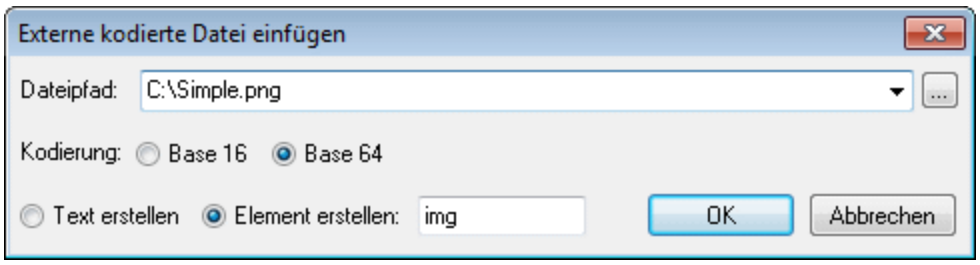
Anmerkung: Der Namespace Binding-Kontext wird im element() Schema nicht verwendet, das das element() Schema qualified Names nicht unterstützt.

 © 2016-2022 Altova GmbH

Externe kodierte Datei einfügen

Der Befehl Externe kodierte Datei einfügen steht in der Text- und der Grid-Ansicht zur Verfügung. Damit kann eine externe Datei an einer beliebigen Position im XML-Dokument als kodierter Base 16 oder Base 64 Text eingefügt werden. Mit Hilfe dieser Funktion können auch externe Dateien in das XML-Dokument eingebettet werden.

Durch Klicken auf den Befehl **Einfügen | Externe kodierte Datei einfügen** wird das Dialogfeld "Externe kodierte Datei einfügen" (*Abbildung unten*) aufgerufen.



Sie können das Dateisystem nach einer externen Datei, die kodiert und eingebettet werden soll, durchsuchen oder ihren Namen eingeben. Es muss entweder eine Base 16 oder eine Base 64 Kodierung angegeben werden. Wenn der kodierte Text von einem Element umschlossen werden soll, aktivieren Sie das Kontrollkästchen Element erstellen und geben Sie den Namen des gewünschten Elements in das Textfeld Element erstellen ein. Wenn das Kontrollkästchen Element erstellen nicht aktiviert ist, wird der kodierte Text direkt an der Position des Cursors eingefügt.

Nachdem Sie auf **OK** geklickt haben wird der kodierte Text der ausgewählten Datei mit einem umschließenden Element - falls dies angegeben wurde - an der Position des Cursors eingefügt.

```
<img ext="png" encoding="xs:base64Binary">
  iVBORw0KGgoAAAANSUhEUgAAABAAAAAQAQMAAAAlPW0iAAAABlBMVEUAAAD/
  //+l2Z/dAAAAM0lEQVR4nGP4/5/h/1+G/58ZDrAz3D/McH8yw83NDDeNGe4U
  g9C9zwz3gVLMDA/A6P9/AFGGFyjOXZtQAAAAAElFTkSuQmCC
</img>
```

Die obige Auflistung zeigt den kodierten Text einer PNG-Bilddatei. Um den kodierten Text wurde ein `img` Element erstellt.

28.2.9 Als Bild speichern

Dieser Befehl ist aktiv, wenn in der Text- oder Grid-Ansicht ein Base64-kodiertes Bild ausgewählt ist. Er konvertiert den ausgewählten Base64-kodierten String in das entsprechende Bildformat. (Beachten Sie, dass in der Textansicht der Base64-String des Bilds angezeigt wird, dass aber in der Grid-Ansicht das anhand des Strings generierte Bild angezeigt wird). Bei Auswahl des Befehls, wird das Dialogfeld "Speichern unter" angezeigt. Wählen Sie darin den Pfad, unter dem Sie das Bild speichern möchten und geben Sie einen Namen für die Bilddatei ein. Die Erweiterung der Bilddatei (`.png`, `.gif`, `.svg`, usw.) wird automatisch anhand der Base64-Kodierung ermittelt und im Speichern-Dialogfeld angezeigt.

Nähere Informationen dazu finden Sie in den Abschnitten über XML-[288] und JSON[1367]-Dokumente unter den Beschreibungen der Text- und der Grid-Ansicht .

28.2.10 Pretty-Print

Symbol

Schaltfläche:	

Beschreibung

Mit dem Befehl **Pretty-Print** wird der Text in Ihrem XML- oder JSON-Dokument neu formatiert. Die Formatierungseigenschaften werden in den Pretty-Print[1611]-Einstellungen des Dialogfelds "Optionen" (**Extras | Optionen**) definiert. Die Einrückung von Text wird im Dialogfeld Einstellungen für Textansicht[1506] (**Ansicht | Einstellungen für Textansicht**) definiert.

Beachten Sie die folgenden Punkte

- Das XML-Dokument muss wohlgeformt sein, damit dieser Befehl funktioniert.
- Bei der Anwendung von Pretty-Print werden beim Speichern des Dokuments Leerzeichen oder Tabulatorzeichen zum Dokument hinzugefügt.
- Um alle mit dem Befehl Pretty-Print erstellten Whitespaces (neue Zeilen und Einrückungen) zu entfernen, verwenden Sie den Befehl Whitespaces entfernen[1306].

28.2.11 Whitespaces entfernen

Schaltfläche

Schaltfläche:	

Beschreibung

Mit dem Befehl **Whitespaces entfernen** werden alle Whitespaces aus den Dokument gelöscht. Dadurch können Sie die Größe einer Datei verringern. Dieser Befehl eignet sich dazu, um Whitespaces, die mit dem Befehl **Pretty-Print**[1306] generiert wurden, wieder zu entfernen.

28.2.12 Alles markieren

Mit dem Befehl **Alles markieren (Strg+A)** (in Visual Studio **Alle auswählen**) wird der Inhalt des gesamten Dokuments markiert.

© 2016-2022 Altova GmbH

28.2.13 Suchen, Weitersuchen...

Schaltflächen und Tastenkürzel

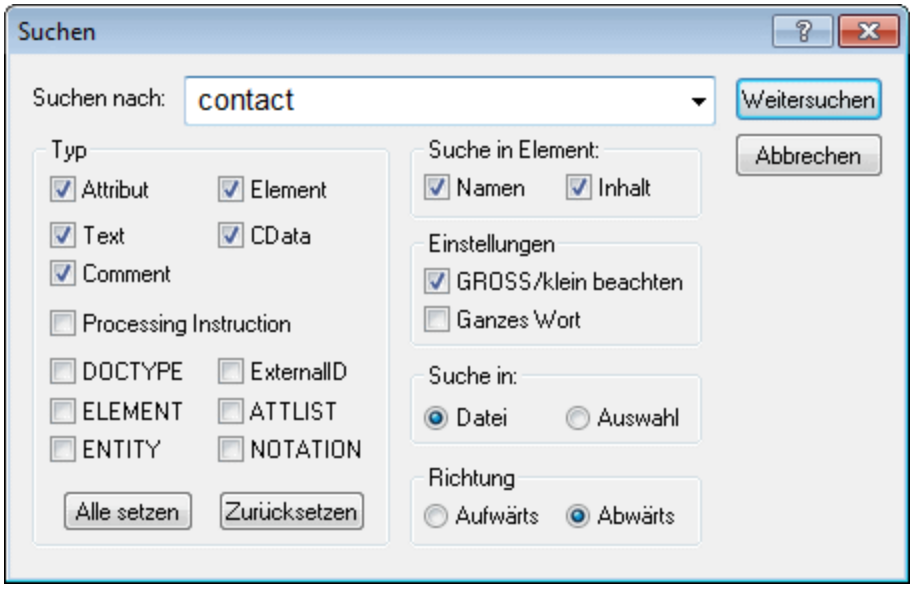
Befehl	*Schaltfläche*	*Tastenkürzel*
Suchen		**Strg+F**
Weitersuchen		**F3**

Suchen

Mit dem Befehl **Suchen** rufen Sie das Dialogfeld "Suchen" auf, in dem Sie den zu suchenden String sowie weitere Suchoptionen spezifizieren können. Abhängig von der aktuellen Ansicht, stehen im Dialogfeld "Suchen" verschiedene Optionen zur Auswahl.Um nach Text zu suchen, geben Sie diesen in das Suchfeld ein oder wählen Sie einen der letzten 10 Einträge aus der Dropdown-Liste aus und legen Sie anschließend die Suchoptionen fest.

Mit den Befehlen **Suchen** und **Weitersuchen** können Sie auch nach Datei- und Ordnernamen suchen, wenn ein Projekt im Fenster "Projekt" ausgewählt ist.

Grid-Ansicht
In der Grid-Ansicht erscheint das folgende Dialogfeld. Wählen Sie die gewünschten Optionen aus oder aktivieren Sie die gewünschten Optionsfelder. Die verfügbaren Optionen sind unten beschrieben.

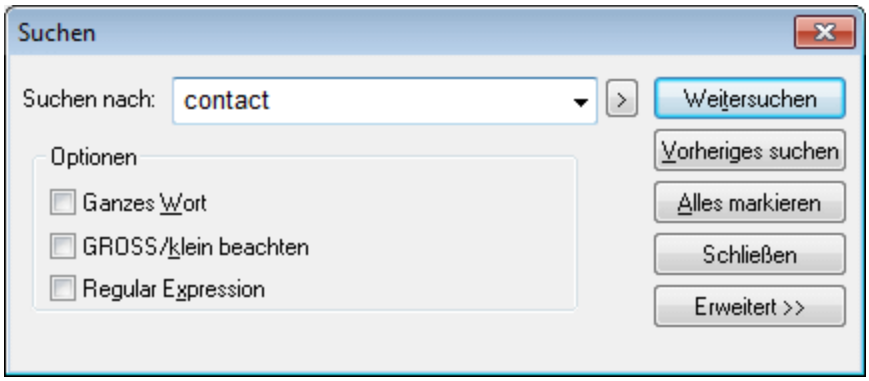


- Im Bereich "*Typ*" wird definiert, welche XML-Dokument-Nodes oder -Komponenten in die Suche einbezogen werden sollen. Hierdurch können Sie bestimmte Node-Typen bei der Suche überspringen und die Suche dadurch beschleunigen. Durch Klicken auf die Schaltfläche **Alle aktiv** werden alle Kontrollkästchen im Bereich Typ aktiviert; die Schaltfläche **Zurücksetzen** deaktiviert alle Kontrollkästchen.

- Im Bereich "*Suche in Element*" können Sie definieren, ob nach Text nur in Node-Namen, dem Inhalt des Node oder in beidem gesucht werden soll.
- Im Bereich "*Einstellungen*" können Sie angeben, ob die Groß- bzw. Kleinschreibung beachtet werden soll, und ob nur nach ganzen Wörtern gesucht werden soll.
- Im Bereich "*Suche in*" wird definiert, in welchem Bereich die Suche erfolgen soll (in der gesamten Datei oder im ausgewählten Text).
- Im Bereich "*Richtung*" definieren Sie, in welche Richtung die Suche erfolgen soll.

Textansicht

Eine Beschreibung der Suchfunktion einer editierbaren Textansicht finden Sie im Abschnitt Textansicht[157]. In der Abbildung unten sehen Sie das Dialogfeld "Suchen" von nicht editierbaren Textansichten (wie z.B. im Fenster XPath/XQuery[124]).



Es stehen die folgenden Optionen zur Verfügung:

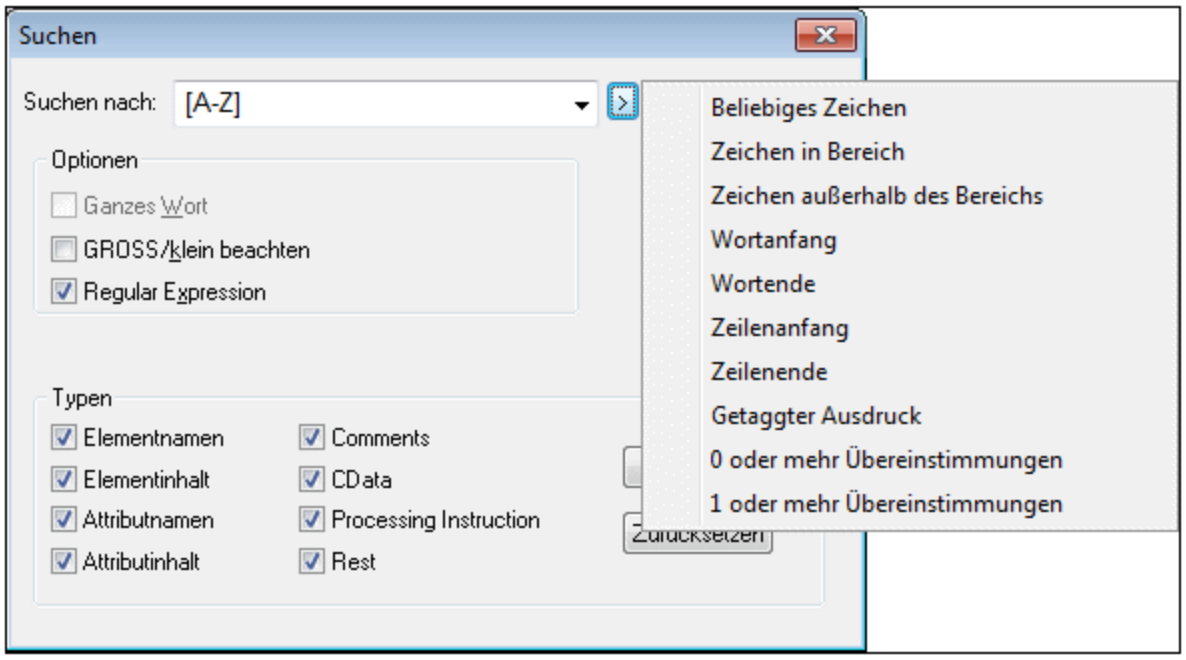
- *Ganzes Wort*: Bei der Suche werden nur die Wörter im Text gefunden, die exakt mit dem gesuchten Text übereinstimmen. Bei Eingabe des Suchstrings `f i t` würde z.B. bei Aktivierung der Option "*Ganzes Wort*" nur das Wort `f i t` gefunden; `f i t in f i t n e s s` wird nicht gefunden.
- *GROSS/klein beachten*: Bei der Suche wird die Groß- und Kleinschreibung beachtet ("`Address`" ist nicht gleich "`address`").
- *Regular expression*: Suche nach Text, der durch die Regular Expression definiert wird, die Sie im Textfeld eingegeben haben. Eine Beschreibung zu Regular Expressions finden Sie unter Regular Expressions[1309].

Bitte beachten Sie:

- Das Suchdialogfeld hat keinen Modus, d.h. es kann geöffnet bleiben, während Sie in der Text-Ansicht arbeiten. Wenn Sie bei geöffnetem Dialogfeld die **Eingabetaste** drücken, wird es geschlossen. Wenn Text vor dem Öffnen des Dialogfelds markiert war, wird der markierte Text automatisch im Textfeld "Suchen nach:" übernommen.
- Wurde das Suchdialogfeld geschlossen, können Sie die aktuelle Suche wiederholen. Drücken Sie die **F3-**Taste um eine Suche nach unten bzw. **UMSCHALT+F3** um eine Suche nach oben durchzuführen.
- Durch Klicken auf die Erweiterungsschaltfläche (>) rechts neben dem "Suchen nach:"-Listenfeld wird ein zweites Fenster geöffnet, über das Sie Regular Expressions[1309] in das Eingabefeld "Suchen nach" eingeben können.

 © 2016-2022 Altova GmbH

Regular Expressions
Sie können Ihre Suchkriterien mit Hilfe von Regular Expressions näher definieren. Für die Erstellung der Regular Expression steht eine Popup-Liste zur Verfügung, die Sie über die Erweiterungsschaltfläche (>) rechts vom Suchbegriff-Eingabefeld aufrufen können.



Wenn Sie auf die Beschreibung des gewünschten Suchausdrucks klicken, wird die entsprechende Expression-Syntax im Eingabefeld übernommen. Im Folgenden sehen Sie eine Liste von Syntax-Zeichen, die für Regular Expressions verwendet werden.

`.`	Steht für jedes beliebige Zeichen. Dies ist ein Platzhalter für ein einzelnes Zeichen.
`(`	Markiert den Beginn einer Region, die gesucht werden soll.
`)`	Markiert das Ende einer zu suchenden Region.
`\n`	`n` steht für `1` bis `9` und bezieht sich auf die erste bis neunte zu ersetzende Suchentsprechung. Lautet der Suchstring beispielsweise `Fred([1-9])XXX` und der Ersetzungsstring `Sam\1YYY`, so wird `Fred2XXX` durch `Sam2YYY` ersetzt.
`\<`	Steht für den Beginn eines Worts.
`\>`	Steht für das Ende eines Worts.
`\x`	Damit können Sie ein Zeichen `x` verwenden, das sonst eine spezielle Bedeutung hätte. So würde z.B. `\[` als `[` und nicht als Beginn einer Zeichenmenge interpretiert werden.
`[...]`	Kennzeichnet eine *Zeichenmenge*, `[abc]` z.B. steht für jedes der Zeichen `a`, `b` oder `c`. Sie können auch Bereiche angeben, z.B. `[a-z]` für alle klein geschriebenen Zeichen.
`[^...]`	Die invertierte Zeichenmenge. `[^A-Za-z]` z.B. steht für jedes Zeichen mit Ausnahme alphabetischer Zeichen.
`^`	Steht für den Zeilenanfang (es sei denn dieses Zeichen wird innerhalb einer Menge verwendet, *siehe oben*).

$	Steht für das Zeilenende. Beispiel: A+$ findet ein oder mehrere A's am Ende der Zeile.
*	Steht für 0 oder öfter. Mit Sa*m werden z.B. Sm, Sam, Saam, Saaam usw. gefunden.
+	Steht für 1 oder öfter. Mit Sa+m werden z.B. Sam, Saam, Saaam usw. gefunden.

Anmerkung: Regular Expressions werden im Feld "Ersetzen[1310]" nicht unterstützt.

Mit dem Befehl **Weitersuchen** wird der letzte Suchbefehl wiederholt, um nach der nächsten Instanz des gesuchten Texts zu suchen.

28.2.14 Ersetzen...

Schaltflächen und Tastenkürzel

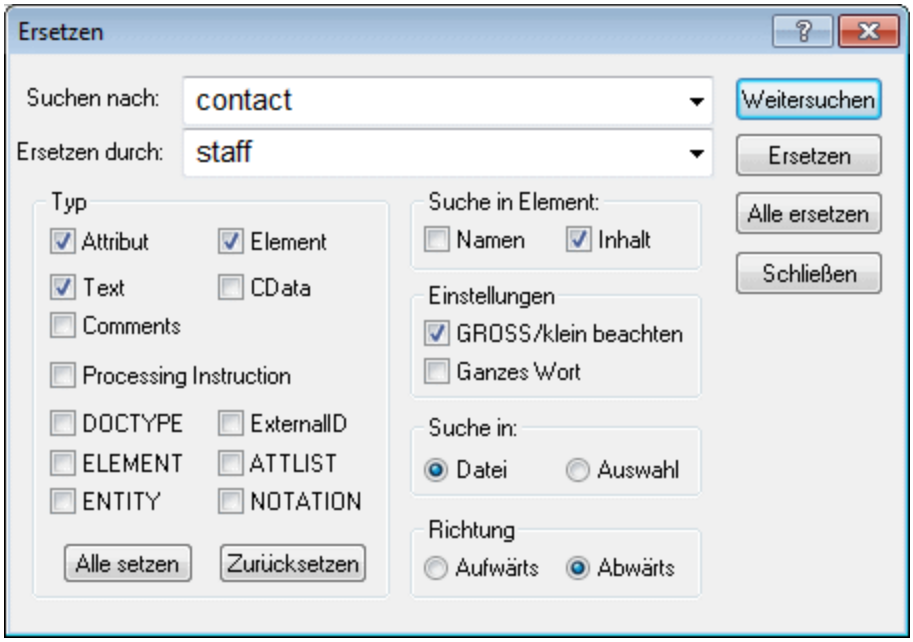
Befehl	*Schaltfläche*	*Tastenkürzel*
Ersetzen		**Strg+H**

Beschreibung

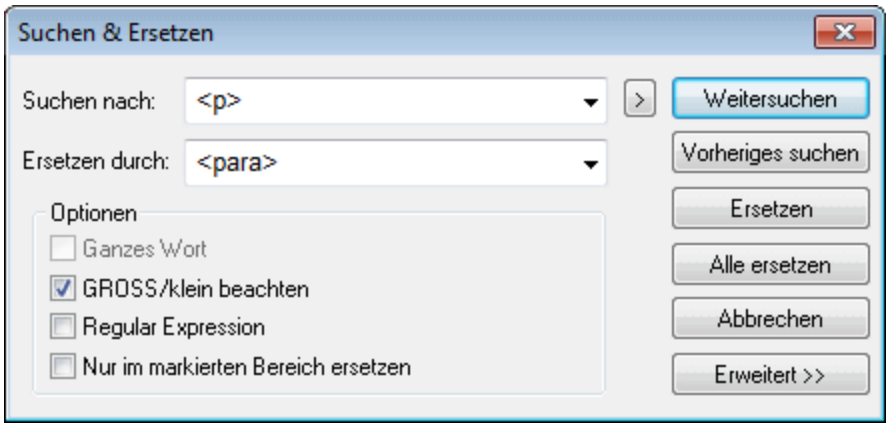
Mit dem Befehl **Ersetzen** (in Visual Studio **Suchen und Ersetzen**) können Sie einen Textstring durch einen anderen ersetzen. Der Befehl verfügt über dieselben Optionen wie der Befehl **Suchen**[1307] ...[1307].Abhängig von der ausgewählten Ansicht, gibt es im Dialogfeld "Ersetzen" verschiedene zur Auswahl stehende Optionen. Sie können diese Funktion Schritt für Schritt verwenden, oder alle Begriffe auf einmal ersetzen, indem Sie auf die Schaltfläche "**Alle ersetzen**" klicken.

Grid-Ansicht
Wenn Sie den Befehl **Ersetzen** in der Grid-Ansicht verwenden, wird das Dialogfeld "Ersetzen"' (*Abbildung unten*) geöffnet. In der unten gezeigten Abbildung sehen Sie die verschiedenen Suchoptionen. Eine Beschreibung dieser Optionen finden Sie unter dem Befehl **Suchen**[1307].

© 2016-2022 Altova GmbH



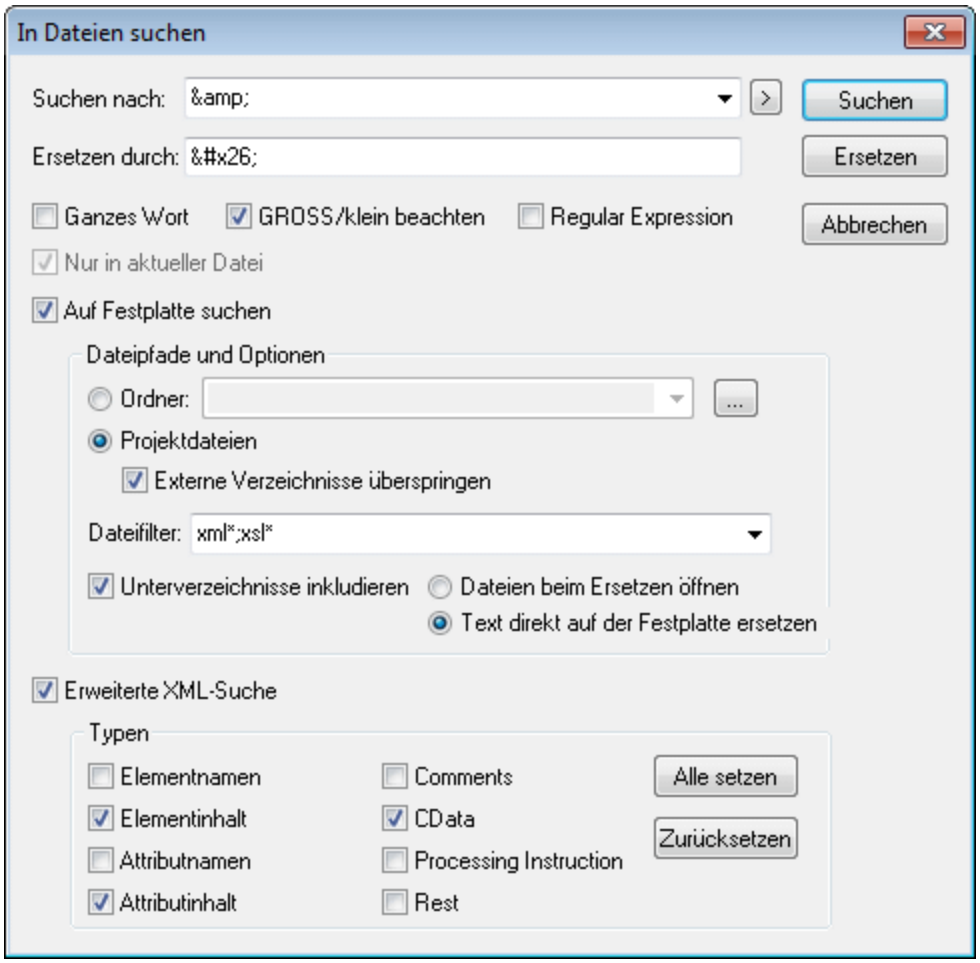
Textansicht
Eine Beschreibung der Suchfunktion einer editierbaren Textansicht finden Sie im Abschnitt Textansicht[157]. Die Suchfunktion der nicht editierbaren Textansicht (wie z.B. im Fenster XPath/XQuery[124]) finden Sie im Folgenden. Wenn Sie in der Text-Ansicht den Befehl "**Ersetzen**" wählen, wird das unten gezeigte Dialogfeld "Suchen & Ersetzen" angezeigt. Die Optionen sind dieselben wie im Dialogfeld **Suchen...**[1307].. Bei Auswahl der Option "*Nur im markierten Bereich ersetzen*" werden die Ersetzungen nur im markierten Dokumentbereich vorgenommen.



Bitte beachten Sie: Bei Auswahl des Befehls "Alle ersetzen" wird jede einzelne Ersetzung als Einzelvorgang aufgezeichnet, sodass Sie diesen Vorgang Schritt für Schritt rückgängig machen können.

28.2.15 In Dateien suchen

Der Befehl **In Dateien suchen** eignet sich ideal dazu, Text schnell in vielen Dateien gleichzeitig zu suchen und zu ersetzen. Wenn Sie den Befehl aufrufen, wird das Dialogfeld "**In Dateien suchen**" (*Abbildung unten*) angezeigt. Der Befehl **In Dateien suchen** unterscheidet sich insofern vom Befehl **Suchen,** als damit alle angegebenen Dateien auf einmal nach dem gewünschten Suchstring durchsucht werden und der Ersetzungsvorgang überall gleichzeitig durchgeführt wird. Anschließend wird im Ausgabefenster "In Dateien suchen" [127] ein Bericht angezeigt. Bei Verwendung des Befehls **Suchen** gibt der Benutzer den Suchstring ein und geht ein einzelnes Dokument (das aktive Dokument) von Instanz zu Instanz durch.



Suchkriterien

Es gibt zwei grundlegende Suchkriterien: (i) Was soll gesucht werden und (ii) wo soll gesucht werden? Eine Anleitung, wie man den zu suchenden Text eingibt (was soll gesucht werden), finden Sie in der Beschreibung zum Befehl **Suchen** [1307]. Wenn es sich bei dem im Textfeld "Suchen nach" eingegebenen Text um eine Regular Expression handelt, muss das Kontrollkästchen "Regular Expression" aktiviert werden. Um eine Eingabehilfe zur Eingabe von Regular Expressions aufzurufen, klicken Sie auf die Schaltfläche Erweiterungsschaltfläche (>) rechts vom Eingabefeld "Suche nach". Die Verwendung von Regular Expressions zur Suche wird unter **Suchen** [1307] erklärt.

© 2016-2022 Altova GmbH

Um festzulegen, welche Node-Typen und Teile eines XML-Dokuments durchsucht werden sollen, aktivieren Sie das Kontrollkästchen "Erweiterte XML-Suche" und aktivieren Sie anschließend die gewünschten Node-Typen.

Sie können festlegen, welche Dateien durchsucht werden, indem Sie entweder die Option *Nur in aktueller Datei* oder die Option *Auf Festplatte suchen* aktivieren. Bei Auswahl der Option *Auf Festplatte suchen* können Sie einen Ordner oder ein Projekt[1316] auswählen, das durchsucht werden soll. Bei Auswahl eines Projektordners können externe Ordner, die zu einem Projekt hinzugefügt wurden, übersprungen werden. Die zu durchsuchenden Dateien können nach Dateierweiterung und einem Sternchen (z.B. `xml*` oder `xsl*`) gefiltert werden. Das Trennzeichen zwischen zwei Dateierweiterungen kann ein Komma oder ein Semikolon (`xml*;xsl*` zum Beispiel) sein. Das Stern-Zeichen kann ebenfalls als Wildcard verwendet werden.

Die Instanzen, die vom Suchstring in allen angegebenen Suchordnern gefunden werden, werden in der "In Dateien suchen"-Ausgabeleiste[127] aufgelistet. Wenn Sie auf einen der aufgelisteten Einträge klicken, wird die entsprechende Datei in der Text-Ansicht geöffnet und die Instanz wird markiert.

Ersetzen

Beachten Sie: Wenn Sie auf die Schaltfläche **Ersetzen** klicken, werden alle Instanzen des Suchstrings durch den Ersetzungsstring ersetzt. Wenn Sie im Dialogfeld "In Dateien suchen" die Option "*Dateien beim Ersetzen öffnen*" aktiviert haben, wird die Datei in der Text-Ansicht geöffnet. Andernfalls geschieht die Ersetzung im Hintergrund. Alle ersetzten Strings werden in der "In Dateien suchen"-Ausgabeleiste[127] aufgelistet. Wenn Sie auf einen der aufgelisteten Einträge klicken, wird die Datei in der Text-Ansicht geöffnet und der Eintrag wird markiert.

Anmerkung: Regular Expressions werden im Feld "Ersetzen durch" nicht unterstützt.

28.2.16 Lesezeichen-Befehle

Schaltflächen und Tastenkürzel

Befehl	*Schaltfläche*	*Tastenkürzel*
Lesezeichen einfügen/löschen		**Strgs+F2**
Alle Lesezeichen löschen		**Strg+Umschalt +F2**
Zum nächsten Lesezeichen		**F2**
Zum vorhergehenden Lesezeichen		**Umschalt+F2**

Lesezeichen einfügen/löschen

Mit dem Befehl **Lesezeichen einfügen/löschen** können Sie ein Lesezeichen an der aktuellen Cursorposition einfügen, bzw. ein vorhandenes Lesezeichen entfernen, wenn sich der Cursor in einer Zeile befindet, in der zuvor ein Lesezeichen eingefügt wurde. Dieser Befehl steht nur in der Textansicht zur Verfügung.

Mit Lesezeichen markierte Zeilen können auf zwei Arten angezeigt werden:

- Wurde die Lesezeichenleiste aktiviert, wird eine blaue Ellipse links vom Text in der Lesezeichenleiste angezeigt.
- Ist die Lesezeichenleiste nicht aktiviert, so wird die gesamte Zeile, in der sich der Cursor befindet markiert.

Mit der **F2**-Taste können Sie der Reihe nach alle Lesezeichen in der Textansicht anzeigen.

Alle Lesezeichen löschen

Mit dm Befehl **Alle Lesezeichen löschen** löschen Sie alle derzeit in der Textansicht definierten Lesezeichen. Dieser Befehl steht nur in der Textansicht zur Verfügung. **Bitte beachten Sie:** Die Auswirkungen des Befehls **Alle Lesezeichen löschen** können mit dem Befehl "Rückgängig" nicht rückgängig gemacht werden.

Zum nächsten Lesezeichen

Mit dem Befehl **Zum nächsten Lesezeichen** setzen Sie den Cursor an den Beginn der nächsten mit einem Lesezeichen versehenen Zeile. Dieser Befehl ist nur in der Textansicht verfügbar.

Zum vorhergehenden Lesezeichen

Mit dem Befehl **Zum vohergehenden Lesezeichen** setzen Sie den Cursor an den Beginn der vorhergehenden mit einem Lesezeichen versehenen Zeile. Dieser Befehl ist nur in der Textansicht verfügbar.

28.2.17 Ein-/Auskommentieren

Der Befehl **Ein-/Auskommentieren** steht in der Textansicht zur Verfügung und dient zum ein- und auskommentieren von XML-Textfragmenten. Text kann in einem XML-Dokument mit Hilfe der Trennzeichen für den Beginn und das Ende eines Kommentars `<!--` bzw. `-->` auskommentiert werden. In XMLSpy können diese Kommentar-Trennzeichen mit dem Befehl **Bearbeiten | Ein-/Auskommentieren** rund um einen ausgewählten Text eingefügt werden.

Um einen Textblock auszukommentieren, wählen Sie den Text aus und wählen Sie anschließend entweder im Menü **Bearbeiten** oder im Kontextmenü den Befehl **Ein-/Auskommentieren** aus. Der auskommentierte Text wird ausgegraut (*siehe Abbildung unten*).

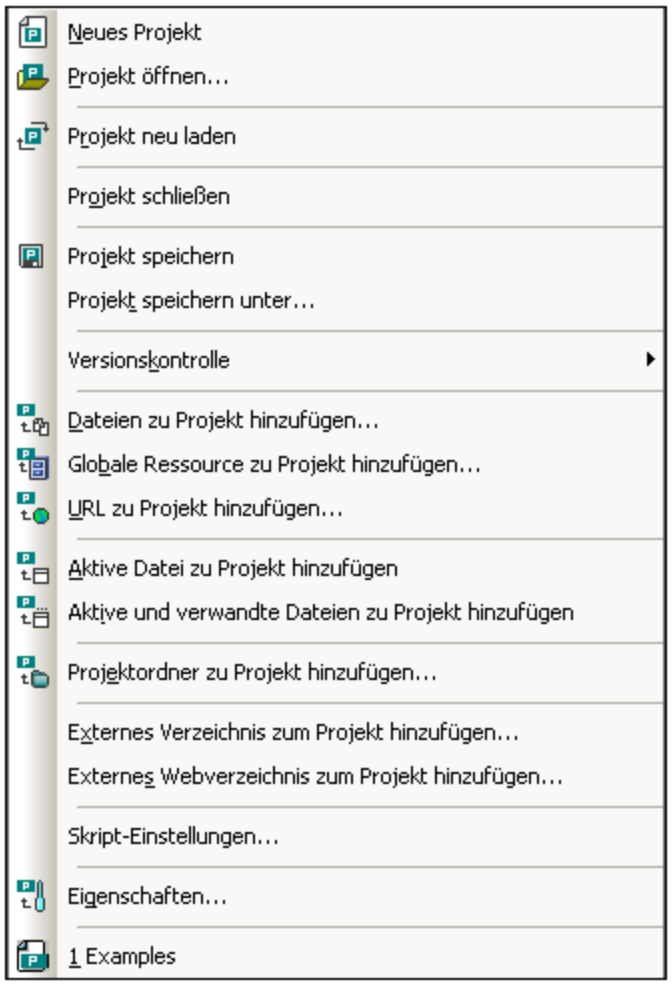
```
<Department>
    <Name>Administration</Name>
    <Person>
    <Person>
    <Person>
    <!--<Person>
        <First
        <Last></Last>
        <PhoneExt></PhoneExt>
        <EMail></EMail>
        <LeaveTotal></LeaveTotal>
        <LeaveUsed></LeaveUsed>
        <LeaveLeft></LeaveLeft>
    </Person>-->
</Department>
```


© 2016-2022 Altova GmbH

Um die Auskommentierung eines Textblocks rückgängig zu machen, setzen Sie den Cursor in den auskommentierten Block und wählen Sie entweder im Menü **Bearbeiten** oder im Kontextmenü den Befehl **Ein-/Auskommentieren** aus. Die Kommentar-Trennzeichen werden entfernt und der Text ist nun nicht mehr ausgegraut.

28.3 Menü "Projekt"

XMLSpy verwendet die bekannte Baumstruktur, um Dateien und URLs in einem XML-Projekt zu gliedern. Dateien[1335] und URLs[1336] mit gleicher Erweiterung können in Ordnern[1337] gesammelt werden, wodurch eine einfache Strukturierung und Batchmanipulation gegeben ist.



Bitte beachten Sie:
Die meisten projektbezogenen Befehle sind auch mittels Rechtsklick auf das gewünschte Objekt über das Kontextmenü verfügbar.

Absolute und relative Pfade

Jedes Projekt ist als Projektdatei gespeichert, die mit der Erweiterung **.spp** versehen ist. Bei diesen Dateien handelt es sich eigentlich um XML-Dokumente, die Sie wie jede reguläre XML-Datei editieren können. In der Projektdatei werden für Dateien/Ordner auf derselben oder einer höheren Ebene absolute Pfade verwendet, während für Dateien/Ordner im aktuellen Ordner oder einem Unterordner relative Pfade definiert werden. Wenn Ihre Verzeichnisstruktur z.B. in etwa so aussieht:

```
|-Folder1
```


© 2016-2022 Altova GmbH

```
|      |
|      |-Folder2
|            |
|            |-Folder3
|                  |
|                  |-Folder4
```

...und Ihre **.spp**-Datei in `Folder3` gespeichert ist, so werden Referenzen auf Dateien in `Folder1` und `Folder2` folgendermaßen angegeben:

```
c:\Folder1\NameOfFile.ext
c:\Folder1\Folder2\NameOfFile.ext
```

Referenzen auf Dateien in `Folder3` und `Folder4` würden in etwa folgendermaßen aussehen:

```
.\NameOfFile.ext
.\Folder4\NameOfFile.ext
```

Wenn Sie sicherstellen möchten, dass alle Pfade relativ angegeben werden, speichern Sie die `.spp`-Dateien im Root-Verzeichnis Ihres Rechners.

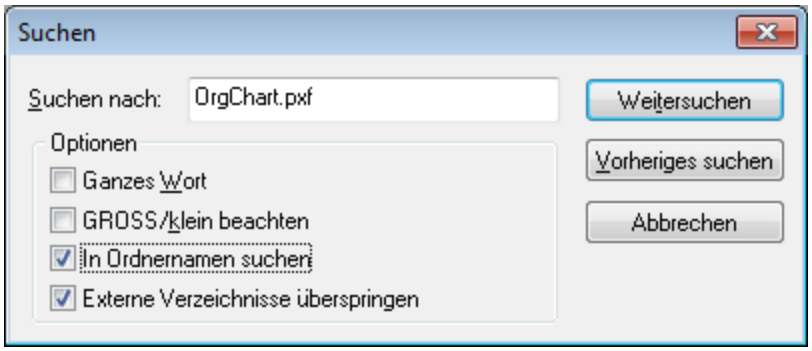
Drag-and-Drop

Im Projekt-Fenster kann ein Ordner in einen anderen Ordner oder an eine andere Stelle im selben Ordner gezogen werden. Eine Datei kann in einen anderen Ordner, nicht aber innerhalb desselben Ordners (in dem die Dateien alphabetisch geordnet sind) verschoben werden. Zusätzlich dazu können Dateien und Ordner aus dem Windows-Datei-Explorer in das Projektfenster gezogen werden.

In Projekt suchen

Sie können Projektdateien und -ordner nach ihrem Namen bzw. einem Teil ihres Namens suchen. Wurden Treffer gefunden, so werden die gefundenen Dateien bzw. Ordner der Reihe nach markiert.

Um eine Suche zu starten, wählen Sie den zu durchsuchenden Projektordner in der Seitenleiste "Projekt" aus und klicken Sie auf den Befehl **Bearbeiten | Suchen** (oder das Tastenkürzel **Strg+F**). Geben Sie nun im Dialogfeld "Suchen" (*Abbildung unten*) den gewünschten Such-String ein und aktivieren bzw. deaktivieren Sie die Suchoptionen nach Bedarf (*Erklärung weiter unten*)



Es stehen die folgenden Suchoptionen zur Verfügung:

- Ganzes Wort: Hierbei ist die Suche stärker eingeschränkt, da der Suchstring mit einem ganzen Wort im Datei- bzw. Ordnernamen übereinstimmen muss. Die Teile vor und nach dem Punkt in den Dateinamen werden (ohne Berücksichtigung des Punkts) jeweils als Wort behandelt.
- Sie können festlegen, dass die Groß- und Kleinschreibung im Suchstring genau mit dem Textstring im Datei- oder Ordnernamen übereinstimmen muss.
- Ordnernamen können in die Suche einbezogen werden. Andernfalls werden nur Dateinamen durchsucht.
- Externe Ordner[1337] können in die Suche einbezogen oder davon ausgeschlossen werden. Bei externen Ordnern handelt es sich um tatsächliche Ordner auf Ihrem Rechner bzw. Netzwerk und nicht um Projektordner, die innerhalb des Projekts und nicht auf dem Rechner angelegt wurden.

Wurden Treffer gefunden, wird der erste Treffer in der Seitenleiste "Projekt" markiert. Sie können anschließend im Dialogfeld **Suchen** mit Hilfe von **Weitersuchen** und **Vorh. suchen** der Reihe nach zu allen gefundenen Suchtreffern springen.

Aktualisieren von Projekten

Wenn eine Änderung an einem externen Ordner vorgenommen wird, wird diese Änderung erst im Projektfenster angezeigt, wenn das Projekt aktualisiert wurde.

Globale Ressourcen im Kontextmenü

Wenn Sie mit der rechten Maustaste im Projektfenster auf einen Ordner klicken, können Sie im daraufhin angezeigten Kontextmenü den Befehl **Globale Ressource hinzufügen** auswählen, um eine globale Ressource[1061] hinzuzufügen. Daraufhin wird das Dialogfeld "Globale Ressource wählen" angezeigt, in dem alle globalen Ressourcen vom Typ Datei und Ordner in der derzeit aktiven XML-Datei für globale Ressourcen aufgelistet sind. Wählen Sie die gewünschte globale Ressource und sie wird zum ausgewählten Projektordner hinzugefügt.

Projekte und Versionskontrolle

Wenn Sie vorhaben, ein XMLSpy-Projekt zu einem Versionskontroll-Repository hinzuzufügen, stellen Sie bitte sicher, dass die Projektdateien in der hierarchischen Dateisystemstruktur in einem Ordner abgelegt werden, von dem aus Dateien nur aus diesem untergeordneten Ordnern hinzugefügt werden können (Nehmen Sie das Root-Verzeichnis als oberste Ebene der Verzeichnisstruktur)

In anderen Worten stellt das Verzeichnis, in dem sich die **Projektdatei** befindet, innerhalb des Versionskontroll-Repository im Grunde das **Root-Verzeichnis** des Projekts dar. Dateien, die dem Projekt aus einem Ordner hinzugefügt werden, der sich oberhalb dieses Root-Verzeichnisses befindet, werden dem XMLSpy-Projekt zwar hinzugefügt, doch ist ihr Speicherort im Repository - falls überhaupt zulässig - manchmal ein unerwarteter.

Nehmen wir als Beispiel die oben gezeigte Verzeichnisstruktur:
Wenn eine Projektdatei in `Folder3` gespeichert wird und einer Versionskontrolle unterliegt:

- so werden Dateien, die zu `Folder1` hinzugefügt werden, unter Umständen nicht von der Versionskontrolle berücksichtigt,
- werden Dateien, die zu `Folder2` hinzugefügt werden, dem Root-Verzeichnis des Repository anstelle des Projektordners hinzugefügt, jedoch von der Versionskontrolle berücksichtigt,
- während die in `Folder3` und `Folder4` gespeicherten Dateien korrekt verarbeitet werden und auch der Versionskontrolle unterliegen.

© 2016-2022 Altova GmbH

28.3.1 Neues Projekt



Mit dem Befehl **Neues Projekt** erstellen Sie ein **neues** Projekt in XMLSpy. Wenn Sie gerade an einem anderen Projekt arbeiten, werden Sie gefragt, ob Sie alle Dokumente schließen wollen, die zu dem aktuellen Projekt gehören. Der Name des Projekts wird beim Speichern des Projekts als `.spp`-Datei zugewiesen.

28.3.2 Projekt öffnen.



Diese Funktion **öffnet** ein bereits vorhandenes Projekt in XMLSpy.
Falls Sie zuvor an einem anderen Projekt gearbeitet haben, wird dieses Projekt zuerst geschlossen.

28.3.3 Projekt neu laden



Mit dieser Funktion wird das aktuelle Projekt von der Festplatte **neu geladen**.
Falls Sie in einer Multiuser-Umgebung arbeiten, kann es oftmals nötig sein, ein Projekt, an dem mehrere Personen gleichzeitig arbeiten, neu zu laden, um deren Änderungen anzusehen.

Bitte beachten Sie: Projektdateien (.spp) sind XML-Dokumente, die Sie wie jede andere XML Datei bearbeiten können.

28.3.4 Projekt schließen

Mit diesem Befehl schließen Sie das **aktuelle** Projekt. Wurde das Projekt geändert, werden Sie vor dem Schließen gefragt, ob die Änderungen gespeichert werden sollen.
Wenn ein Projekt oder eine XML-Datei geändert wurde, wird der Name der Datei im Projektfenster mit einem Stern "*" versehen.

28.3.5 Projekt speichern, Projekt speichern unter



Mit dem Befehl **Projekt speichern** wird das aktuelle Projekt **gespeichert**. Sie können ein Projekt auch speichern, indem Sie das Projektfenster zum aktiven machen und auf das Symbol  klicken.
Mit dem Befehl **Projekt speichern unter** wird das aktuelle Projekt mit einem neuen Namen **gespeichert**, den Sie eingeben können, wenn Sie vom Programm dazu aufgefordert werden.

28.3.6 Versionskontrolle

Ihre Altova-Applikation unterstützt Microsoft Source-Safe und andere kompatible Speicherschnittstellen. Die von Altova-Applikationen unterstützten Versionskontrollsysteme sind im Abschnitt Unterstützte Versionskontrollsysteme[1118] aufgelistet. In diesem Abschnitt werden die Befehle aus dem Untermenü **Projekt | Versionskontrolle** beschrieben, mit denen Sie von Ihrer Altova-Applikation aus auf Ihr Versionskontrollsystem zugreifen können.

Übersicht über die Funktion "Versionskontrolle"

Um Dateien in einem Applikationsprojekt unter Versionskontrolle zu stellen, gehen Sie folgendermaßen vor:

1. In XMLSpy wird ein Projektordner mit den unter Versionskontrolle zu stellenden Dateien erstellt. Normalerweise entspricht der Projektordner einem lokalen Ordner, in dem sich die Projektdateien befinden. Der Pfad zum lokalen Ordner wird als lokaler Pfad referenziert.
2. In der Datenbank des Versionskontrollsystems (auch als Versionskontrolle oder Repository bezeichnet) wird ein Ordner für die unter Versionskontrolle zu stellenden Dateien erstellt.
3. Die Projektdateien werden mit Hilfe des Befehls **Projekt | Versionskontrolle | Zu Versionskontrolle hinzufügen**[1327] zur Versionskontrolle hinzugefügt.
4. Versionskontrollaktionen wie z.B. Einchecken, Auschecken und Entfernen von Dateien aus der Versionskontrolle können mit Hilfe der Befehle im **Projekt | Versionskontrolle**[1320] Untermenü[1320] durchgeführt werden. Die Befehle in diesem Untermenü sind in der Benutzerreferenz im Abschnitt "Projektmenü" aufgelistet und beschrieben.

Anmerkung: Der aktuelle Versionskontroll-Provider kann auf zwei Arten gewechselt werden: (i) über die Versionskontrolloptionen (**Extras | Optionen | Versionskontrolle**[1645]) oder (ii) im Dialogfeld "Versionskontrolle wechseln" (**Projekt | Versionskontrolle | Versionskontrolle wechseln**[1334]).

Anmerkung: Beachten Sie, dass ein Versionskontrollprojekt nicht dasselbe wie ein Applikationsprojekt ist. Versionskontrollprojekte sind verzeichnisabhängig, während XMLSpy-Projekte logische Konstruktionen ohne direkte Verzeichnisabhängigkeit sind.

Nähere Informationen finden Sie im Abschnitt Versionskontrolle[1115].

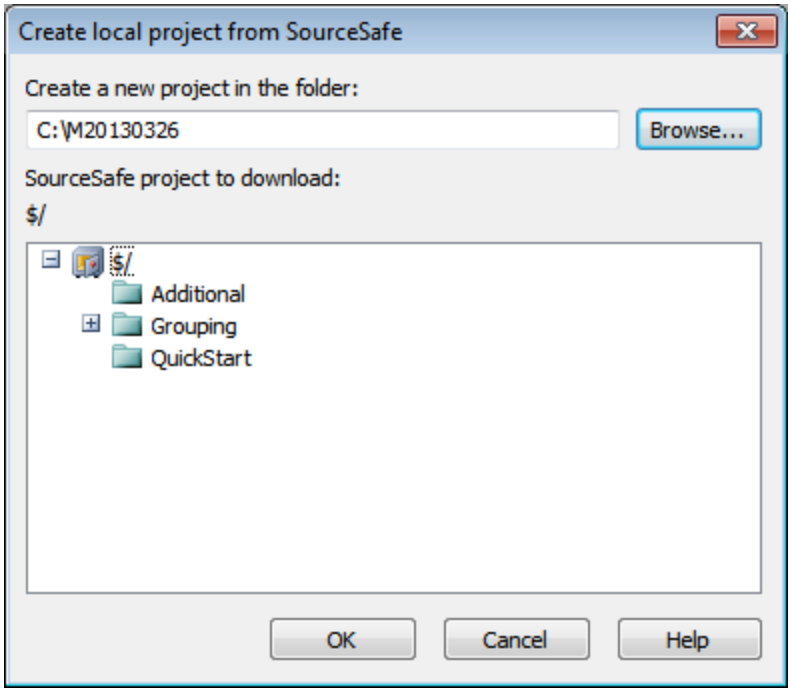
© 2016-2022 Altova GmbH

28.3.6.1 Aus Versionskontrolle öffnen

Mit dem Befehl **Aus Versionskontrolle öffnen** erstellen Sie ein neues Applikationsprojekt von einem unter Versionskontrolle gestellten Projekt.

Um ein neues Projekt zu erstellen, gehen Sie folgendermaßen vor:

1. Je nach verwendetem Versionskontrollsystem müssen Sie vor dem Erstellen eines neuen Projekts über die Versionskontrolle sicherstellen, dass keine Datei aus dem Projekt ausgecheckt ist.
2. In der Applikation kann ein Projekt geöffnet sein, dies muss aber nicht der Fall sein.
3. Wählen Sie den Befehl "Projekt | Versionskontrolle | Aus Versionskontrolle öffnen".
4. Das aktuell eingestellte Versionskontrollsystem wird mit den jeweiligen Überprüfungs- und Verbindungsdialogfeldern angezeigt. Stellen Sie eine Verbindung zum gewünschten Repository, d.h. zum gebundenen Ordner im Repository, der dem lokalen Ordner entspricht, her.
5. Navigieren Sie im Dialogfeld, das daraufhin angezeigt wird (*Abbildung unten*), zum lokalen Ordner, in den der Inhalt des gebundenen Ordners im Repository (zu dem Sie soeben eine Verbindung hergestellt haben) kopiert werden soll. In der Abbildung unten hat der gebundene Ordner den Namen `MyProject` und wird durch das Zeichen `$` repräsentiert; der lokale Ordner ist `C:\M20130326`.



6. Klicken Sie auf **OK**. Der Inhalt des gebundenen Ordners (`MyProject`) wird in den lokalen Ordner `C:\M20130326`. kopiert und ein Dialogfeld wird aufgerufen, in dem Sie aufgefordert werden, die Projektdatei (`.spp` Datei) auszuwählen, die als das neue Projekt erstellt werden soll.
7. Wählen Sie die `.spp` Datei aus, die in den lokalen Ordner kopiert worden sein sollte. In unserem Beispiel handelt es sich um `MyProject.spp` im Ordner `C:\M20130326`. In der Applikation wird ein neues Projekt namens `MyProject` erstellt und im Fenster "Projekt" angezeigt. Die Dateien des Projekts werden im Ordner `C:\M20130326` gespeichert.

Symbole der Versionskontrolle:

Die Dateien und Projektordner sind mit bestimmten Symbolen versehen, deren Bedeutung im Folgenden erklärt wird.

	Eingecheckt. Die Datei kann ausgecheckt werden.
	Die Datei wurde von einem anderen Benutzer ausgecheckt und kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Die Datei kann bearbeitet und eingecheckt werden.

28.3.6.2 Versionskontrolle aktivieren

Mit Hilfe dieses Befehls können Sie die Versionskontrolle für ein Applikationsprojekt aktivieren oder deaktivieren. Bei Auswahl dieser Option für eine Datei oder einen Ordner, wird die Versionskontrolle für das gesamte Projekt aktiviert/deaktiviert. Nach Aktivierung der Versionskontrolle wird der Ein- und Auscheckstatus der verschiedenen Dateien im Projektfenster angezeigt.

	Eingecheckt. Die Datei kann ausgecheckt werden.
	Die Datei wurde von einem anderen Benutzer ausgecheckt und kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Die Datei kann bearbeitet und eingecheckt werden.

28.3.6.3 Aktuellste Version holen

Mit dem Befehl "Aktuellste Version holen" (Menü **Projekt | Versionskontrolle**) wird die aktuellste Version der ausgewählten Datei(en) aus dem Versionskontrollspeicher in das Arbeitsverzeichnis geholt. Die Dateien werden als schreibgeschützte Dateien und nicht ausgecheckt abgerufen. Dieser Befehl funktioniert wie der **Befehl Abrufen** 1323, aber das Dialogfeld "Abrufen" wird nicht angezeigt.

Wenn die ausgewählten Dateien derzeit ausgecheckt sind, hängt es von Ihrem Versionskontrollsystem ab, was geschieht. Normalerweise werden Sie gefragt, ob die ausgecheckte Datei ersetzt, zusammengeführt oder so belassen werden soll.

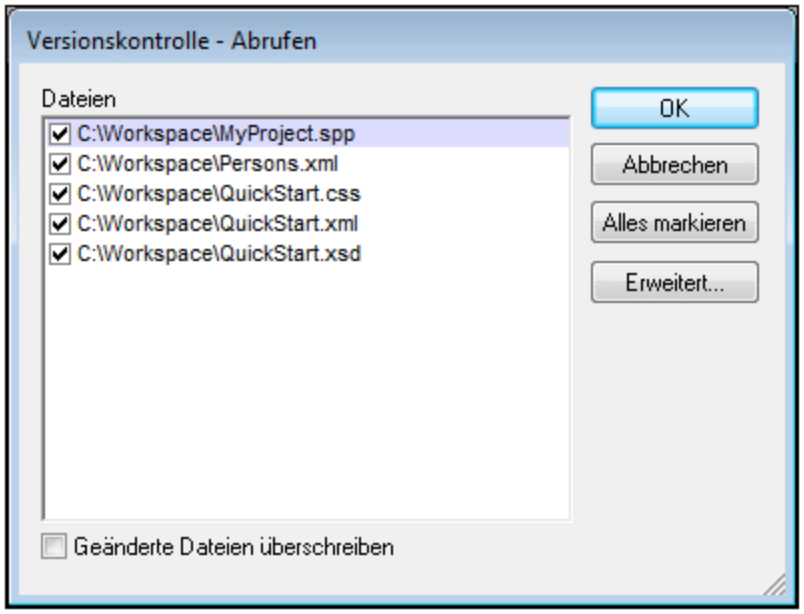
Anmerkung: Dieser Befehl ist bei Ausführung an einem Ordner rekursiv, d.h. er wird auf alle Dateien unterhalb des aktuellen Ordners in der Ordnerhierarchie angewendet.

© 2016-2022 Altova GmbH

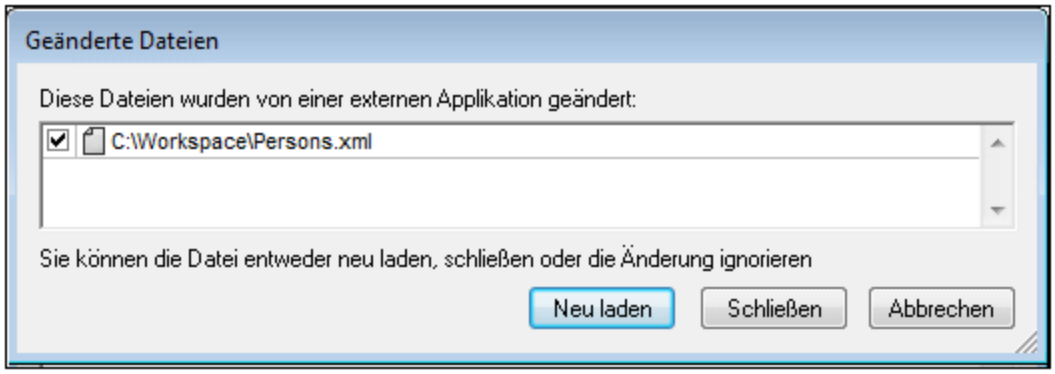
28.3.6.4 Abrufen, Ordner abrufen

Mit dem Befehl **Abrufen** (im Menü **Projekt | Versionskontrolle**) werden Dateien aus dem Repository als schreibgeschützte Dateien abgerufen. (Um eine Datei bearbeiten zu können, müssen Sie sie auschecken.) Im Dialogfeld "Abrufen" werden die Dateien im Objekt (Projekt oder Ordner), an denen der Befehl "Abrufen" ausgeführt wurde, aufgelistet (*siehe Abbildung unten*). Sie können die abzurufenden Dateien markieren, indem Sie sie mit einem Häkchen versehen.

Anmerkung: Mit dem Befehl **Ordner abrufen** können Sie einzelne Unterordner im Repository auswählen, falls dies Ihr Versionskontrollsystem gestattet.

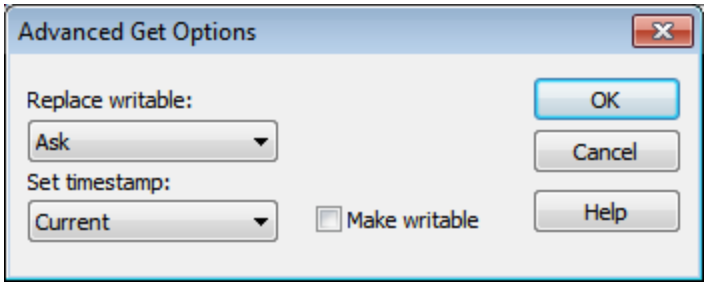


Sie können auswählen, ob geänderte ausgecheckte Dateien überschrieben werden sollen, indem Sie die entsprechende Option am unteren Rand des Dialogfelds "Abrufen" aktivieren. Wenn Sie auf **OK** klicken, werden die Dateien überschrieben. Wenn eine der überschriebenen Dateien gerade geöffnet ist, wird ein Dialogfeld angezeigt (*Abbildung unten*), in dem Sie gefragt werden, ob die Datei(en) neu geladen (Schaltfläche **Neu laden**), geschlossen (Schaltfläche **Schließen**) werden sollen oder ob die aktuelle Ansicht der Datei beibehalten werden soll (**Abbrechen**).



Erweiterte Abrufoptionen

Das Dialogfeld "Erweiterte Abrufoptionen" (*Abbildung unten*) wird über die Schaltfläche **Erweitert** im Dialogfeld "Abrufen" (*siehe erste Abbildung in diesem Abschnitt*) aufgerufen.



Hier können Sie Optionen zum (i) Ersetzen ausgecheckter beschreibbarer Dateien (ii) zum Zeitstempel und (iii) ob die Eigenschaft "Schreibgeschützt" der abgerufenen Datei geändert werden soll, damit die Datei beschreibbar wird, einstellen.

28.3.6.5 Auschecken, Einchecken

Nachdem Sie eine Projektdatei unter Versionskontrolle gestellt haben, kann sie durch Auswahl der Datei (im Projektfenster) und Klicken auf den entsprechenden Befehl (**Einchecken/Auschecken**) im Menü **Projekt | Versionskontrolle** ein- und ausgecheckt werden.

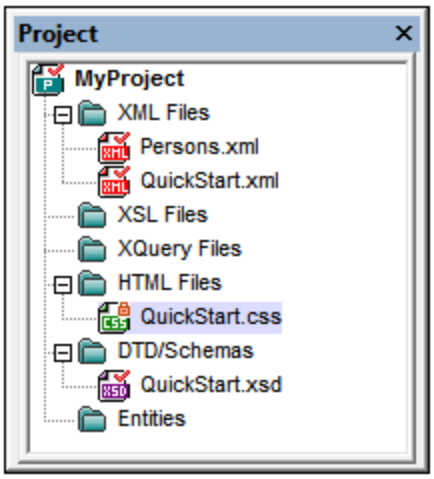
Wenn eine Datei ausgecheckt wird, wird eine Kopie aus dem Repository in den lokalen Ordner platziert. Eine ausgecheckte Datei kann bearbeitet werden. Wenn eine Datei, die sich unter Versionskontrolle befindet, nicht ausgecheckt ist, kann sie nicht bearbeitet werden. Nachdem eine Datei bearbeitet wurde, können die Änderungen durch Einchecken der Datei im Repository gespeichert werden. Selbst wenn die Datei nicht gespeichert wurde, werden die Änderungen an der Datei durch Einchecken im Repository gespeichert. Der Ein- und Auscheckstatus einer Datei wird durch ein Häkchen bzw. ein Schloßsymbol im Dateisymbol angezeigt.

Zu Dateien und dem Projektordner können bestimmte Symbole angezeigt werden, deren Bedeutung unten erklärt ist.

	Eingecheckt. Für das Auschecken verfügbar.
	Von einem anderen Benutzer ausgecheckt. Kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Kann bearbeitet und eingecheckt werden.

Durch Auswahl des Projekts oder eines Ordners im Projekt werden alle Dateien im ausgewählten Objekt ausgewählt. Um mehrere Objekte (Dateien und Ordner) auszuwählen, halten Sie beim Anklicken der Objekte die Strg-Taste gedrückt. In der Abbildung unten sehen Sie ein ausgechecktes Projekt. Die Datei `QuickStart.css` wurde im Anschluss daran eingecheckt.

© 2016-2022 Altova GmbH



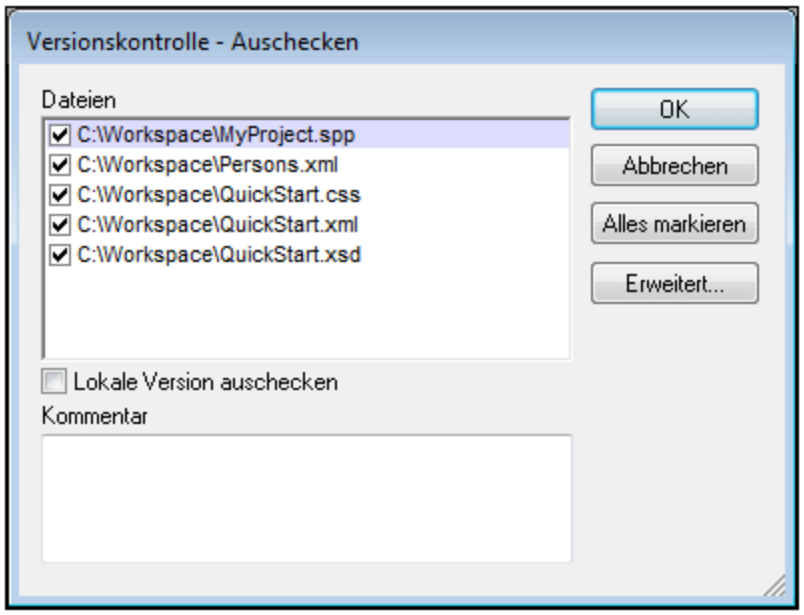
Speichern und Ablehnen von Bearbeitungen

Beachten Sie, dass Sie eine Datei beim Einchecken ausgecheckt lassen können. Dabei werden die Änderungen im Repository gespeichert, während die Datei weiterhin ausgecheckt bleibt. Diese Option ist nützlich, wenn Sie Ihre Änderungen regelmäßig im Repository speichern und mit der Bearbeitung fortfahren möchten.

Wenn Sie eine Datei ausgecheckt haben, diese bearbeitet haben und die Änderungen anschließend aber nicht übernehmen möchten, können Sie die im Repository gespeicherte Dokumentversion in Ihrem lokalen Ordner wiederherstellen, indem Sie den Befehl **Projekt | Versionskontrolle | Auschecken rückgängig machen** auswählen.

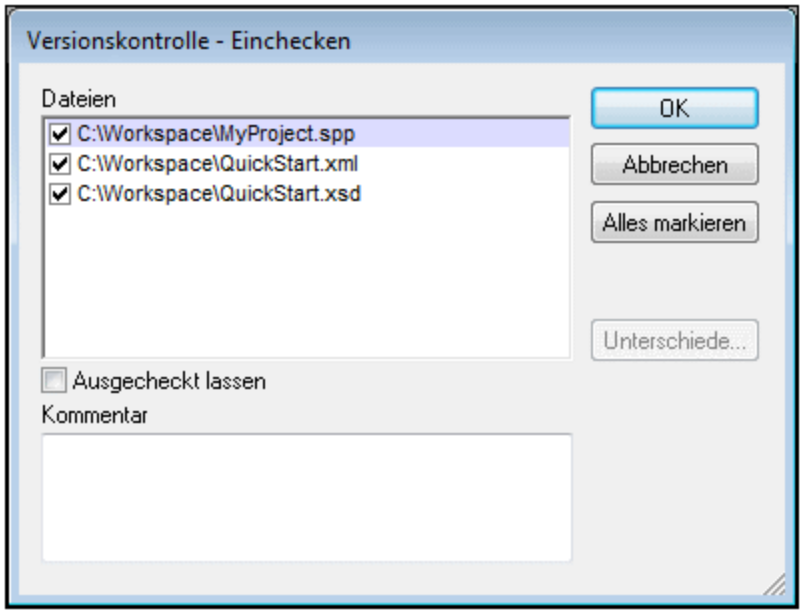
Auschecken

Im Dialogfeld "Auschecken" (*Abbildung unten*) können Sie (i) die auszucheckenden Dateien auswählen und (ii) wählen, ob die Repository-Version oder die lokale Version ausgecheckt werden soll.



Einchecken

Im Dialogfeld "Einchecken" (*Abbildung unten*) können Sie (i) die einzucheckenden Dateien auswählen und (ii) wählen, ob die Datei ausgecheckt bleiben soll.



Anmerkung: In beiden Dialogfeldern (Einchecken und Auschecken) werden mehrere Dateien angezeigt, wenn das ausgewählte Objekt (Projekt oder Projektordner) mehrere Dateien enthält.

© 2016-2022 Altova GmbH

28.3.6.6 Auschecken rückgängig...

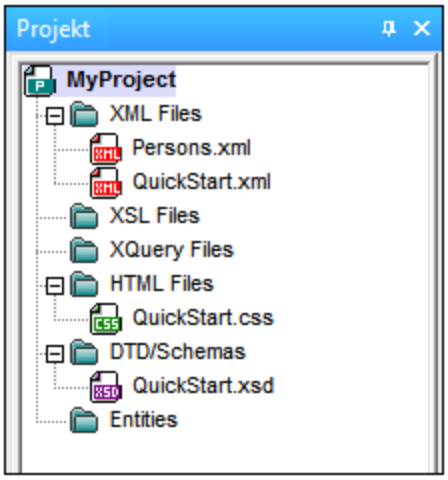
Wenn Sie eine Datei ausgecheckt und bearbeitet haben und diese Änderung nicht beibehalten möchten, können Sie die im Repository gespeicherte Dokumentversion wiederherstellen. Wählen Sie dazu den Befehl **Projekt | Versionskontrolle | Auchecken rückgängig...**.

Die Ordner und Dateien sind mit bestimmten Symbolen versehen, deren Bedeutung im Folgenden erklärt wird.

	Eingecheckt. Die Datei kann ausgecheckt werden.
	Die Datei wurde von einem anderen Benutzer ausgecheckt und kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Die Datei kann bearbeitet und eingecheckt werden.

28.3.6.7 Zu Versionskontrolle hinzufügen

Nachdem ein Projekt zu einer Versionskontrolle hinzugefügt wurde, können Sie Dateien entweder einzeln oder in Gruppen zur Versionskontrolle hinzufügen. Wählen Sie die Datei im Fenster "Projekt" aus und klicken Sie auf den Befehl **Projekt | Versionskontrolle | Zu Versionskontrolle hinzufügen**. Um mehrere Dateien auszuwählen, halten Sie beim Anklicken der Dateien die **Strg**-Taste gedrückt. Wenn Sie den Befehl an einem (grünen) Projektordner (*siehe Abbildung unten*) ausführen, werden alle Dateien im Ordner und seinen Unterordnern zur Versionskontrolle hinzugefügt.



Wenn Dateien zur Versionskontrolle hinzugefügt werden, wird die lokale Ordnerhierarchie im Repository repliziert (jedoch nicht die Projektordnerhierarchie). Wenn eine Datei sich also im lokalen Ordner in einem Unterordner mehrere Ebenen unterhalb des lokalen Ordners befindet, so werden im Repository automatisch auch der der Datei übergeordnete Ordner und alle darüber liegenden erstellt.

Wenn die erste Datei aus einem Projekt zur Versionskontrolle hinzugefügt wird, werden im Repository die korrekten Bindings erstellt und die Projektdatei (`.spp` Datei) wird automatisch hinzugefügt. Nähere Informationen dazu finden Sie im Abschnitt Zur Versionskontrolle hinzufügen[1123].

Versionskontrollsymbole

Zu Dateien und dem Projektordner können bestimmte Symbole angezeigt werden, deren Bedeutung unten erklärt ist.

	Eingecheckt. Für das Auschecken verfügbar.
	Von einem anderen Benutzer ausgecheckt. Kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Kann bearbeitet und eingecheckt werden.

28.3.6.8 Von Versionskontrolle ausgliedern

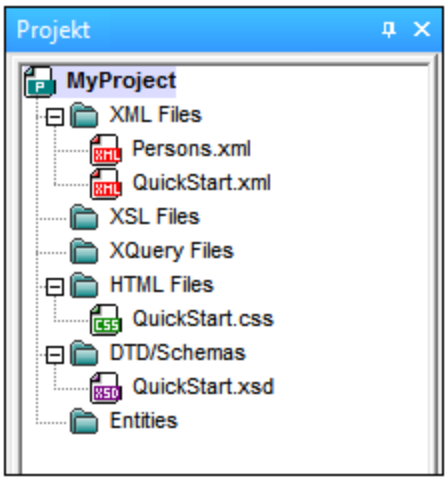
Um eine Datei aus der Versionskontrolle zu entfernen, wählen Sie sie aus und klicken Sie auf den Befehl **Projekt | Versionskontrolle | Von Versionskontrolle ausgliedern**. Außerdem können Sie: (i) Dateien in einem Projektordner durch Ausführen des Befehls am Ordner entfernen, (ii) durch Drücken der **Strg**-Taste mehrere Dateien auswählen und entfernen und (iii) das gesamte Projekt durch Ausführen des Befehls am Projekt entfernen.

28.3.6.9 Aus Versionskontrolle freigeben

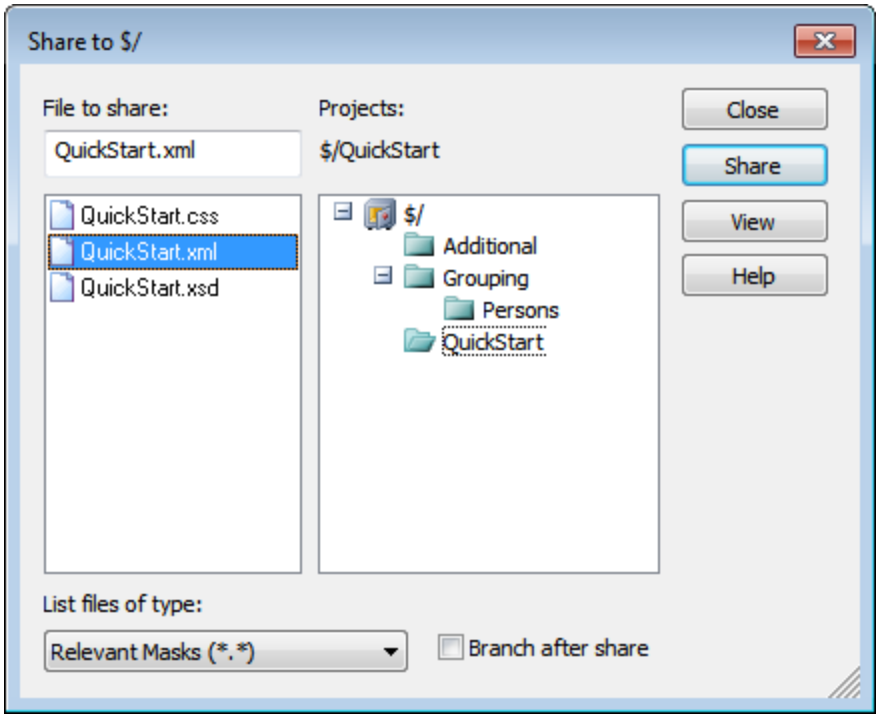
Der Befehl **Aus Versionskontrolle freigeben** wird unterstützt, wenn das verwendete Versionskontrollsystem die Freigabe von Dateien unterstützt. Sie können eine Datei freigeben, sodass sie auf mehreren lokalen Rechnern verfügbar ist. Wenn an einer dieser Dateien eine Änderung vorgenommen wird, wird dies in allen anderen freigegebenen Versionen übernommen.

Wählen Sie das (*in der Abbildung unten markierte)* Projekt im Fenster "Projekt" der Applikation aus und klicken Sie anschließend auf den Befehl **Aus Versionskontrolle freigeben**.

© 2016-2022 Altova GmbH



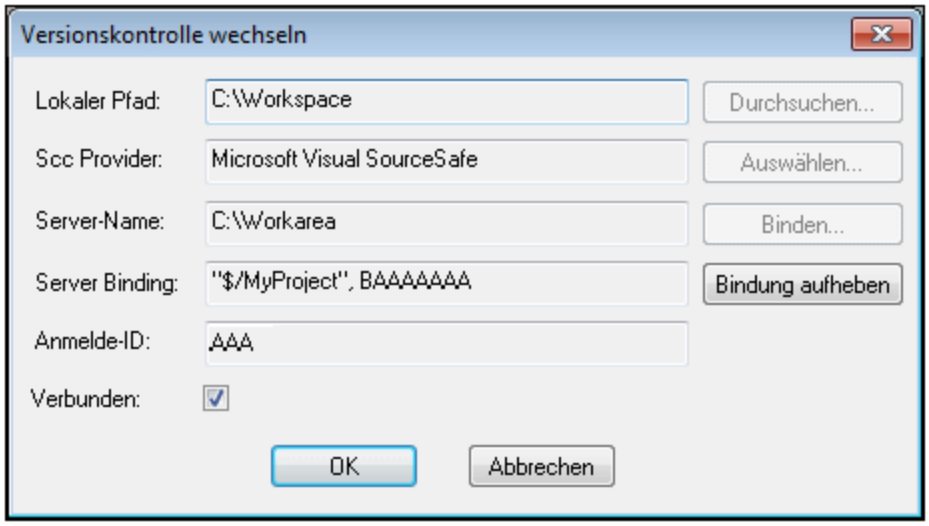
Daraufhin wird das Dialogfeld "Share To [Ordner]" (*Abbildung unten*) angezeigt.



Um die gewünschten Dateien auszuwählen, wählen Sie zuerst den Ordner, in dem sich die Dateien befinden, in der Projektstruktur auf der rechten Seite aus. Die Dateien im ausgewählten Ordner werden im Fenster auf der linken Seite angezeigt. Wählen Sie die freizugebende Datei aus (um mehrere Dateien auszuwählen, halten Sie beim Anklicken die **Strg**-Taste gedrückt). Die ausgewählten Datei(en) werden im Textfeld *Files to Share* (*links oben*) angezeigt. Die Dateien verschwinden aus dem Fenster auf der linken Seite. Klicken Sie auf **Freigeben** und anschließend **Schließen,** um die ausgewählte(n) Datei(en) in den lokalen freigegebenen Ordner zu kopieren.

Der freigegebene Ordner wird im Namen des Dialogfelds "Share to [Ordner]" angegeben. In der Abbildung oben ist es der lokale Ordner (da sich das Zeichen `$` in dem Ordner im Repository, an den der lokale Ordner

gebunden ist, befindet). Sie können den freigegebenen Ordner im Dialogfeld "Versionskontrolle wechseln" (*Abbildung unten,* **Versionskontrolle wechseln***)* sehen und den Ordner durch Ändern des lokalen Pfads und Binding einstellen.



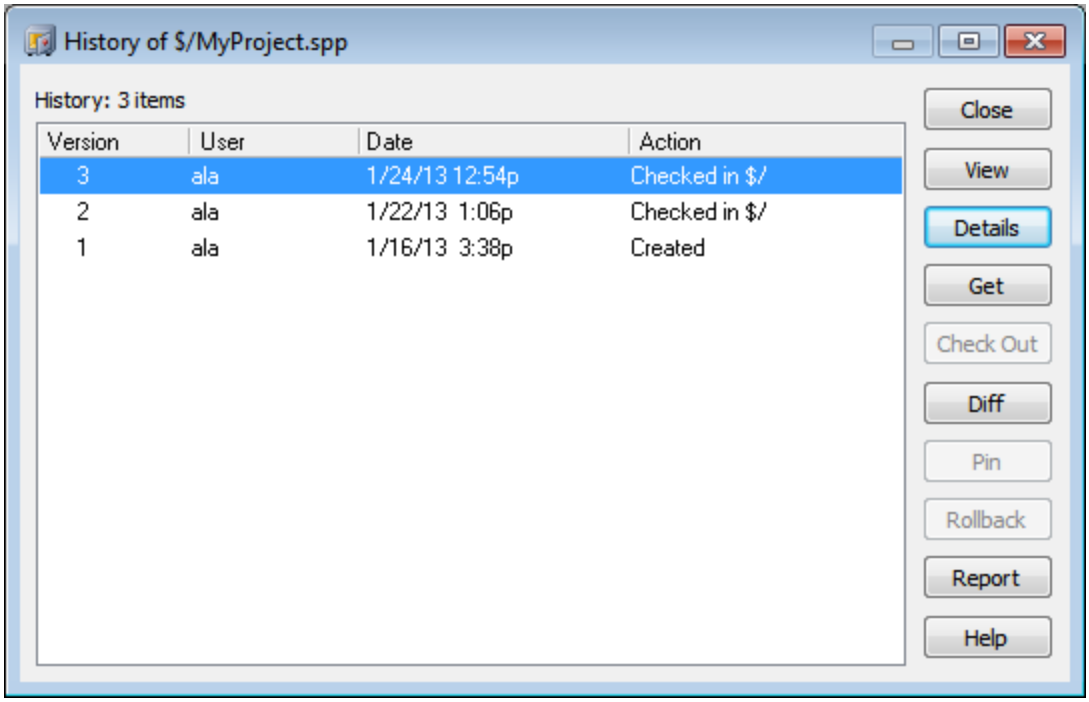
Nähere Informationen zum Freigeben von Dateien über Ihr Versionskontrollsystem finden Sie in der Benutzerdokumentation zu Ihrem Versionskontrollsystem.

28.3.6.10 Verlauf anzeigen

Mit dem Befehl **Verlauf anzeigen** aktivieren Sie die Verlaufsfunktion des aktiven Versionskontrollsystems. Damit zeigen Sie den Verlauf der im Projektfenster ausgewählten Datei an. Um den Verlauf der Projektdatei (`.spp` Datei) anzuzeigen, wählen Sie den Projekttitel aus. Sie können Informationen über die früheren Versionen einer Datei und die Unterschiede anzeigen und frühere Versionen der Datei abrufen.

In der Abbildung unten sehen Sie das Dialogfeld "Verlauf" des Versionskontrollsystems Visual SourceSafe. Darin werden die verschiedenen Versionen der Datei `MyProject.spp` aufgelistet.

© 2016-2022 Altova GmbH



Dieses Dialogfeld bietet verschiedene Möglichkeiten, um bestimmte Versionen der ausgewählten Datei zu vergleichen und abzurufen. Wenn Sie auf einen Eintrag in der Liste doppelklicken, wird das Dialogfeld "Verlauf" für diese Datei geöffnet. Die Schaltflächen in Dialogfeld haben die folgenden Funktionen:

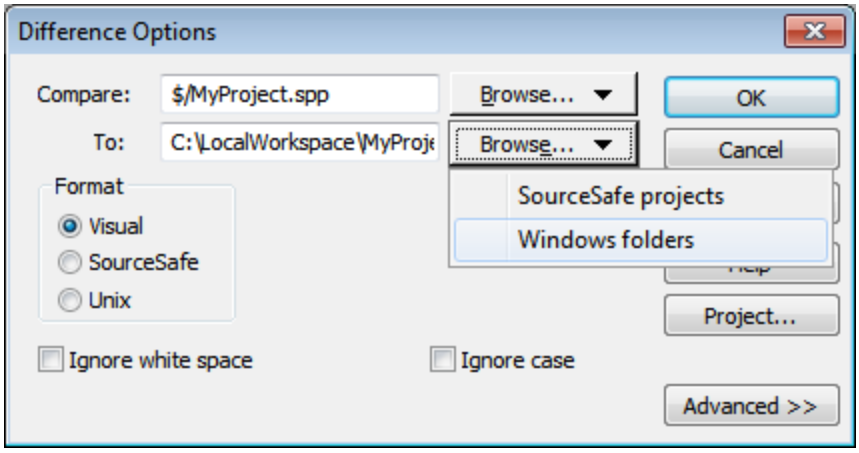
- *Schließen*: Schließt dieses Dialogfeld.
- *Ansicht*: Öffnet ein weiteres Dialogfeld, in dem Sie die Art des Ansichtsprogramms auswählen können, in dem Sie die Datei anzeigen möchten.
- *Details*: Öffnet ein Dialogfeld, in dem Sie die Eigenschaften[1333] der gerade aktiven Datei sehen.
- *Abrufen*: Damit können Sie eine der vorherigen Versionen der Datei aus der Versionsliste in das Arbeitsverzeichnis holen.
- *Auschecken*: Damit können eine frühere Version der Datei auschecken.
- *Diff*: Öffnet das Dialogfeld Vergleichsoptionen[1331], in dem Sie die Vergleichsoptionen zur Anzeige der Unterschiede zwischen den beiden Dateiversionen definieren können. Mit Hilfe von **Strg + Klick** können Sie zwei Dateiversionen in diesem Fenster markieren. Klicken Sie anschließend auf Diff, um die Unterschiede zwischen den beiden Dateien anzuzeigen.
- *Pin*: Markiert eine Version der Datei mit einem Pin bzw. hebt die Marierung auf. Damit können Sie die Dateiversion definieren, die für den Dateivergleich verwendet werden soll.
- *Rollback*: Führt ein Rollback für die ausgewählte Version der Datei durch.
- *Report*: Generiert einen Verlaufsbericht, den Sie an den Drucken, die Datei oder die Zwischenablage senden können.
- *Hilfe*: Öffnet die Online-Hilfe des Versionskontrollanbieters für das Plugin.

28.3.6.11 Unterschiede anzeigen

Der Befehl **Unterschiede anzeigen** ist verfügbar, wenn eine Datei im Projektfenster ausgewählt ist. Um die Projektdatei (`.spp` Datei) auszuwählen, wählen Sie den Titel des Projekts im Projektfenster aus. Mit dem

Befehl **Unterschiede anzeigen** wird das Vergleichstool des Versionskontrollsystems gestartet, sodass Sie die Unterschiede zwischen den Dateien direkt in Ihrer Altova-Applikation überprüfen können.

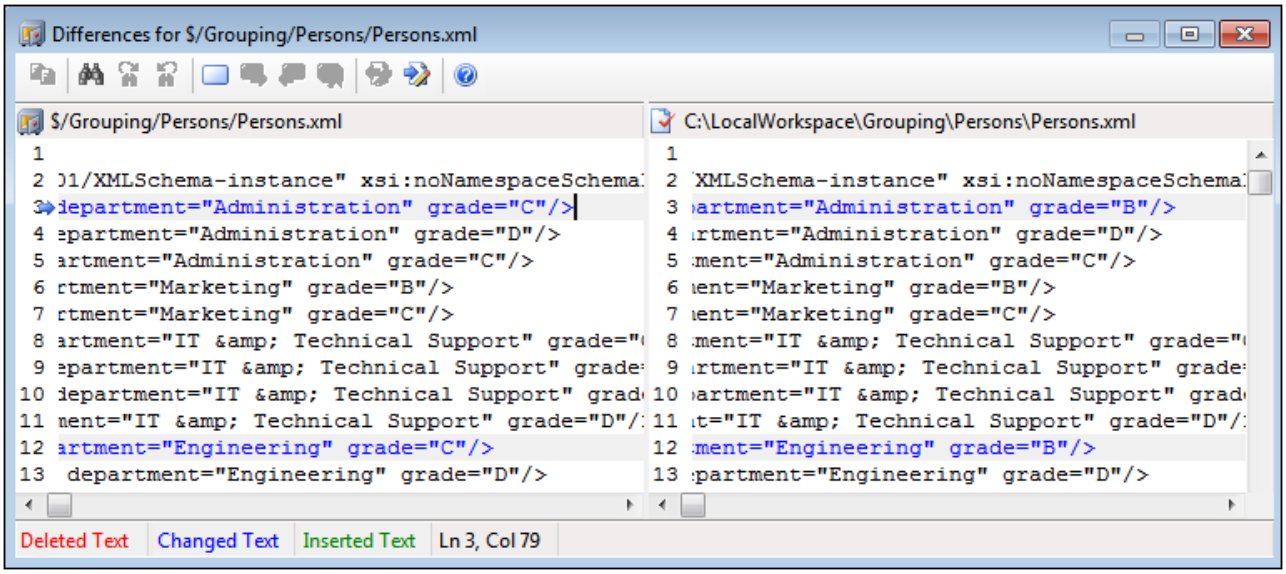
In der Abbildung unten sehen Sie das Vergleichstool des Versionskontrollsystems Visual SourceSafe.



Standardmäßig werden in den Textfeldern *Compare* und *To* die Repository-Version bzw. die lokale Version angezeigt. Folgendermaßen können Sie zu anderen Dateien navigieren:

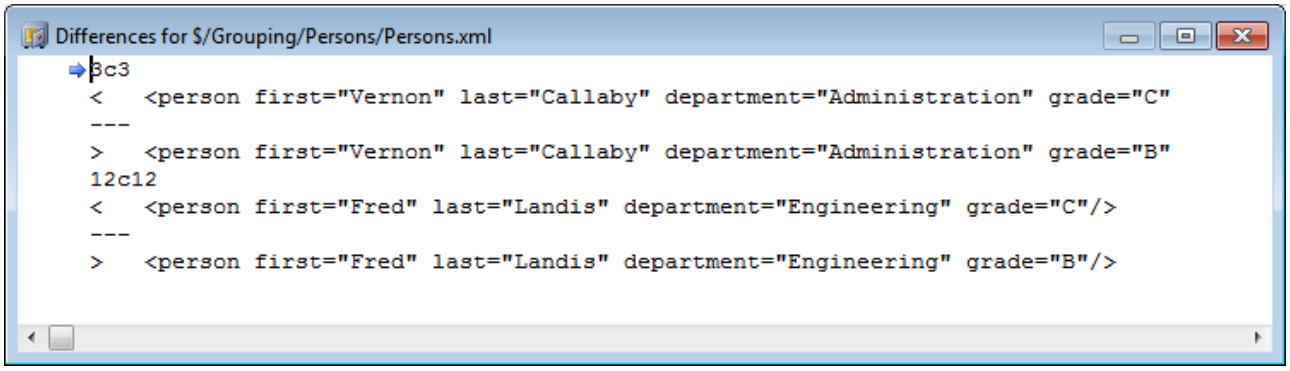
1. Über die Dropdown-Liste der Schaltfläche **Durchsuchen** können Sie SourceSafe Projekte (zum Durchsuchen der Repository-Dateien) oder Windows-Ordner (zum Durchsuchen lokaler Ordner) auswählen.
2. Navigieren Sie zu den gewünschten Dateien und wählen Sie sie aus.

Wählen Sie die gewünschten Optionen aus und klicken Sie auf **OK,** um die Dateien zu vergleichen. Die Vergleichsergebnisse werden in einem separaten Fenster angezeigt. In den Abbildung unten sehen Sie die Ergebnisse des Vergleichs in zwei Formaten.



 © 2016-2022 Altova GmbH

In der Abbildung oben sehen Sie das Visual SourceSafe-Vergleichsergebnis (*siehe Dialogfeld "Optionen" oben*), während die Ergebnisse in der Abbildung unten im Unix-Format angezeigt werden. In beiden Abbildungen gibt es zwei Unterschiede - in beiden Fällen eine Änderung des "grade" von `C` in `B`.

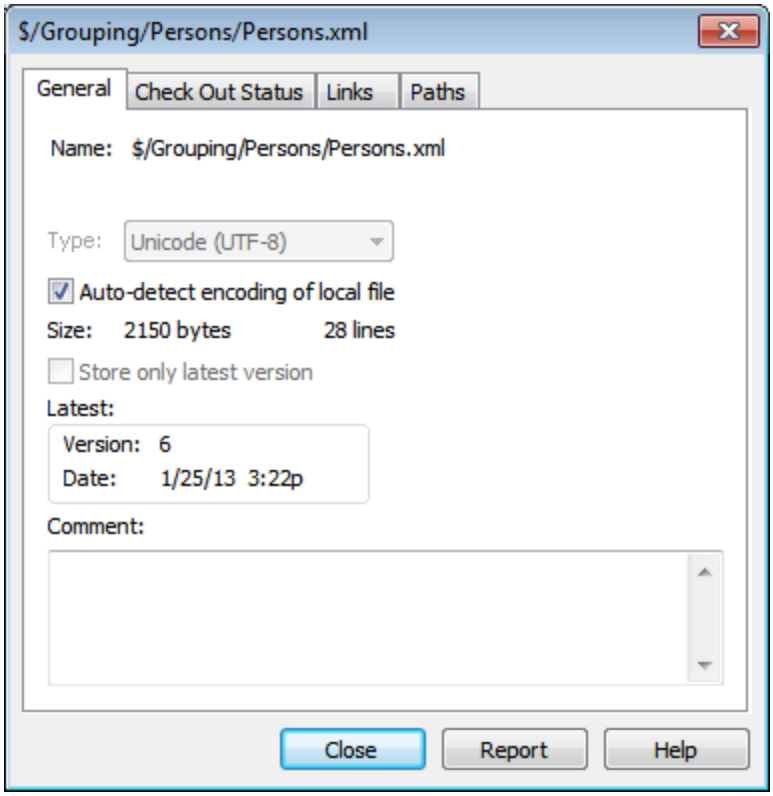


Eine ausführliche Beschreibung dazu, wie Ihr Versionskontrollsystem Unterschiede behandelt, finden Sie in der jeweiligen Produktdokumentation.

28.3.6.12 Eigenschaften anzeigen

Mit dem Befehl **Eigenschaften anzeigen** werden die Eigenschaften der gerade ausgewählten Datei angezeigt (*Abbildung unten*). Welche Eigenschaften angezeigt werden, hängt vom verwendeten Versionskontrollsystem ab. In der Abbildung unten sehen Sie die von Visual SourceSafe angezeigten Eigenschaften.

Beachten Sie, dass dieser Befehl nur für einzelne Dateien aktiv ist.



Nähere Informationen dazu finden Sie in der Dokumentation zum jeweiligen Versionskontrollsystem.

28.3.6.13 Status aktualisieren

Mit diesem Befehl **aktualisieren** Sie den Status aller Projektdateien unabhängig von ihrem derzeitigen Status.

28.3.6.14 Versionskontrollmanager

Mit diesem Befehl **starten** Sie die native Benutzeroberfläche Ihrer Versionskontroll-Software.

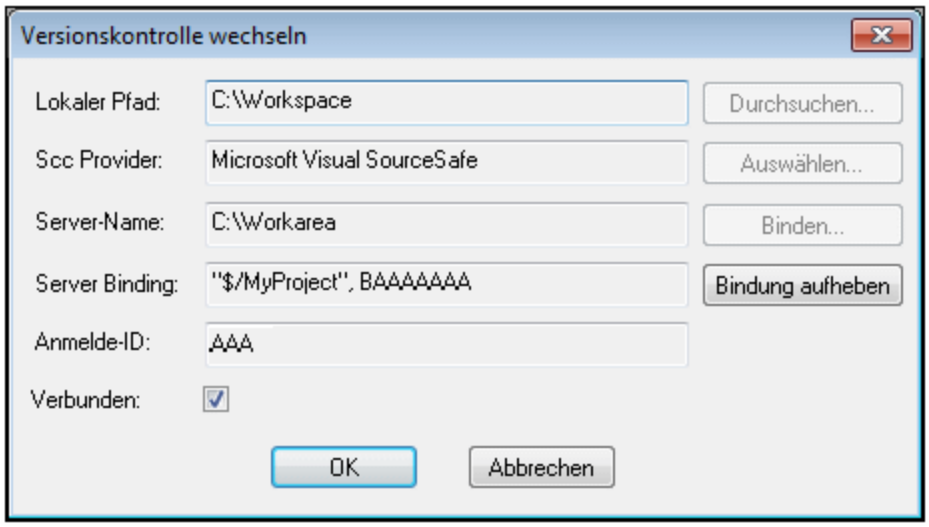
28.3.6.15 Versionskontrolle wechseln

Über das aktuelle Binding stellt das aktive Applikationsprojekt eine Verbindung zur Versionskontrolldatenbank her, daher muss das aktuelle Binding korrekt sein, d.h. die Applikationsprojektdatei (`.spp` Datei) muss sich im Ordner unter dem lokalen Pfad befinden und der gebundene Ordner im Repository muss die Datenbank sein, in der die Dateien dieses Projekts gespeichert sind. Normalerweise entsprechen der gebundene Ordner und seine Substruktur dem lokalen Arbeitsbereichordner und dessen Substruktur.

© 2016-2022 Altova GmbH

Im Dialogfeld "Versionskontrolle wechseln" (*Abbildung unten*) können Sie das Versionskontrollsystem (*SCC Provider*), den lokalen Ordner (*Lokaler Pfad*) und das Repository-Binding (*Server-Name* und *Server Binding*) wechseln.

Die Einstellungen können erst geändert werden, wenn Sie die aktuelle Bindung aufgehoben haben. Verwenden Sie dazu die Schaltfläche **Bindung aufheben**. Alle Einstellungen können nun bearbeitet werden.



Ändern Sie die Versionskontrolleinstellungen folgendermaßen:

1. Navigieren Sie über die Schaltfläche **Durchsuchen** zum lokalen Ordner und klicken Sie auf die Schaltfläche **Auswählen,** um eines der installierten Versionskontrollsysteme auszuwählen.
2. Anschließend können Sie den lokalen Ordner an eine Repository-Datenbank binden. Klicken Sie dazu auf die Schaltfläche **Binden**. Daraufhin wird das Verbindungsdialogfeld Ihres Versionskontrollsystems aufgerufen.
3. Wenn Sie eine *Anmelde-ID* eingegeben haben, wird diese an das Versionskontrollsystem übergeben; andernfalls müssen Sie Ihre Anmeldeinformationen im Verbindungsdialogfeld eingeben.
4. Wählen Sie im Repository die Datenbank aus, an die Sie diesen lokalen Ordner binden möchten. Eventuell müssen diese Einstellungen in mehreren Dialogfeldern vorgenommen werden.
5. Nachdem Sie alle Einstellungen vorgenommen haben, klicken Sie im Dialogfeld "Versionskontrolle wechseln" auf **OK**.

28.3.7 Dateien zu Projekt hinzufügen



Mit diesem Befehl fügen Sie Dateien in das aktuelle Projekt ein. Verwenden Sie diesen Befehl um Dateien in einen beliebigen Ordner Ihres Projekts einzufügen. Sie können einzelne Dateien oder Dateigruppen (Auswahl mit **Strg + Klick**) im Dialogfeld "Öffnen" hinzufügen. Wenn Sie Dateien in ein Projekt einfügen, werden diese automatisch - nach Erweiterung sortiert - in die entsprechenden Projektordner verteilt (Einstellungen hierzu im Dialogfeld Projekteigenschaften[1344]).

28.3.8 Globale Ressource zu Projekt hinzufügen

Mit dem Befehl **Projekt | Globale Ressource zu Projekt hinzufügen** wird das Dialogfeld "Globale Ressource auswählen" aufgerufen, in dem Sie eine globale Ressource vom Typ Datei oder Ordner auswählen können, um diese zum Projekt hinzuzufügen. Bei Auswahl einer globalen Ressource vom Typ Datei wird eine Datei auf Basis der im Dialogfeld Projekteigenschaften[1344] definierten Dateierweiterung zum entsprechenden Ordner hinzugefügt. Bei Auswahl einer globalen Ressource vom Typ Ordner wird ein Ordner in einem "Datei öffnen" Dialogfeld geöffnet und Sie werden aufgefordert, eine Datei auszuwählen; die ausgewählte Datei wird auf Basis der im Dialogfeld Projekteigenschaften[1344] definierten Dateierweiterung zum entsprechenden Ordner hinzugefügt. Eine Beschreibung globaler Ressourcen finden Sie im Abschnitt "Globale Ressourcen" in dieser Dokumentation.

28.3.9 URL zu Projekt hinzufügen



Mit diesem Befehl fügen Sie eine URL in das aktuelle Projekt ein. Bei Verwendung von URLs in einem Projekt wird das Ziel der URL in das Projekt einbezogen. Wenn eine Batch-Operation an einer URL oder einer Datei mit URL ausgeführt wird, sucht XMLSpy nach dem Dokument aus der URL, und führt die gewünschte Operation aus.

28.3.10 Aktive Datei zu Projekt hinzufügen



Mit diesem Befehl fügen Sie die aktive Datei in das aktuelle Projekt ein. Wenn Sie eine Datei von der Festplatte oder über eine URL geöffnet haben, können Sie die Datei mit diesem Befehl in das aktuelle Projekt aufnehmen.

28.3.11 Aktive und verwandte Dateien zu Projekt hinzufügen



Mit diesem Befehl fügen Sie das aktive XML-Dokument und alle verwandten Dateien in das Projekt ein. Beim Arbeiten mit einem XML-Dokument, das auf einem Schema oder einer DTD basiert, fügt dieser Befehl nicht nur

© 2016-2022 Altova GmbH

das XML-Dokument zum Projekt hinzu sondern auch alle verwandten Dateien (z.B. DTDs und sämtliche geparste Entities, auf die die DTD referenziert).

Bitte beachten Sie: Dateien, die durch Processing Instructions (Verarbeitungsanweisungen) referenziert werden (z.B. XSLT-Dateien) werden nicht als verwandte Dateien behandelt.

28.3.12 Projektordner zu Projekt hinzufügen



Mit diesem Befehl fügen Sie einen neuen Ordner in das aktuelle Projekt ein. Verwenden Sie diesen Befehl, um einen neuen Ordner oder Unterordner im aktuellen Projekt anzulegen. Sie können diesen Befehl auch über das Kontextmenü (Rechtsklick auf einen Ordner im Projektfenster) aufrufen.

Anmerkung: Ein Projektordner in einen anderen Projektordner oder an eine andere Stelle im selben Projekt gezogen werden. Zusätzlich dazu kann ein Ordner aus dem Windows-Datei-Explorer in jeden beliebigen Projektordner gezogen werden.

Anmerkung: Projektordner sind grün, während externe Ordner[1337] gelb sind.

28.3.13 Externes Verzeichnis zum Projekt hinzufügen

Mit dem Befehl **Projekt | Externes Verzeichnis zum Projekt hinzufügen** fügen Sie einen neuen externen Ordner in das aktuelle Projekt ein. Verwenden Sie diesen Befehl, um einen lokalen Ordner oder einen Netzwerkordner zum aktuellen Projekt hinzuzufügen. Sie können diesen Befehl auch über das Kontextmenü (Rechtsklick auf ein Verzeichnis im Projektfenster) aufrufen.

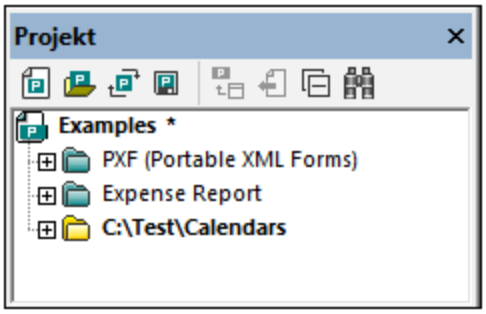
Anmerkung: Externe Ordner sind gelb, während Projektordner[1337] grün sind.

Anmerkung: In externen Ordnern enthaltene Dateien können nicht unter Versionskontrolle gestellt werden.

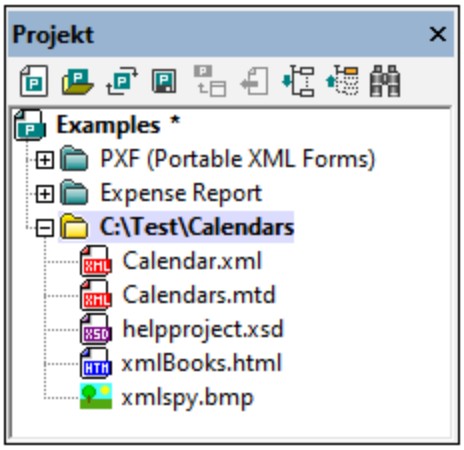
Hinzufügen von externen Ordnern zu Projekten

So fügen Sie einen externen Ordner zum Projekt hinzu:

1. Wählen Sie den Menübefehl **Projekt | Externes Verzeichnis zum Projekt hinzufügen**.
2. Wählen Sie den gewünschten Ordner aus und klicken Sie auf **OK**. Das ausgewählte Verzeichnis wird nun im Projektfenster angezeigt.



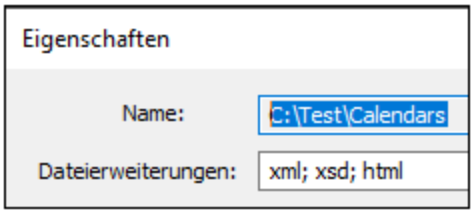
3. Klicken Sie auf das Plus-Symbol, um den Ordnerinhalt zu sehen.



Filtern von Ordnerinhalt

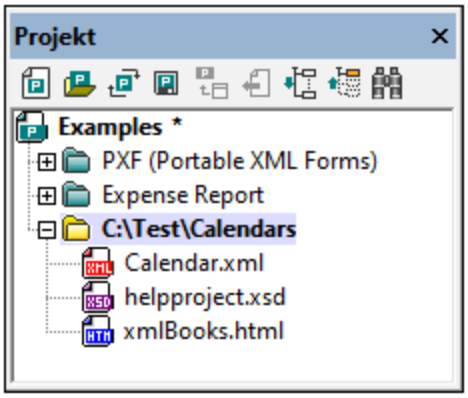
So filtern Sie den Inhalt eines Ordners:

1. Rechtsklicken Sie auf den externen von Ihnen hinzugefügten Ordner. Wählen Sie den Befehl **Eigenschaften**. Daraufhin wird das Dialogfeld "Eigenschaften" geöffnet.



2. Klicken Sie in das Feld *Dateierweiterungen* und geben Sie die Erweiterungen der Dateietypen an, die angezeigt werden sollen. Bei mehreren Erweiterungen trennen Sie diese durch ein Semikolon (*siehe Abbildung oben*).
3. Klicken Sie zur Bestätigung auf **OK**.

© 2016-2022 Altova GmbH



Im Projektfenster werden jetzt ausschließlich die ausgewählten Dateiarten angezeigt.

Validieren von externen Ordnern

So validieren Sie ein externes Verzeichnis und prüfen es auf Wohlgeformtheit:

1. Wählen Sie aus, welche Dateiarten angezeigt werden sollen oder markieren Sie diese im externen Ordner.
2. Wählen Sie den Ordner aus und klicken Sie anschließend auf den Befehl **XML | Wohlgeformtheit prüfen** oder **XML validieren** (Kürzel **F7** bzw. **F8**). Alle im Ordner angezeigten Dateien werden nun überprüft. Falls eine Datei nicht wohlgeformt oder ungültig ist, wird diese im Hauptfenster geöffnet. Hier kann der Fehler korrigiert werden.
3. Korrigieren Sie den Fehler, und starten Sie die Überprüfung erneut, um den Ordner erneut zu überprüfen.

Aktualisieren eines Projektverzeichnisses

In lokalen oder netzwerkgebundenen Verzeichnissen können jederzeit Dateien gelöscht oder hinzugefügt werden. Um die Verzeichnisansicht zu aktualisieren, rechtsklicken Sie auf den externen Ordner und wählen Sie im Kontextmenü die Option **Aktualisieren**.

Löschen von externen Verzeichnissen und Dateien

Um ein Verzeichnis aus dem Projektfenster zu löschen, klicken Sie auf das **Verzeichnis** und drücken Sie die **Entf-**Taste. Klicken Sie alternativ dazu mit der rechten Maustaste auf den externen Ordner und wählen Sie den Befehl **Löschen**. In beiden Fällen wird der externe Ordner nur aus dem Projektfenster gelöscht und nicht von der Festplatte bzw. im Netzwerk.

Um eine Datei in einem externen Ordner zu löschen, müssen Sie diese physisch von der Festplatte bzw. vom Netzwerk löschen. Um die Änderungen im Projekt zu sehen, müssen Sie den Inhalt des externen Ordners (durch Rechtsklick auf den externen Ordner und Auswahl des Befehls **Aktualisieren**) aktualisieren.

Anmerkung: Ein externer Ordner kann in einen Projektordner oder an eine andere Stelle im selben Projekt (aber nicht in einen anderen externen Ordner) gezogen werden. Zusätzlich dazu kann ein Ordner aus dem Windows-Datei-Explorer an jede beliebige Stelle im Projektfenster (mit Ausnahme eines anderen externen Ordners) gezogen werden.

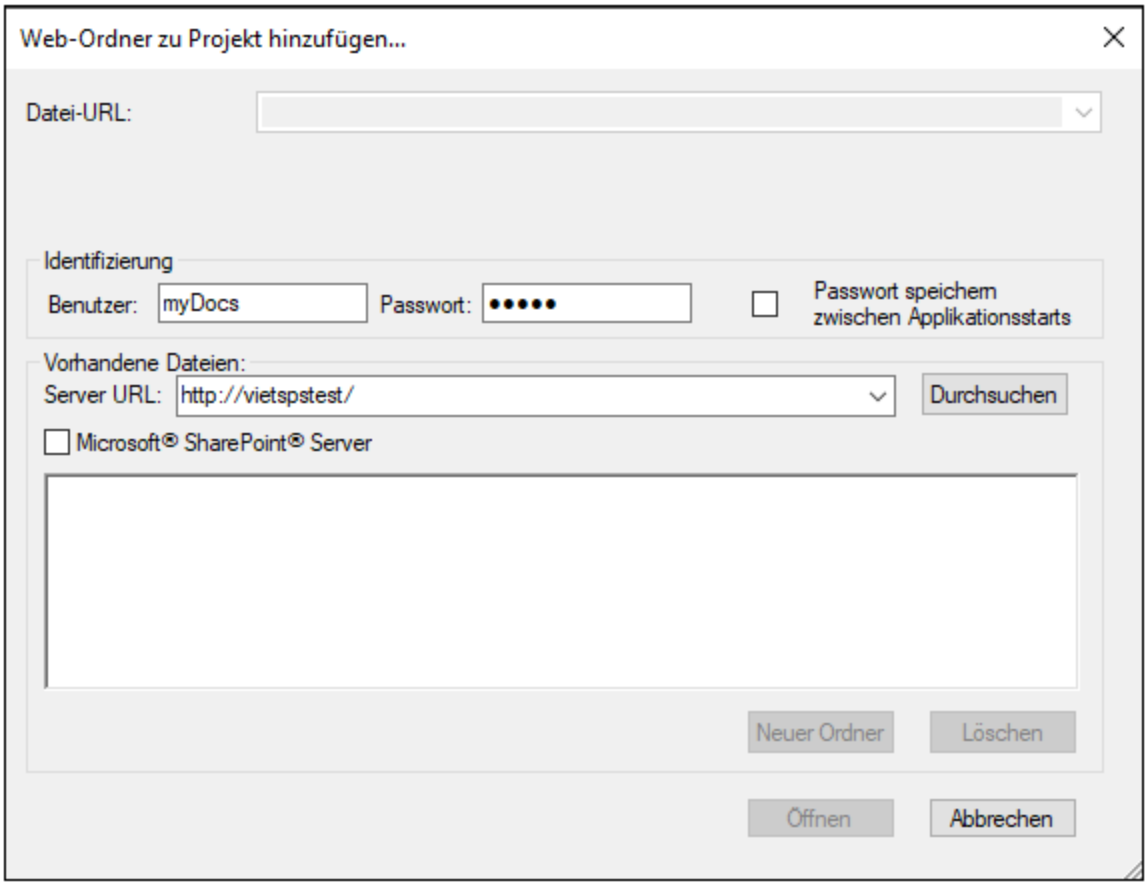
28.3.14 Externes Webverzeichnis zum Projekt hinzufügen

Verwenden Sie diese Funktion, um ein neues Webverzeichnis in das aktuelle Projekt einzufügen. Sie können diesen Befehl auch über das Kontextmenü (Rechtsklick auf ein Verzeichnis im Projektfenster) aufrufen. Bitte beachten Sie, dass Dateien in externen Verzeichnissen nicht unter Versionskontrolle gestellt werden können.

Hinzufügen eines externen Webverzeichnisses zum Projekt

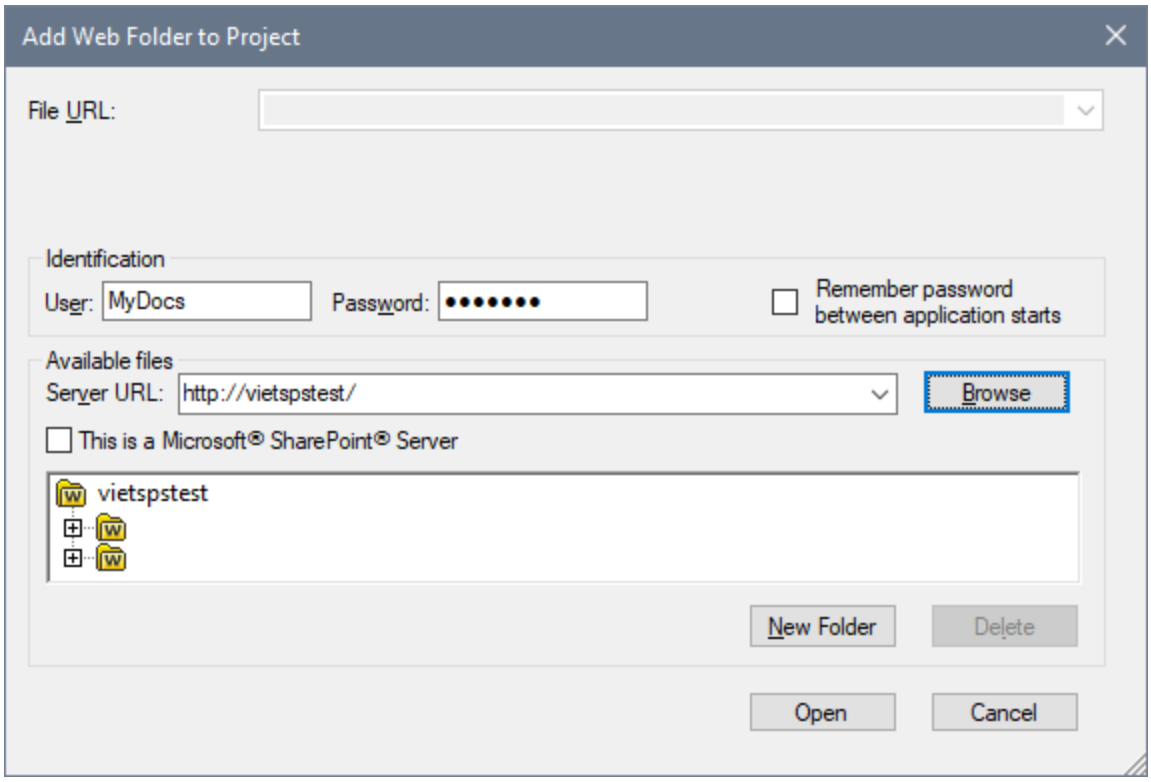
Um ein externes Webverzeichnis zum Projekt hinzuzufügen, gehen Sie folgendermaßen vor:

1. Wählen Sie die Menüoption **Projekt | Externes Webverzeichnis zum Projekt hinzufügen**. Daraufhin wird das Dialogfeld "Webordner in Projekt einfügen" angezeigt (*Abbildung unten*).

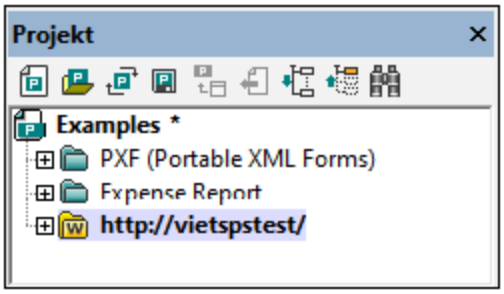


2. Klicken Sie in das Feld "Server URL" und geben Sie die URL des Servers ein. Wenn es sich beim Server um einen Microsoft® SharePoint® Server handelt, aktivieren Sie das Kontrollkästchen *Microsoft® SharePoint® Server*. Nähere Informationen zum Arbeiten mit Dateien dieses Servertyps finden Sie im Abschnitt *Ordner auf einem Microsoft® SharePoint® Server* weiter unten.
3. Geben Sie Ihre Benutzer-ID und Ihr Passwort in die Felder *Benutzer* und *Passwort* ein, wenn der Server passwortgeschützt ist.
4. Klicken Sie auf "**Durchsuchen**" um die Verbindung mit dem Server herzustellen, und die verfügbaren Ordner anzuzeigen.

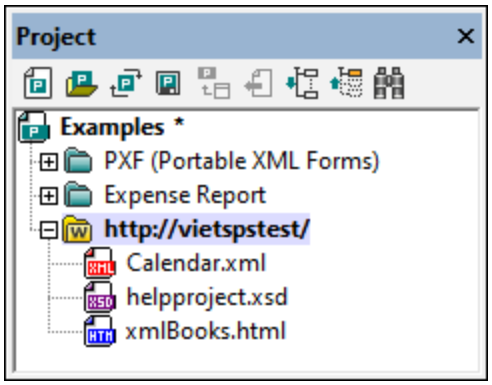
© 2016-2022 Altova GmbH



5. Klicken Sie auf das Verzeichnis, welches Sie in die Projektansicht übernehmen möchten. Die Schaltfläche **Öffnen** wird erst danach aktiv. Die URL des Ordners wird nun im Feld Datei-URL angezeigt.
6. Klicken Sie auf **Öffnen**, um das Verzeichnis in das Projekt zu übernehmen.



7. Klicken Sie auf das Symbol "+" um den Inhalt des Verzeichnisses anzuzeigen.



Filtern von Verzeichnisinhalt

Um den Inhalt eines Ordners zu filtern, rechtsklicken Sie auf den Ordner und wählen Sie im Kontextmenü die Option **Eigenschaften**. Klicken Sie im Dialogfeld "Eigenschaften", das daraufhin geöffnet wird, auf das Feld *Dateierweiterungen* und geben Sie die Erweiterungen der Dateien an, die angezeigt werden sollen (z.B. XML- und XSD-Dateien). Trennen Sie die einzelnen Dateitypen durch ein Semikolon (z.B. `xml; xsd; sps`). Im Projektfenster wird jetzt nur mehr dieser Ordner mit Dateien mit der definierten Erweiterung angezeigt.

Validierung und Überprüfen der Wohlgeformtheit eines Ordners

Um die Dateien in einem Ordner auf Wohlgeformtheit zu prüfen oder sie zu validieren, klicken Sie auf den Ordner und anschließend auf den Menübefehl **XML | Wohlgeformtheit prüfen** bzw. **XML | XML validieren** (Kürzel F7 bzw. F8). Alle angezeigten Dateien im Ordner werden nun überprüft. Falls eine Datei nicht wohlgeformt oder ungültig ist, wird diese im Hauptfenster geöffnet. Hier können Sie den Fehler korrigieren. Nach erfolgter Korrektur, starten Sie die Prüfung erneut, um das restliche Verzeichnis zu kontrollieren. Bitte beachten Sie: Sie können mehrere nicht nebeneinander liegende Dateien gleichzeitig im Verzeichnis markieren, wenn Sie die Strg-Taste gedrückt halten, und die gewünschten Dateien anklicken. In diesem Fall werden beim Betätigen der Funktionstasten **F7** oder **F8** nur die markierten Dateien überprüft.

Aktualisieren des Inhalts des Projektordners

Sie können jederzeit Dateien aus dem Webverzeichnis hinzufügen oder löschen. Um die Verzeichnis-Ansicht zu aktualisieren, rechtsklicken Sie auf den externen Ordner und wählen Sie im Kontextmenü die Option **Aktualisieren**.

Löschen von Dateien oder Verzeichnissen

Da es sich um einen Webordner handelt, der zum Projekt hinzugefügt wurde, kann nur der Webordner (und nicht die Dateien darin) aus dem Projekt gelöscht werden. Um einen Webordner aus dem Projektfenster zu löschen, klicken Sie entweder (i) mit der rechten Maustaste auf den Ordner und wählen Sie den Befehl **Löschen** oder (ii) wählen Sie den Ordner aus und drücken Sie die **Entf**-Taste. Der Ordner wird nur aus der Projekt-Ansicht und nicht vom Webserver entfernt.

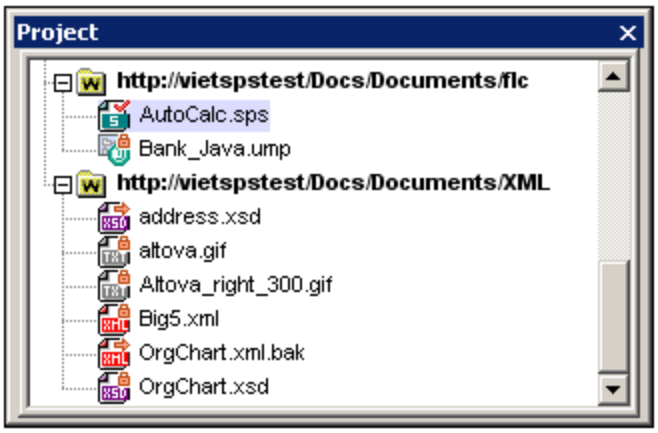
Anmerkung: Durch Klick auf eine **einzelne Datei** und Betätigen der **Entf**-Taste wird die Datei **nicht aus dem Projektfenster gelöscht**. Sie müssen sie physisch löschen, und dann den Inhalt des externen Ordners aktualisieren.

© 2016-2022 Altova GmbH

Ordner auf einem Microsoft® SharePoint® Server

Wenn ein Ordner auf einem Microsoft® SharePoint® Server zum Projekt hinzugefügt wurde, können Dateien im Ordner über Befehle im Kontextmenü zur Dateiliste im Projektfenster (*siehe Abbildung unten*) ein- und ausgecheckt werden. Um diese Befehle aufzurufen, klicken Sie mit der rechten Maustaste auf die gewünschte Datei und wählen Sie den entsprechenden Befehl aus (**Auschecken**, **Einchecken**, **Auschecken rückgängig**).

Die Benutzer-ID und das Passwort können in den Eigenschaften der einzelnen Ordner im Projekt[1344] gespeichert werden, sodass Sie diese Zugriffsdaten nicht jedes Mal eingeben müssen, wenn Sie auf den Server zugreifen.



Die Dateien im Projektfenster (*Abbildung unten*) sind mit Symbolen versehen, die den Status der Dateien (Eingecheckt/Ausgecheckt) kennzeichnen. Hier sehen Sie die verschiedenen Dateisymbole:

	Eingecheckt. Die Datei kann ausgecheckt werden.
	Die Datei wurde von einem anderen Benutzer ausgecheckt und kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Die Datei kann bearbeitet und eingecheckt werden.

Beachten Sie bitte die folgenden Punkte:

- Nachdem Sie eine Datei ausgecheckt haben, können Sie sie in Ihrer Altova-Applikation bearbeiten und mit **Datei | Speichern (Strg+S)** speichern.
- Sie können die bearbeitete Datei über das Kontextmenü im Projektfenster (*siehe Abbildung oben*) oder über das Kontextmenü, das angezeigt wird, wenn Sie mit der rechten Maustaste auf das Dateiregister im Hauptfenster Ihrer Applikation klicken (*Abbildung unten*), einchecken.



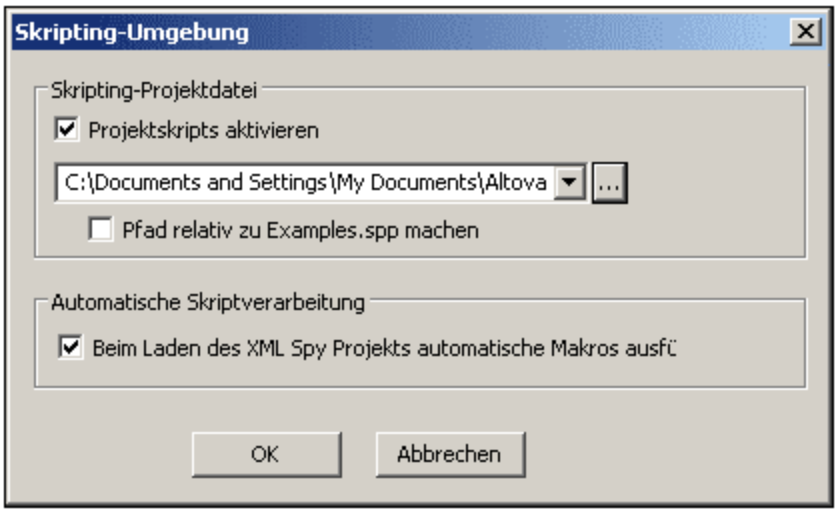
- Wenn eine Datei von einem anderen Benutzer ausgecheckt wurde, kann sie nicht auscheckt werden.

- Wenn eine Datei lokal von Ihnen ausgecheckt wurde, können Sie dies mit dem Kontextmenübefehl "Auschecken rückgängig" rückgängig machen. In diesem Fall wird die Datei dem Server unverändert zurückgegeben.
- Wenn Sie eine Datei in einer Altova-Applikation auschecken, können Sie sie in einer anderen Altova-Applikation nicht auschecken. Die Datei gilt bereits als für Sie ausgecheckt. Daher stehen in diesem Fall in jeder Altova-Applikation, die Microsoft® SharePoint® Server unterstützt, nur die Befehle **Einchecken** und **Auschecken rückgängig** zur Verfügung.

28.3.15 Skript-Einstellungen

Ein Skripting-Projekt wird einem XMLSpy-Projekt auf folgende Weise zugewiesen:

1. Öffnen Sie auf der XMLSpy-Benutzeroberfläche das gewünschte Applikationsprojekt.
2. Wählen Sie den Menübefehl **Projekt | Skript-Einstellungen**. Daraufhin wird das Dialogfeld "Skripting-Umgebung" (*Abbildung unten*) geöffnet.



3. Aktivieren Sie das Kontrollkästchen *Projektskripts aktivieren* und wählen Sie das gewünschte Skripting-Projekt (`.asprj`-Datei) aus. Wenn beim Laden des XMLSpy-Projekts automatische Makros ausgeführt werden sollen, aktivieren Sie das Kontrollkästchen *Automatische Makros ausführen*.
4. Klicken Sie zum Fertigstellen auf **OK**.

Anmerkung: Um das Skripting-Projekt eines XMLSpy-Projekts zu deaktivieren, d.h. die Zuweisung aufzuheben, deaktivieren Sie das Kontrollkästchen *Projektskripts aktivieren*.

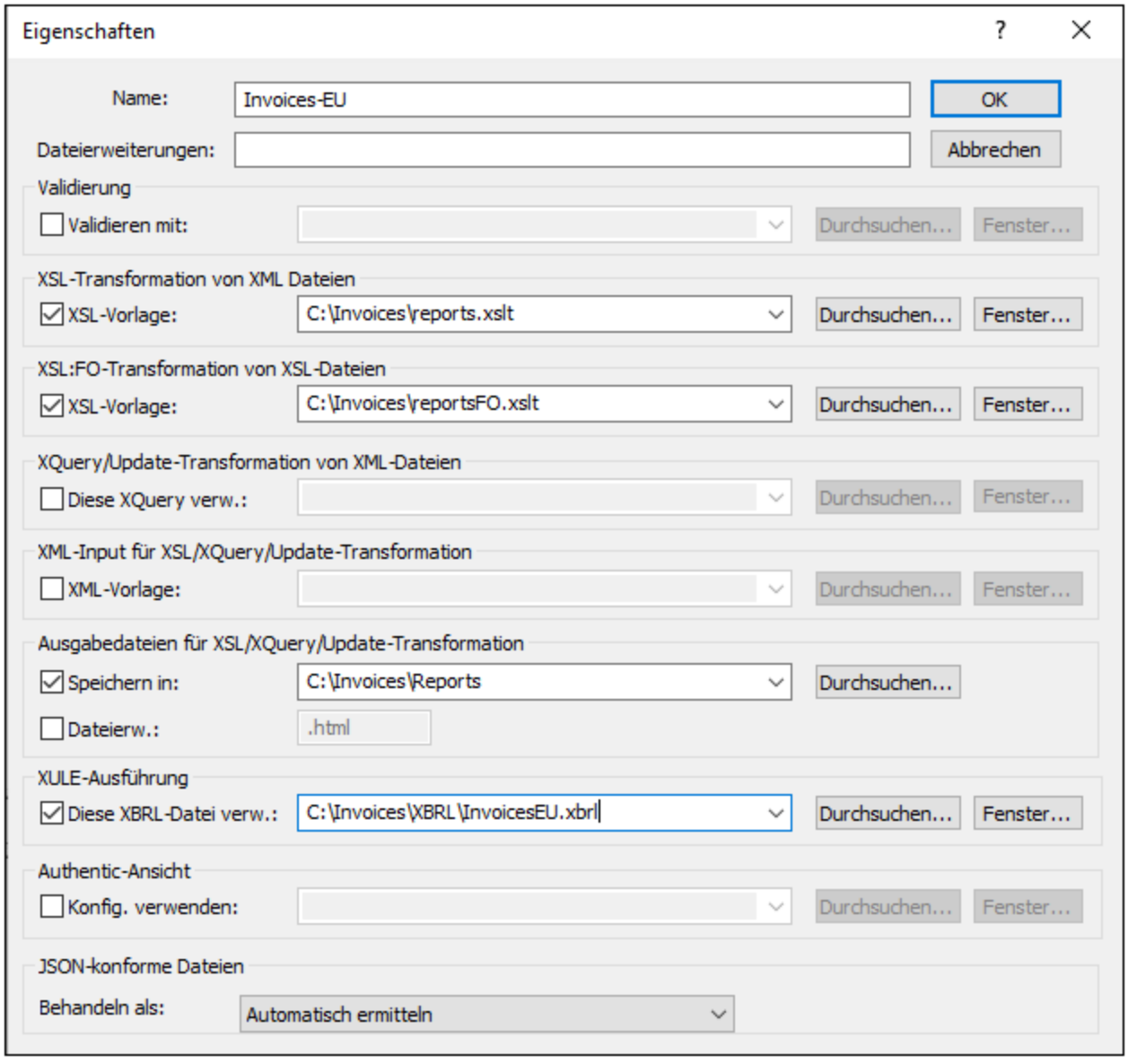
28.3.16 Eigenschaften



© 2016-2022 Altova GmbH

Mit dem Befehl **Projekt | Eigenschaften** wird das Dialogfeld "Eigenschaften" (*Abbildung unten*) des aktiven Projekts geöffnet. Wenn Sie mit der rechten Maustaste im Projektfenster auf einen Ordner klicken (und nicht auf den Projektordner selbst) und den Befehl **Eigenschaften** wählen, wird das Eigenschaftsdialogfeld für diesen Ordner geöffnet. Im Folgenden finden Sie eine Beschreibung der Einstellungen des Dialogfelds.

Anmerkung: Falls die Projektdatei unter Versionskontrolle steht, werden Sie gefragt, ob die Projektdatei (`.spp`) ausgecheckt werden soll. Klicken Sie auf **OK**, wenn Sie Einstellungen bearbeiten und diese speichern möchten.



Einstellungen

Dateierweiterungen
Die Einstellung *Dateierweiterungen* ist nur bei Ordnern, nicht aber für den Projektordner aktiv. Wenn eine Datei zu einem Projekt hinzugefügt wird, wird sie zu dem Ordner hinzugefügt, für den ihre Dateierweiterungen definiert wurden. Wenn z.B. eine Datei namens `MyReport.xml` zum Projekt hinzugefügt wird und die Dateierweiterung `.xml` für den Ordner `Invoices-EU` definiert wurde (siehe Abbildung oben), so wird `MyReport.xml` zum Ordner `Invoices-EU` hinzugefügt. Wenn mehrere Ordner vorhanden sind, zu denen Sie XML-Dateien hinzufügen möchten, so sollten Sie die einzelnen XML-Dateien direkt zum Ordner (und nicht zum Projekt) hinzufügen.

Benutzer-ID und Passwort für externe Ordner
Für externe Ordner (auch externe Webverzeichnisse) können Sie die Benutzer-ID und das Passwort speichern, die unter Umständen für den Zugriff auf den Server benötigt werden.

Validierung
Die DTD, das XML-Schema oder das JSON-Schema 765, die/das für die Validierung 1353 der Dateien im aktuellen Ordner (oder im gesamten Projekt, wenn es sich um die Eigenschaften des Projekts handelt) verwendet wird.

XSL-Transformation von XML-Dateien
Das XSLT-Stylesheet, das für die XSLT-Transformation 1411 von XML-Dateien im Ordner verwendet wird.

XSL-FO-Transformation von XML-Dateien
Das XSLT-Stylesheet, das für die Transformation von XML-Dateien in XSL-FO in diesem Ordner verwendet wird.

XQuery/Update-Transformation von XML-Dateien
Die XQuery- oder XQuery Update-Datei, die in diesem Ordner für XQuery- oder XQuery Update-Ausführungen von XML-Dateien verwendet wird.

XML-Input für XSL/XQuery-Update-Transformation von XML-Dateien
Die XML-Datei, die in diesem Ordner für die XSLT-Transformation oder die XQuery/XQuery Update-Ausführung mit der entsprechenden XSLT-, XQuery- oder XQuery Update-Datei in diesem Ordner verwendet wird.

Ausgabedateien für XSL/XQuery/Update-Transformation
Das Zielverzeichnis von Transformationen und optional dazu die Dateierweiterung des Ergebnisdokuments.

XULE-Ausführung
Die XBRL-Instanzdatei, die mit dem XULE-Dokument, das im XMLSpy-Applikationsfenster aktiv ist, verarbeitet werden soll.

Authentic-Ansicht
Mit der Option *Konfig. verw.* wird die StyleVision Power Stylesheet-Datei (SPS-Datei) definiert, die für die Anzeige von XML-Dateien aus diesem Ordner in der Authentic verwendet werden soll. Beachten Sie, dass die XML-Arbeitsdatei gemäß dem für das SPS ausgewählten Schema gültig sein muss.

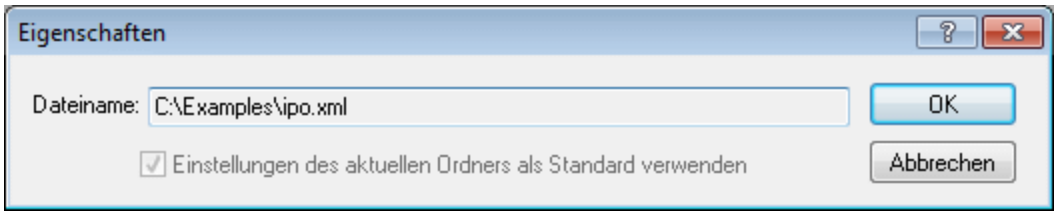
© 2016-2022 Altova GmbH

JSON-konforme Dateien
Mit dieser Eigenschaft wird definiert, ob ein Projektordner JSON-Schema-Dateien oder JSON-Instanzdateien enthält. Sie ist sehr nützlich, wenn es darum geht, JSON-Schema-Dateien zu erkennen, wenn die Dateien nicht durch das Schlüsselwort `$schema` eindeutig als JSON-Schema-Dateien gekennzeichnet sind und die Dateien einander referenzieren. Sie können diese Eigenschaft auf *JSON-Instanz, JSON-Schema* oder *Automatisch ermitteln* setzen. Mit der Standardeinstellung *Automatisch ermitteln* würde XMLSpy den Typ der Datei anhand der Struktur und des Inhalts von JSON-Dateien ermitteln.

Anmerkungen zu Projekteigenschaften

Beachten Sie hinsichtlich Vorrangigkeit bitte die folgenden Punkte:

- Wenn die Validierung oder XSLT/XQuery-Transformation über Kontextmenüs des Projektordners ausgeführt wird, so haben die in diesem Dialogfeld definierten Validierungs- bzw. Transformationsdateien Vorrang vor in der XML-Datei getätigten Zuweisungen. Außerdem haben für den jeweiligen Projektordner getroffene Einstellungen Vorrang vor Einstellungen, die für übergeordnete Ordner getroffen wurden.
- Wenn eine Datei in mehreren Ordnern des Projekts vorhanden ist und ihr in den verschiedenen Ordnern unterschiedliche Validierungs- und Transformationsdateien zugewiesen wurden, so können Sie auswählen, welche Zuweisung verwendet werden soll, wenn die Datei außerhalb des Projekts verarbeitet wird. Gehen Sie dazu folgendermaßen vor: Navigieren Sie in dem Projektordner, dessen Zuweisung(en) Sie verwenden möchten, zur gewünschten Datei. Klicken Sie mit der rechten Maustaste auf die Datei in diesem Projektordner und wählen Sie den Befehl **Eigenschaften.** Aktivieren Sie im daraufhin angezeigten Dialogfeld (*Abbildung unten*) die Option *Einstellungen des aktuellen Ordners als Standard verwenden*. (Der aktuelle Ordner ist der Projektordner, in dem sich die Datei befindet.) Wenn die Option deaktiviert ist, bedeutet dies, dass die Einstellungen des aktuellen Ordners bereits als Standardeinstellungen ausgewählt sind. Wenn Sie eine Dateiinstanz auswählen, die sich in einem Projektordner befindet, der nicht der Standardordner ist, so steht die Option zur Verfügung und Sie können die Standardeinstellungen in die Einstellungen dieses Ordners ändern. Beachten Sie, dass die lokale Zuweisung (d.h. eine Zuweisung innerhalb der Datei selbst) verwendet wird, falls die Datei eine lokale Zuweisung hat. In diesem Fall werden die Einstellungen des Standardordners ignoriert.



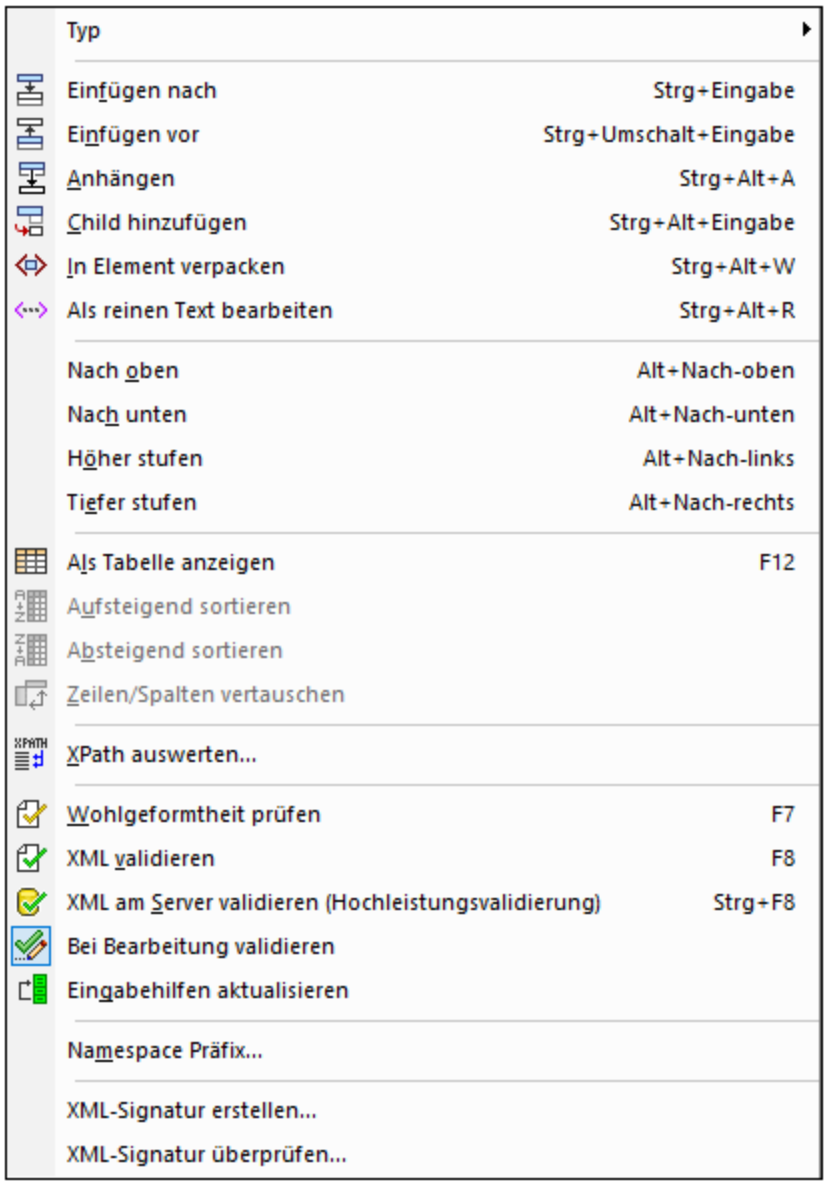
28.3.17 Zuletzt verwendete Projekte

Mit dieser Funktion werden die neun zuletzt verwendeten Projekte angezeigt um einen schnelleren Zugriff auf diese zu erhalten.

Beachten Sie bitte, dass XMLSpy automatisch das zuletzt verwendete Projekt[1603] beim Starten von XMLSpy öffnen kann. (**Extras | Optionen|** Register **Datei**, Projekt | Letztes Projekt beim Programmstart öffnen).

28.4 Menü "XML"

Das XML-Menü enthält alle nötigen Befehle um mit XML-Dokumenten zu arbeiten. Sie können Elemente einfügen oder anhängen, Elementhierarchien ändern, ein Namespace Präfix für Objekte setzen und XPaths im Zusammenhang mit einzelnen XML-Dokumenten evaluieren.



Zu den am häufigsten verwendeten XML-Funktionen zählen die Überprüfung von Dokumenten auf ihre Wohlgeformtheit [1351] und Gültigkeit [1353]. Sie finden diese Befehle in diesem Menü.

© 2016-2022 Altova GmbH

28.4.1 Typ

Der Befehl **Typ** hat ein Untermenü mit einer Liste von XML-Node-Typen. Sie können den Typ des aktuell in der Grid-Ansicht ausgewählten Node in einen anderen Typ aus der Liste ändern. Die Node-Typen in diesem Untermenü werden nur in der Grid-Ansicht aktiv und es sind nur diejenigen Node-Typen aktiv, in die der aktuell ausgewählte Node geändert werden kann.

Nähere Informationen dazu finden Sie im Abschnitt XML | Dokumentinhalt 305.

28.4.2 Einfügen nach/vor

Die Befehle **Einfügen nach** und **Einfügen vor** sind aktiviert, wenn in der Grid-Ansicht ein Node ausgewählt ist. Sie fügen damit einen Element-Node auf derselben Ebene vor bzw. nach dem ausgewählten Eintrag hinzu. Durch Doppelklick auf die Namenszelle und Bearbeitung dieser können Sie den Namen des neu hinzugefügten Elements ändern. Durch Klick auf das Symbol des Elements (links von seinem Namen) und Auswahl des gewünschten Node-Typs oder mit dem Befehl **XML | Typ** 1349 können Sie den Node-Typ ändern.

Nähere Informationen dazu finden Sie im Abschnitt XML | Dokumentstruktur 302.

28.4.3 Anhängen, Child hinzufügen

Die Befehle **Anhängen** und **Child hinzufügen** sind aktiviert, wenn in der Grid-Ansicht ein Node ausgewählt ist. Mit dem Befehl **Anhängen** wird in neuer Element-Node als letztes gleichrangiges Element des ausgewählten Eintrags hinzugefügt. Mit dem Befehl **Child hinzufügen** wird ein neuer Element-Node als Child angehängt. Durch Doppelklick auf die Namenszelle und Bearbeitung dieser können Sie den Namen des neu hinzugefügten Elements ändern. Durch Klick auf das Symbol des Elements (links von seinem Namen) und Auswahl des gewünschten Node-Typs oder mit dem Befehl **XML | Typ** 1349 können Sie den Node-Typ ändern.

Nähere Informationen dazu finden Sie im Abschnitt XML | Dokumentstruktur 302.

28.4.4 In Element verpacken

Der Befehl **In Element verpacken** ist aktiviert, wenn in der Grid-Ansicht ein Node ausgewählt ist. Der ausgewählte Node erhält ein Parent-Element mit einem Standardnamen. Durch Doppelklick auf die Namenszelle und Bearbeitung dieser können Sie den Namen des neu hinzugefügten Parent-Elements ändern.

Nähere Informationen dazu finden Sie im Abschnitt XML | Dokumentstruktur 302.

28.4.5 Als reinen Text bearbeiten

Der Befehl **Als reinen Text bearbeiten** ist aktiviert, wenn in der Grid-Ansicht ein Node ausgewählt ist. Damit können Sie Textinhalt des ausgewählten Eintrags als reinen Text bearbeiten. Dies ist z.B. dann nützlich, wenn Sie komplexen Inhalt wie z.B. HTML-Code bearbeiten.

Nähere Informationen finden Sie im Abschnitt XML in der Grid-Ansicht 296.

28.4.6 Nach oben/nach unten/höher stufen/tiefer stufen

Diese Befehle zum **Verschieben** sind aktiviert, wenn in der Grid-Ansicht ein Node ausgewählt ist. Ein Node kann im Raster nach oben, unten, links oder rechts verschoben werden. In diesem Fall sind die entsprechende Befehle aktiv. Wählen Sie den entsprechenden Befehl aus, um die Komponente zu verschieben.

28.4.7 Als Tabelle anzeigen

Der Befehl **Als Tabelle anzeigen** ist aktiviert, wenn in der Grid-Ansicht ein sich wiederholendes Element ausgewählt ist. Dies ist ein Ein/Aus-Befehl, mit dem Sie zwischen der Anzeige sich wiederholender Elemente in der Standard-Grid-Ansicht 169 und der Tabellenansicht 312 wechseln können. In der Tabellenansicht können sich wiederholende Elemente als Tabelle, in der Zeilen die Instanzen und Spalten die Child-Nodes abbilden, angezeigt werden.

Nähere Informationen dazu finden Sie im Abschnitt Tabellenansicht 312.

28.4.8 Aufsteigend/Absteigend sortieren

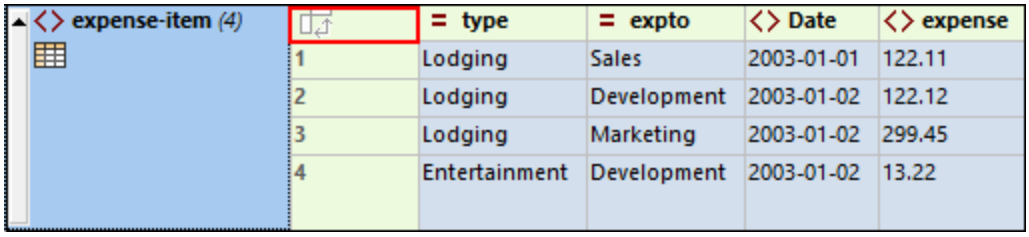
Die Befehle **Aufsteigend sortieren** und **Absteigend sortieren** sind in der Tabellenansicht 312 aktiviert, wenn eine Spalte oder Zelle ausgewählt ist. Sie sortieren die ausgewählte Spalte je nach Datentyp der Spalte entweder in alphabetischer oder numerischer aufsteigender/absteigender Reihenfolge.

Nähere Informationen dazu finden Sie im Abschnitt Tabellenansicht 312.

28.4.9 Zeilen/Spalten vertauschen

Der Befehl **Zeilen/Spalten vertauschen** ist in der Tabellenansicht 312 aktiviert, wenn die oberste linke Zelle einer Tabelle ausgewählt ist (*in der Abbildung unten rot markiert*). Mit dem Befehl werden Zeilen als Spalten angezeigt bzw. Spalten als Zeilen.

© 2016-2022 Altova GmbH



expense-item (4)	type	expto	Date	expense
1	Lodging	Sales	2003-01-01	122.11
2	Lodging	Development	2003-01-02	122.12
3	Lodging	Marketing	2003-01-02	299.45
4	Entertainment	Development	2003-01-02	13.22

Nähere Informationen dazu finden Sie im Abschnitt Tabellenansicht[312].

28.4.10 XPath auswerten



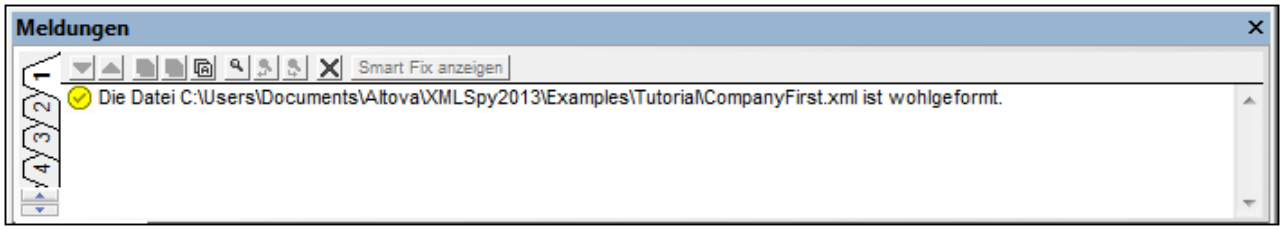
Bei Auswahl des Befehls **XML** | **XPath auswerten** werden die Ausgabefenster geöffnet, wenn Sie noch nicht offen sind und das Register "XPath" wird in den Ausgabefenstern aktiviert[124]. im Abschnitt "XPath" können Sie einen XPath-Ausdruck im aktiven Dokument auswerten und das Ergebnis im Fenster "Ausgabe" anzeigen.

28.4.11 Wohlgeformtheit prüfen

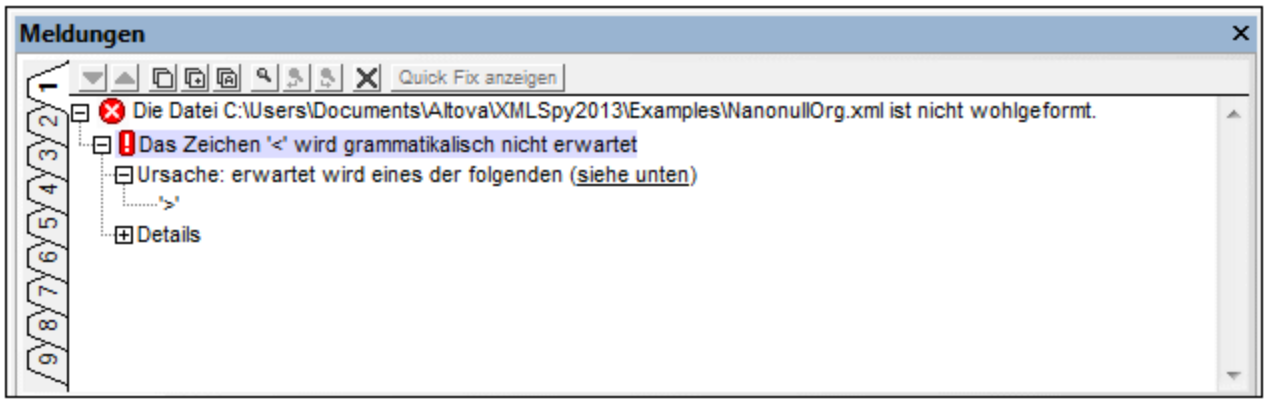
 **F7**

Mit diesem Befehl überprüfen Sie das Dokument auf Wohlgeformtheit nach der Definition der XML 1.0 Spezifikation. Jedes XML-Dokument muss wohlgeformt sein; deswegen überprüft XMLSpy ein Dokument automatisch auf Wohlgeformtheit, sobald es geöffnet oder gespeichert wird, oder wenn eine andere als die Text-Ansicht ausgewählt wird. Sie können das Dokument auch während der Bearbeitung jederzeit auf Wohlgeformtheit prüfen.

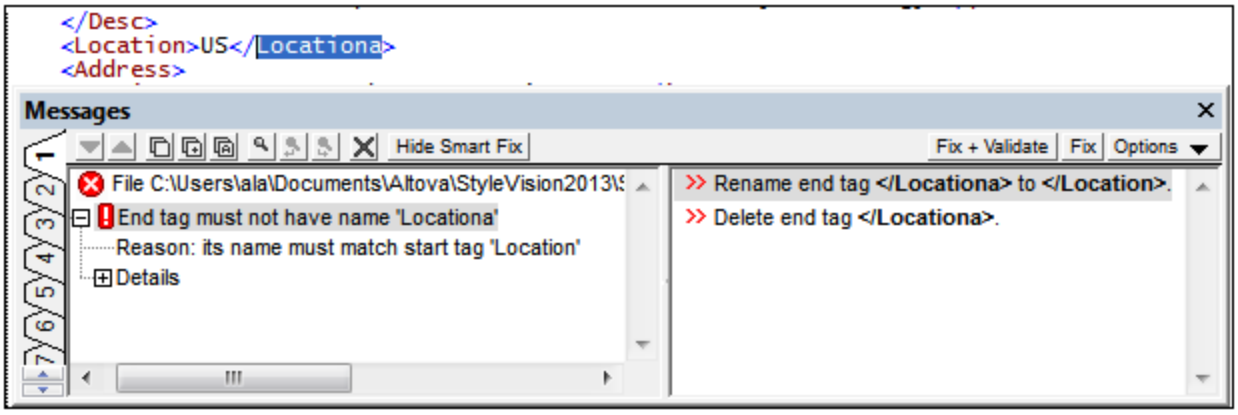
Bei erfolgreicher Wohlgeformtheitsprüfung wird am unteren Rand des Hauptfensters eine Meldung angezeigt (*Abbildung unten*).



Wenn bei der Wohlgeformtheitsprüfung ein Fehler entdeckt wurde, wird das Fehler verursachende Objekt markiert und eine entsprechende Fehlermeldung am unteren Rand des Hauptfensters angezeigt (*Abbildung unten*).



Auf Basis der im Schema enthaltenen Informationen werden bei Ausführung der Validierung in der **Text**- oder **Grid-Ansicht** Smart Fix-Optionen vorgeschlagen. Um eine Liste von Smart Fix-Optionen anzuzeigen, klicken Sie auf die Schaltfläche **Smart Fix anzeigen** (*siehe Abbildung oben*). Daraufhin wird im Fenster "Meldungen" ein Fenster mit den vorgeschlagenen Smart Fix-Optionen angezeigt (*Abbildung unten*). Beachten Sie, dass immer nur ein Fehler im Fenster "Meldungen" angezeigt wird.



Wählen Sie im Fenster "Smart Fix" einen der vorgeschlagenen Smart Fixes aus und klicken Sie entweder auf die Schaltfläche **Beheben + validieren** oder auf die Schaltfläche **Beheben** (*siehe Abbildung oben*). Der ungültige Abschnitt im XML-Dokument wird daraufhin durch den ausgewählten Smart Fix ersetzt. Alternativ dazu können Sie auch auf den gewünschten Smart Fix-Vorschlag doppelklicken. Je nachdem, welche Option in der Dropdown-Liste *Optionen* ausgewählt ist, wird der Fehler damit behoben oder der Fehler wird behoben und das Dokument validiert. Der Befehl **Beheben + validieren** ist nützlich, da die Validierung auch hinter dem behobenen Fehler fortgesetzt werden und der nächste Fehler gefunden und behoben werden kann.

Um das Smart Fix-Fenster auszublenden, klicken Sie auf die Schaltfläche **Smart Fix ausblenden** (*siehe Abbildung oben*).

Anmerkung: Das Fenster "Meldungen" enthält neun Ausgaberegister. Das Ergebnis der Validierung wird immer auf dem aktuellen Register angezeigt. Sie können ein XML-Dokuments auf Register 1 validieren und das Ergebnis auf diesem Register aufbewahren, Um ein zweites Dokument zu validieren, wechseln Sie zu Register 2 (oder Register 3, wenn Sie möchten), bevor Sie die Validierung durchführen. Wenn Sie die Register nicht wechseln, wird Register 1 (bzw. das aktive Register) durch das Validierungsergebnis überschrieben.

© 2016-2022 Altova GmbH

Validierung über das Projektfenster

Der Befehl **Validieren** kann auch auf eine Datei, einen Ordner oder eine Gruppe von Dateien im aktiven Projekt angewendet werden. Klicken Sie im Projektfenster auf die gewünschte Datei/den gewünschten Ordner und wählen Sie den Befehl **XML | XML validieren.** Sie können auch den Befehl **XML am Server validieren (Hochleistungsvalidierung)**[1357] verwenden**.** Ungültige Dateien werden als aktive Datei im Hauptfenster geöffnet und die entsprechende Fehlermeldung wird angezeigt.

Anmerkung: Das Fenster "Meldungen" enthält neun Register. Das Ergebnis der Wohlgeformtheitsprüfung wird immer auf dem aktuellen Register angezeigt. Sie können daher die Wohlgeformtheit eines XML-Dokuments auf Register 1 überprüfen und das Ergebnis auf diesem Register aufbewahren, Um die Wohlgeformtheit eines zweiten Dokuments zu überprüfen, wechseln Sie zu Register 2 (oder Register 3, wenn Sie möchten), bevor Sie die Wohlgeformtheitsprüfung durchführen. Wenn Sie die Register nicht wechseln, wird Register 1 (bzw. das aktive Register) durch das Ergebnis der letzten Überprüfung überschrieben.

Eigentlich ist es nicht erlaubt, ein schlecht geformtes XML-Dokument zu speichern; XMLSpy enthält die Funktion "trotzdem speichern", die angezeigt wird, wenn die automatische Wohlgeformtheitsprüfung beim Speichern des Dokument einen Fehler entdeckt. Dies ist nützlich, wenn sie Ihre Arbeit kurzzeitig unterbrechen müssen, und den Fehler nicht sofort beheben können.

Anmerkung: Sie können den Befehl **Wohlgeformtheit prüfen** auch an jeder Datei, jedem Ordner oder jeder Gruppe von Dateien im aktiven Projektfenster[118] anwenden. Klicken Sie auf den entsprechenden Eintrag, und anschließend auf das Symbol "Wohlgeformtheit prüfen".

28.4.12 XML validieren

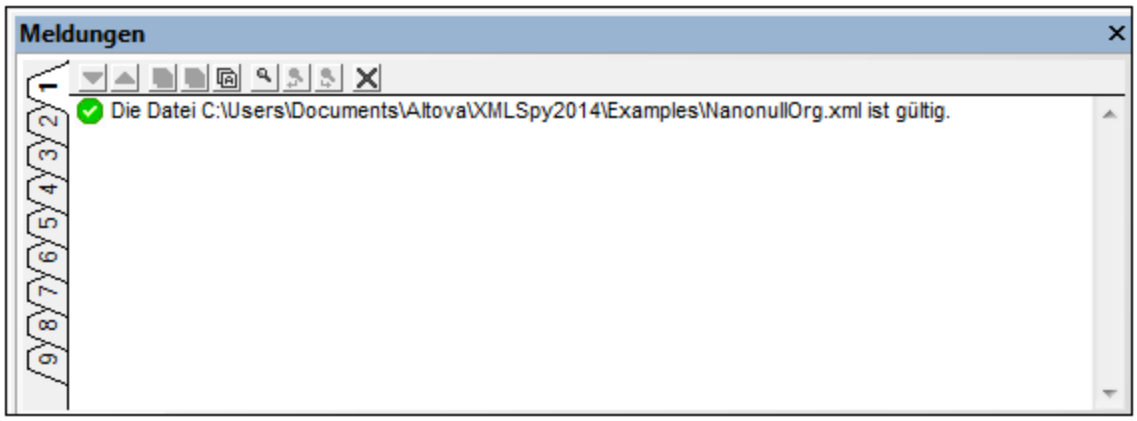


F8

Mit dem Befehl **XML | Validieren (F8)** können Sie ein XML-Dokument anhand von DTDs, XML-Schemas und anderen Schemas validieren. Die Validierung wird automatisch durchgeführt, wenn Sie von der Textansicht in eine andere Ansicht wechseln. Sie können einstellen, dass ein Dokument beim Öffnen oder Speichern automatisch validiert wird (**Extras | Optionen | Datei**). Der Befehl **Validieren** führt vor Überprüfung der Gültigkeit gleichzeitig eine Wohlgeformtheitsprüfung durch, sodass diese vorher nicht separat mit dem Befehl **Wohlgeformtheit prüfen**[1351] überprüft werden muss.

Anmerkung: Um Ihre Daten während der Bearbeitung zu validieren, können Sie auch den Befehl **Bei Bearbeitung validieren**[1359] aktivieren.

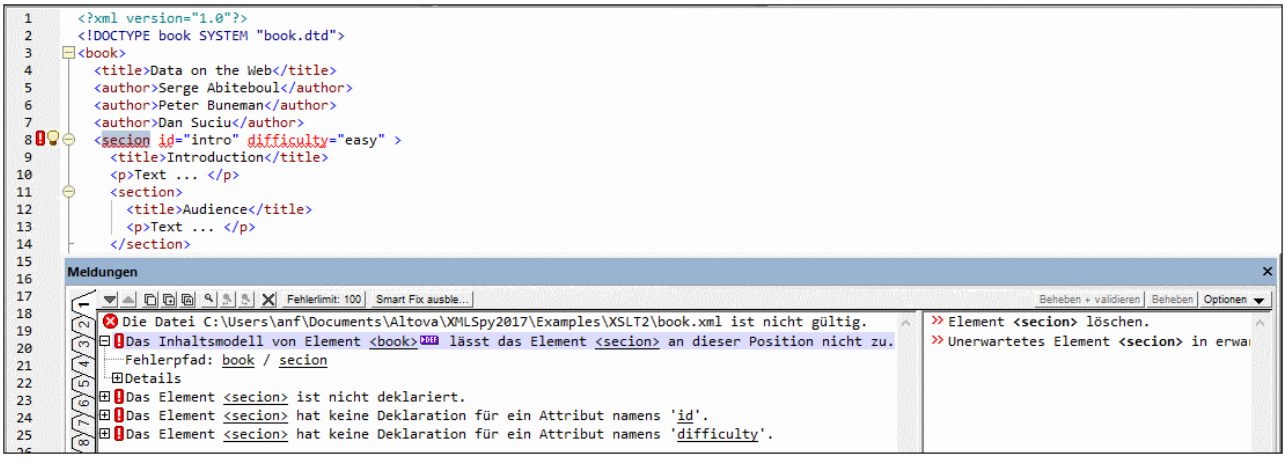
Bei erfolgreicher Validierung wird im Fenster "Meldungen" eine entsprechende Meldung angezeigt:



Andernfalls wird eine Meldung, die den Fehler beschreibt, angezeigt. Wenn Sie auf die Links in der Fehlermeldung klicken, können Sie zu der Stelle im XML-Dokument springen, an der der Fehler gefunden wurde. Eine Beschreibung der Fehlermeldung und deren Behebung mit Hilfe der Smart Fixes von XMLSpy finden Sie im nächsten Abschnitt unten.

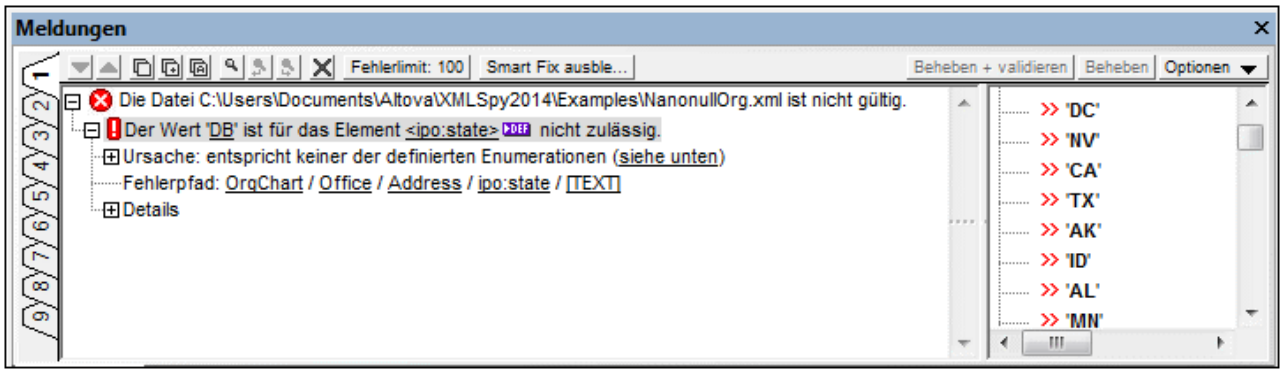
Validierungsfehler und deren Behebung

Wenn im Fenster "Meldungen" ein Validierungsfehler angezeigt wird, werden die Fehlerursachen im Bereich auf der linken Seite angezeigt (*siehe Abbildung unten*). Wenn im linken Bereich eine Fehlerursache ausgewählt wird, so werden im rechten Bereich Smart Fix-Vorschläge dazu angezeigt, falls vorhanden. Smart Fix-Vorschläge stehen in der **Text**- und der **Grid-Ansicht** zur Verfügung und basieren auf den Informationen im damit verknüpften Schema. Um Smart Fix-Vorschläge anzuzeigen, klicken Sie auf die Schaltfläche **Smart Fix anzeigen**. Klicken Sie auf die Schaltfläche **Smart Fix ausblenden,** wenn diese Vorschläge nicht angezeigt werden sollen. Beachten Sie, dass Wohlgeformtheitsfehler (wie nicht übereinstimmende Start- und End-Tags), falls vorhanden, angezeigt werden, bevor Validierungsfehler angezeigt werden. Die Schaltfläche **Smart Fix anzeigen/ausblenden** ist daher nur aktiv, wenn ein Validierungsfehler erreicht wird (d.h. nachdem alle Wohlgeformtheitsfehler korrigiert wurden).

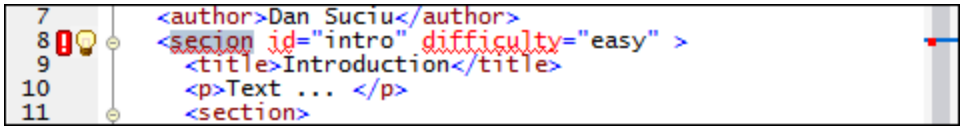


Um einen Smart Fix-Vorschlag anzuwenden, (i) doppelklicken Sie entweder darauf oder (ii) wählen Sie ihn aus und klicken Sie entweder auf die Option **Beheben** oder **Beheben + validieren** (*siehe Abbildung unten*). Mit dem Befehl **Beheben + validieren** wird das Dokument über den behobenen Fehler hinaus validiert und der nächste Fehler wird, falls vorhanden, angezeigt.

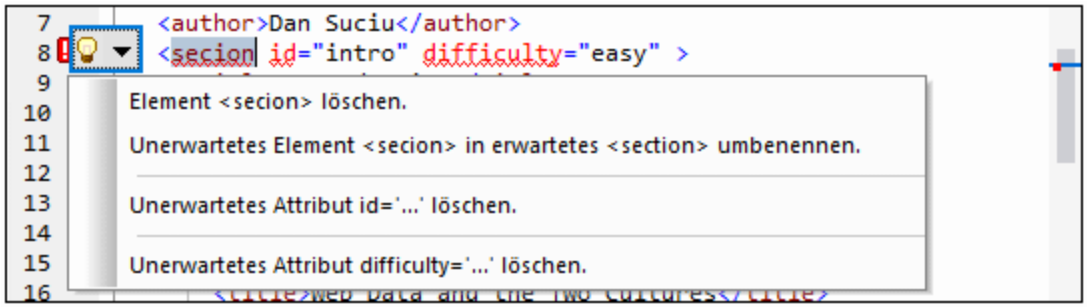
© 2016-2022 Altova GmbH



In der Textansicht finden Sie zwei weitere Markierungen für einen Validierungsfehler (*siehe Abbildung unten*): (i) ein rotes Ausrufezeichen in der Zeilennummernleiste und (ii) ein rotes Markierungsquadrat in der Bildlaufleiste (auf der rechten Seite des Fensters).



Das Glühbirnensymbol neben dem Ausrufezeichensymbol (*siehe Abbildung oben*) ist das Smart Fix-Symbol. Wenn Sie den Mauszeiger darüber platzieren, werden alle Smart Fixes für alle Fehlerursachen angezeigt (*siehe Abbildung unten*). Wählen Sie einen Smart Fix-Vorschlag aus, um diesen anzuwenden.



Anmerkung: Die oben beschriebenen Validierungsfehlermarkierungen und Smart Fix-Vorschläge, werden erst bei Ausführung des Befehls **XML** | **Validieren (F8)** aktualisiert; sie werden nicht im Hintergrund aktualisiert. Sie müssen den Befehl **Validieren (F8)** erneut ausführen, um zu überprüfen, ob der Fehler wirklich behoben wurde.

Anmerkung: Das Fenster "Meldungen" enthält neun Register. Das Ergebnis der Validierung wird immer auf dem aktuellen Register angezeigt. Sie können ein XML-Dokuments auf Register 1 validieren und das Ergebnis auf diesem Register aufbewahren. Um ein zweites Dokument zu validieren, wechseln Sie zu Register 2 (oder Register 3, wenn Sie möchten), bevor Sie die Validierung durchführen. Wenn Sie die Register nicht wechseln, wird Register 1 (bzw. das aktive Register) durch das Ergebnis der neuesten Validierung überschrieben.

Validierung über das Projektfenster

Der Befehl **Validieren** kann auch auf eine Datei, einen Ordner oder eine Gruppe von Dateien im aktiven Projekt angewendet werden. Wählen Sie die gewünschte Datei bzw. den gewünschten Ordner im Projekt-Fenster (durch Anklicken) aus und klicken Sie auf **XML | Validieren** oder **F8**. Ungültige Dateien werden im Projekt als aktive Datei im Hauptfenster geöffnet und die entsprechende Fehlermeldung wird angezeigt.

Validieren von XML-Dokumenten

Um eine XML-Datei zu validieren, machen Sie das XML-Dokument im Hauptfenster zum aktiven und klicken Sie auf **XML | Validieren** oder drücken Sie **F8**. Das XML-Dokument wird daraufhin anhand des in der XML-Datei referenzierten Schemas validiert. Existiert keine Referenz, wird im Fenster "Meldungen" eine Fehlermeldung angezeigt. Solange das XML-Dokument geöffnet ist, bleibt das Schema im Speicher (siehe Cache-Speicher leeren 1385 im DTD/Schema-Menü).

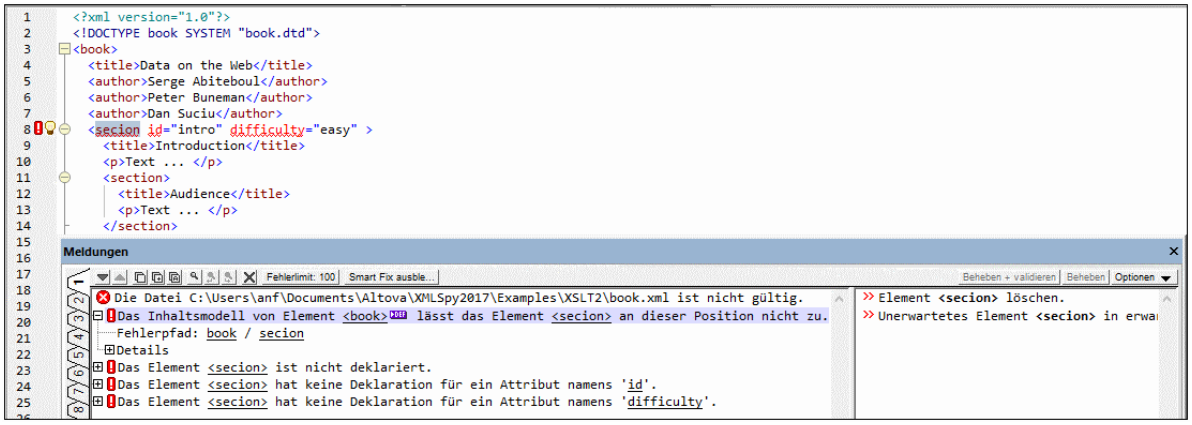
Validieren von Schema-Dokumenten (DTDs und XML-Schema)

XMLSpy unterstützt die gebräuchlichen Schemadialekte einschließlich DTD und XML Schema. Um ein Schema-Dokument zu validieren, machen Sie es im Hauptfenster aktiv und klicken Sie auf **XML | Validieren** oder drücken Sie **F8**.

Validierungsmeldungen

Es gibt zwei Arten von Meldungen:

- Wenn das Schema (DTD oder XML-Schema) gültig ist, wird im Fenster "Meldungen" eine entsprechende Meldung angezeigt.
- Wenn das Schema nicht gültig ist, wird im Fenster "Meldungen" eine Fehlermeldung angezeigt (*Abbildung unten*).



In einer Fehlermeldung wird jede mögliche Ursache für den Fehler separat angezeigt. So werden z.B. in der Abbildung oben vier mögliche Ursachen für den Validierungsfehler angezeigt; die erste ist erweitert, die anderen drei reduziert. Jede Ursache ist in drei Abschnitte gegliedert:

1. Eine Beschreibung der möglichen Ursache. Die Beschreibung enthält Links zur entsprechenden Definition im verknüpften Schemadokument. Sie können die jeweilige Schemadefinition schnell aufrufen, um genau zu sehen, warum das Dokument ungültig ist.

© 2016-2022 Altova GmbH

2. Der Pfad zum fehlerverursachenden Node im XML-Dokument. Wenn Sie in diesem Pfad auf einen Node klicken, wird der Node im Dokument markiert.
3. Detaillierte Informationen über den Fehler sowie ein Link zum entsprechenden Absatz in der Schema-Spezifikation. Hier sind die jeweiligen Schemaregeln zu diesem Punkt definiert.

Anmerkung: Wenn Sie bei Ausführung der Validierung in der Text-Ansicht im Fenster "Meldungen" auf einen Link klicken, wird die entsprechende Definition in der Text-Ansicht *markiert*. Wenn die Validierung in der Schema-Ansicht durchgeführt wird, wird die entsprechende Definition bei Anklicken des Definitions-Links *geöffnet* und Sie *können die Komponente direkt bearbeiten*.

Kataloge

Informationen zur Unterstützung von Katalogen in XMLSpy finden Sie im Abschnitt Kataloge in XMLSpy 463.

Automatisches Validieren mit RaptorXML 2023

Altova RaptorXML ist eine eigenständige Altova-Applikation zur XML-Validierung, XSLT-Transformation und XQuery-Ausführung. Sie kann über die Befehlszeile, über eine COM-Schnittstelle, in Java-Programmen und in .NET-Applikationen aufgerufen werden. So können mit RaptorXML Validierungsaufgaben automatisiert werden. Sie können z.B. eine Batch-Datei erstellen, die RaptorXML aufruft, um damit eine Reihe von Dokumenten zu validieren und die Ausgabe an eine Textdatei zu senden. Nähere Informationen dazu finden Sie in der Dokumentation zu RaptorXML.

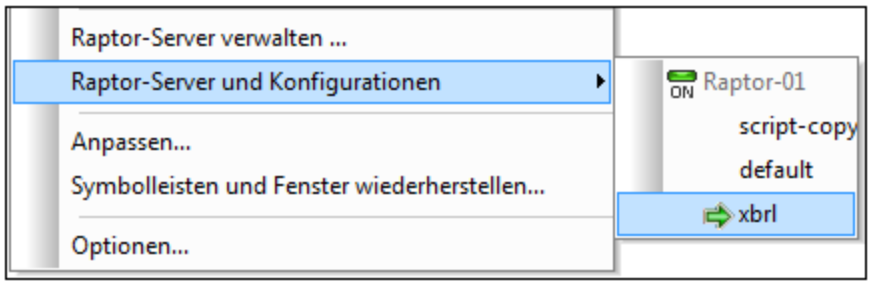
28.4.13 XML am Server validieren (Hochleistungsvalidierung)

 **Strg+F8**

Mit dem Befehl **XML | Am Server validieren (Hochleistungsvalidierung) (Strg+F8)** wird das aktive XML-Dokument unter Verwendung des derzeit aktiven RaptorXML-Servers 1583 und seiner aktiven Konfiguration 1583 validiert. Mit dem Befehl wird die Validierung sofort durchgeführt und die Ergebnisse werden im Fenster "Meldungen" angezeigt.

Anmerkung: Die tatsächliche Leistung hängt von der Anzahl der PC-Prozessorkerne ab, die von RaptorXML Server für die Validierung verwendet werden: Je mehr Kerne verwendet werden, desto schneller ist die Verarbeitung.

Wenn Sie mehrere Konfigurationen auf mehreren Servern definiert haben, können Sie einen Server und eine seiner Konfigurationen als aktive Konfiguration auswählen. Die aktive Konfiguration wird in der Folge für Validierungen verwendet. Wenn Sie den Cursor über den Befehl **Extras | Raptor-Server und Konfigurationen** (*siehe Abbildung unten*) platzieren, wird ein Untermenü mit allen hinzugefügten Servern und deren Konfiguration angezeigt. Wählen Sie die Konfiguration, die Sie zur aktiven Konfiguration machen wollen, aus. In der Abbildung unten wurde die Konfiguration `xbrl` des Servers mit dem Namen `Raptor-01` als aktive Konfiguration ausgewählt (gekennzeichnet durch einen grünen Pfeil).



Der Befehl **XML am Server validieren (Hochleitungsvalidierung) (Strg+F8)** steht auch im Fenster "Projekt" zur Verfügung. Klicken Sie mit der rechten Maustaste auf das Projekt, einen Ordner oder eine Datei und wählen Sie den Befehl **XML am Server validieren,** um XML- oder XBRL-Daten im ausgewählten Objekt zu validieren.

Anmerkung: Die Raptor-Validierung steht in der Text- der Grid- und der XBRL-Ansicht zur Verfügung.

28.4.14 Validieren von WSDL-Dateien

Ein WSDL-Dokument ist nicht nur ein WSDL-Dokument, sondern gleichzeitig auch ein XML-Dokument. Es kann daher als XML- und auch als WSDL-Dokument validiert werden. Die folgende Liste enthält wichtige Informationen, darüber, wie die Validierung von WSDL-Dateien[1353] in der *Enterprise* und der *Professional Edition* von XMLSpy gehandhabt wird.

- Die *Professional Edition* führt eine einfache Schemavalidierung durch, d.h. die WSDL-Datei wird als XML-Datei behandelt und dem unter `http://schemas.xmlsoap.org/wsdl/` definierten Schema entsprechend validiert.
- Die *Enterprise Edition* bietet Funktionen zur Validierung von WSDL-Dateien, die über den Umfang der XML-Validierungsfunktion in der *Professional Edition* hinausgehen. Die Validierung darin wird nicht anhand von `http://schemas.xmlsoap.org/wsdl/` durchgeführt, sondern anhand des Dokuments `http://www.altova.com/specs_wsdl.html#_document-s` sowie der eigenen Programmlogik. Dadurch stehen im Kontext von WSDL zusätzliche Validierungsinformationen zur Verfügung. Dadurch kann es vorkommen, dass eine WSDL-Datei zwar in der *Professional Edition* gültig ist, nicht aber in der *Enterprise Edition* (*siehe Beispiel unten*).
- Es besteht ein Unterschied zwischen `http://schemas.xmlsoap.org/wsdl/` und `http://www.altova.com/specs_wsdl.html#_document-s`. Ersteres Schema enthält keine Definitionen von **Erweiterungselementen**, welche in der WSDL-Spezifikation definiert sind. Offenbar handelt es sich hierbei um einen Fehler im offiziellen W3C-Schema; dieser Fehler wurde im zweitgenannten (von der *Enterprise Edition*) verwendeten Schema korrigiert.
- Da die Validierung in der *Professional Edition* anhand von `http://schemas.xmlsoap.org/wsdl/` durchgeführt wird, werden Erweiterungselemente in der *Professional Edition* als ungültig behandelt, während sie in der *Enterprise Edition* (in der die Validierung anhand von `http://www.altova.com/specs_wsdl.html#_document-s` durchgeführt wird) gültig sind..
- Da es sich beim W3C-Schema um ein offizielles Schema der W3C-Arbeitsgruppe handelt, sind Altova bei Fehlern darin die Hände gebunden.

Beispiel

Das folgende Beispiel stammt aus einer WSDL-Datei. Beachten Sie das in der Datei deklarierte Element `getCityTime`. Dieses Element wird irrtümlicherweise als `getCityTimes` referenziert. Die *Enterprise Edition* überprüft, ob referenzierte Elemente zuvor in der Datei deklariert wurden; die *Professional Edition* tut dies nicht. Diese Datei wäre in der *Professional Edition* gültig (vorausgesetzt der Rest der Datei ist gültig), nicht aber in der *Enterprise Edition* (es sei denn, `getCityTimes` ist an einer anderen Stelle in der Datei definiert).

```
<s:element name="getCityTime">
   <s:complexType>
      <s:sequence>
         <s:element minOccurs="0" maxOccurs="1" name="city" type="s:string"/>
      </s:sequence>
   </s:complexType>
</s:element>
<s:element name="abc">
   <s:complexType>
      <s:sequence>
         <s:element ref="getCityTimes"/>
      </s:sequence>
   </s:complexType>
</s:element>
```

28.4.15 Bei Bearbeitung validieren

Mit dem Befehl **Bei Bearbeitung validieren** wird der *Validierung bei Bearbeitung*-Modus, in dem das Dokument während der Eingabe in die Text- 144, JSON-Grid-Ansicht 701 oder Authentic-Ansicht validiert wird, aktiviert bzw. deaktiviert. Dieser Modus kann auch über die Symbolleisten-Schaltfläche des Befehls oder die Option *Validierung* > *Bei Bearbeitung* des Abschnitts "Datei" des Dialogfelds "Optionen" 1603 aktiviert bzw. deaktiviert werden.

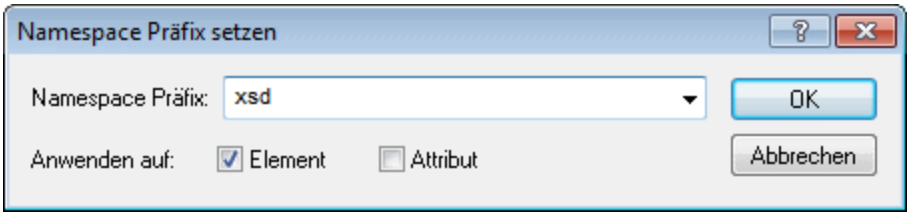
28.4.16 Eingabehilfen aktualisieren



Mit dem Befehl **Eingabehilfen aktualisieren** aktualisieren Sie die Eingabehilfen-Fenster. Dies erfolgt durch erneutes Laden der zugrunde liegenden DTD / des zugrunde liegenden Schemas. Sollten Sie das Schema oder die DTD, auf dem/auf der ein offenes XML-Dokument basiert, verändert haben, empfiehlt es sich, die Eingabehilfen zu aktualisieren, damit die intelligenten Editierinformationen die Änderungen aus dem Schema anzeigen.

28.4.17 Namespace Präfix

Dieser Befehl steht in der Grid-Ansicht zur Verfügung und öffnet ein Dialogfeld, in dem Sie das Namespace Präfix des ausgewählten Elements oder Attributs - und im Fall von Elementen - seiner Nachfahren definieren können.

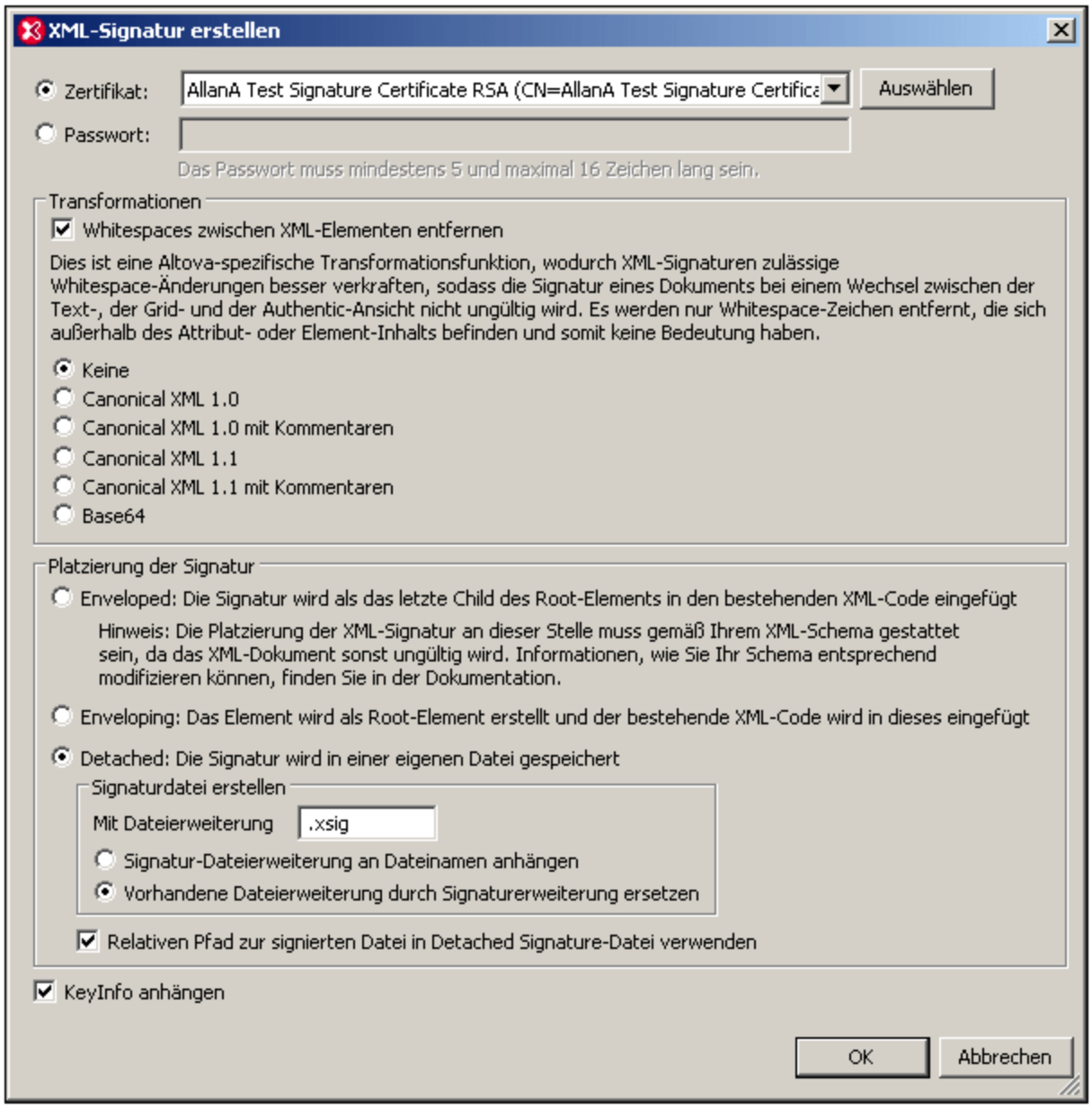


Sie können das Namespace Präfix für Elemente, Attribute oder für beide setzen. Das Namespace Präfix wird auf das ausgewählte Element oder Attribut angewendet und - wenn ein Element ausgewählt ist - auch auf die untergeordneten Nodes des ausgewählten Elements.

28.4.18 XML-Signatur erstellen

Der Befehl **XML-Signatur erstellen** ist in der Textansicht, der Grid-Ansicht, der Schema-Ansicht, der WSDL-Ansicht und der XBRL-Ansicht aktiv und dient zum Erstellen einer XML-Signatur für das aktive Dokument. Wenn Sie auf den Menübefehl **XML | XML-Signatur erstellen** klicken, wird das Dialogfeld "XML-Signatur erstellen" (*Abbildung unten*) geöffnet, dessen Einstellungen im Folgenden erläutert werden.

© 2016-2022 Altova GmbH



Authentifizierungsmethode: Zertifikat oder Passwort

Die Signatur kann auf einem Zertifikat oder einem Passwort basieren. Wählen Sie das Optionsfeld für die gewünschte Methode aus.

- *Zertifikat:* Wenn Sie ein Zertifikat verwenden möchten, muss dieses einen privaten Schlüssel (private key) haben und sich in einem Zertifikatspeicher[426] befinden, auf den Zugriff besteht. Die Signatur wird anhand des privaten Schlüssels des Zertifikats generiert. Zur Überprüfung der Signatur muss Zugriff auf das Zertifikat (oder eine Public-Key-Version davon) bestehen. Der öffentliche Schlüssel (public key) des Zertifikats dient zur Überprüfung der Signatur. Klicken Sie zur Auswahl des Private-Public-Key-

Zertifikats auf die Schaltfläche **Auswählen** und navigieren Sie zum Zertifikat. Nähere Informationen über Zertifikate finden Sie im Abschnitt Arbeiten mit Zertifikaten[425].

- *Passwort:* Geben Sie ein Passwort mit einer Länge von fünf bis 16 Zeichen ein. Dieses Passwort wird später benötigt, um die Signatur zu überprüfen.

Transformationen

Die XML-Daten werden transformiert und das Ergebnis der Transformation wird zur Erstellung der Signatur verwendet. Sie können den Kanonisierungsalgorithmus, der auf die XML-Daten der Datei (den `SignedInfo`-Inhalt) anzuwenden ist, definieren, bevor die Signatur berechnet wird. Im Folgenden einige wichtige Unterschiede zwischen den Algorithmen:

- *Canonical XML mit oder ohne Kommentare:* Wenn Kommentare bei der Berechnung der Signatur einbezogen werden, so führt jede Änderung an den Kommentaren in den XML-Daten zu einem Fehler bei der Überprüfung der Signatur. Im anderen Fall dürfen Kommentare nach der Signierung des Dokuments geändert oder hinzugefügt werden, ohne dass die Signatur dadurch ungültig wird.
- *Base64:* Das Root- (oder Dokument-) Element des XML-Dokuments wird als Base64-kodiert betrachtet und in seiner Binärform gelesen. Wenn das Root-Element nicht in Base64 kodiert ist, wird ein Fehler zurückgegeben oder das Element als leer gelesen, je nachdem, welcher Elementtyp verwendet wird.
- *Keine:* Es wird keine Transformation durchgeführt und die XML-Daten aus der auf der Festplatte gespeicherten Binärdatei werden direkt an die Signaturüberprüfung übergeben. Alle späteren Änderungen an den Daten führen dazu, dass die Signatur nicht verifiziert werden kann. Wenn allerdings das Kontrollkästchen *Whitespaces entfernen* aktiviert ist, so werden alle Whitespace-Zeichen entfernt und Änderungen an Whitespace-Zeichen werden bei der Signaturüberprüfung ignoriert. Ein grundlegender Unterschied zwischen der Option *Keine* und einer der *Canonical*-Optionen ist, dass bei der Kanonisierung ein XML-Datenstrom erzeugt wird, in dem einige Unterschiede wie z.B. die Reihenfolge der Attribute normalisiert wird. Als Folge davon werden Änderungen wie eine andere Attributreihenfolge bei einer Kanonisierungstransformation normalisiert (sodass die Signatur verifiziert werden kann), während eine solche Änderung berücksichtigt wird, wenn keine Transformation durchgeführt wird (d.h. die Signatur kann nicht verifiziert werden).

Platzierung der Signatur

Die Signatur kann in die XML-Datei platziert oder als separate Datei erstellt werden. Es stehen die folgenden Optionen zur Verfügung:

- *Enveloped:* Das Signaturelement wird als das letzte Child-Element des Root (Dokument)-Elements erstellt.
- *Enveloping:* Das Signaturelement wird als das Root- (Dokument)-Element erstellt und das XML-Dokument wird als Child-Element in dieses Element eingefügt.
- *Detached:* Die XML-Signatur wird in Form einer separaten Datei erstellt. In diesem Fall können Sie die Dateierweiterung der Signatur definieren und festlegen ob: (i) die Erweiterung an den Namen der XML-Datei angehängt werden soll (z.B. `test.xml.xsig`) oder (ii) ob die Erweiterung die XML-Erweiterung der Datei ersetzen soll (z.B. `test.xsig`). Außerdem können Sie festlegen, ob die Referenz auf die XML-Datei in der Signaturdatei ein relativer oder absoluter Pfad sein soll.

Anmerkung: XML-Signaturen für XML-Schema- (`.xsd`) und XBRL-Dateien können nur als externe Signaturdateien erstellt werden. Signaturen für WSDL-Dateien können als externe Dateien erstellt und über die Option "Enveloped" in die WSDL-Datei inkludiert werden.

Anmerkung: Die XML-Signatur wird als separate Datei erstellt. Anschließend werden die XML-Datei und die Signaturdatei durch eine Referenz in der Signaturdatei miteinander verknüpft. Folglich muss, wenn sich

 © 2016-2022 Altova GmbH

die Signatur in einer externen Datei befindet, bei der Überprüfung der Signatur die Signaturdatei aktiv sein und nicht die XML-Datei.

Schlüsselinformationen anhängen

Die Option *KeyInfo anhängen* steht zur Verfügung, wenn die Signatur auf einem Zertifikat basiert. Wenn die Signatur auf einem Passwort basiert, steht sie nicht zur Verfügung.

Wenn die Option aktiv ist, werden die Daten des öffentlichen Schlüssels in die Signatur platziert. Im anderen Fall werden die Schlüsselinformationen nicht in die Signatur inkludiert. Der Vorteil bei der Inkludierung der Schlüsselinformationen ist, dass das Zertifikat selbst (insbesondere die Daten des öffentlichen Schlüssels) für die Überprüfung nicht benötigt werden (da die Schlüsselinformationen ja in der Signatur vorhanden sind).

28.4.19 XML-Signatur überprüfen

Eine XML-Signatur wird als korrekt bestätigt, wenn die XML-Datei seit der Signierung nicht mehr geändert wurde. Andernfalls schlägt die Überprüfung fehl. Mit dem Befehl **XML-Signatur überprüfen** wird die Überprüfung durchgeführt und die Ergebnisse der Überprüfung werden im Fenster "Meldungen" angezeigt. Im Folgenden sind die verschiedenen Überprüfungsszenarios in XMLSpy beschrieben:

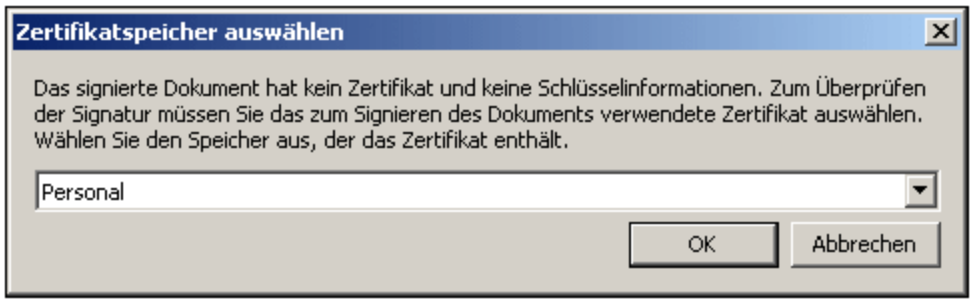
- Die XML-Datei enthält eine zertifikatbasierte Signatur, wobei die Schlüsselinformationen in der Signatur inkludiert sind 1363
- Die XML-Datei enthält eine zertifikatbasierte Signatur, wobei die Schlüsselinformationen nicht in der Signatur inkludiert sind 1363
- Die zertifikatbasierte Signatur befindet sich in einer externen Datei, die Schlüsselinformationen sind in der Signatur enthalten 1364
- Die zertifikatbasierte Signatur befindet sich in einer externen Datei, wobei die Schlüsselinformationen nicht in der Signatur enthalten sind 1365
- Die XML-Datei enthält eine passwortbasierte Signatur 1365
- Die passwortbasierte Signatur befindet sich in einer externen Datei 1365

Die XML-Datei enthält eine zertifikatbasierte Signatur, wobei die Schlüsselinformationen in der Signatur inkludiert sind

Um die XML-Signatur in diesem Szenario zu überprüfen, machen Sie die XML-Datei in XMLSpy zu aktiven Datei. Wenn Sie den Befehl **XML | XML-Signatur überprüfen** auswählen, wird die Überprüfung durchgeführt und das Ergebnis wird im Fenster "Meldungen" angezeigt (Überprüfung erfolgreich oder fehlgeschlagen).

Die XML-Datei enthält eine zertifikatbasierte Signatur, wobei die Schlüsselinformationen nicht in der Signatur inkludiert sind

Falls die Schlüsselinformationen nicht in der zertifikatbasierten Signatur enthalten sind, werden Sie von XMLSpy aufgefordert, das Zertifikat anzugeben, aus dem die Daten des öffentlichen Schlüssels für die Überprüfung ausgelesen werden können. Bei der Überprüfung muss die XML-Datei in XMLSpy aktiv sein. Wenn Sie den Befehl **XML | XML-Signatur überprüfen** auswählen, werden Sie aufgefordert, den Zertifikatspeicher 425, dem das Zertifikat sich befindet, auszuwählen (*Abbildung unten*).



Nach Auswahl eines Zertifikatspeichers [425] und Klicken auf **OK** wird ein Dialogfeld angezeigt, in dem die Zertifikate in diesem Speicher angezeigt werden (*Abbildung unten*). Wählen Sie das gewünschte Zertifikat aus und klicken Sie auf **OK**.



Die Signatur wird überprüft und das Ergebnis wird im Fenster "Meldungen" angezeigt.

Die zertifikatbasierte Signatur befindet sich in einer externen Datei, die Schlüsselinformationen sind in der Signatur enthalten

Wenn sich eine zertifikatbasierte XML-Signatur in einer externen Datei befindet, muss bei der Überprüfung der Signatur die Signaturdatei in XMLSpy aktiv sein. Wenn Sie den Befehl **XML | XML-Signatur überprüfen** auswählen, wird die Überprüfung durchgeführt und das Ergebnis im Fenster "Meldungen" angezeigt (Überprüfung erfolgreich oder fehlgeschlagen).

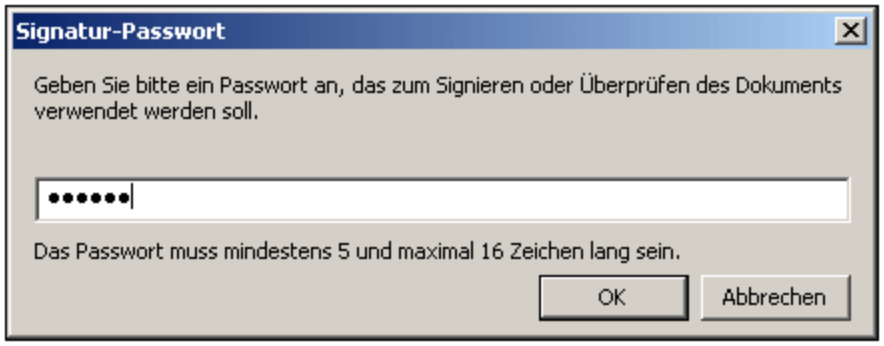
© 2016-2022 Altova GmbH

Die zertifikatbasierte Signatur befindet sich in einer externen Datei, wobei die Schlüsselinformationen nicht in der Signatur enthalten sind

Wenn sich eine zertifikatbasierte XML-Signatur in einer externen Datei befindet, muss bei der Überprüfung der Signatur die Signaturdatei in XMLSpy aktiv sein. Wenn Sie den Befehl **XML | XML-Signatur überprüfen** auswählen, werden Sie aufgefordert das Zertifikat anzugeben, aus dem die Daten für den öffentlichen Schlüssel für die Überprüfung ausgelesen werden können. Wählen Sie das Zertifikat wie im Abschnitt: *Die XML-Datei enthält eine zertifikatbasierte Signatur, wobei die Schlüsselinformationen nicht in der Signatur inkludiert sind*[1363] beschrieben ist, aus. Die Überprüfung wird durchgeführt und das Ergebnis wird im Fenster "Meldungen" angezeigt (Überprüfung erfolgreich oder fehlgeschlagen).

Die XML-Datei enthält eine passwortbasierte Signatur

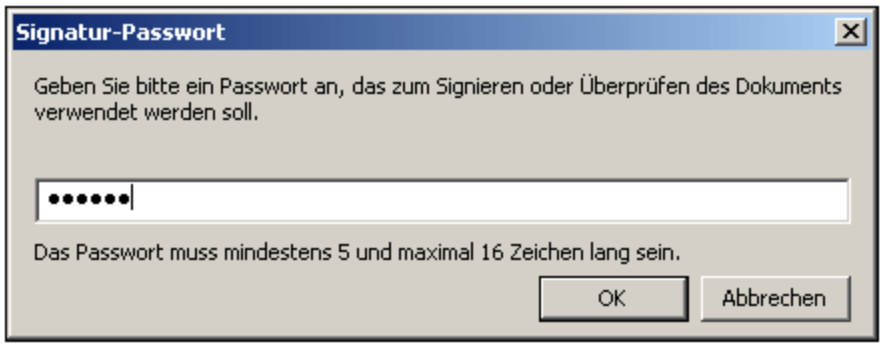
Wenn die XML-Datei eine passwortbasierte XML-Signatur enthält, muss bei der Überprüfung der Signatur die XML-Datei in XMLSpy aktiv sein. Wenn Sie den Befehl **XML | XML-Signatur überprüfen** auswählen, erscheint ein Dialogfeld, in dem Sie aufgefordert werden, das Passwort einzugeben (*Abbildung unten*).



Geben Sie das Passwort ein, das fünf bis sechzehn Zeichen lang sein muss, und klicken Sie auf **OK**. Die Überprüfung wird durchgeführt und das Ergebnis wird im Fenster "Meldungen" angezeigt (Überprüfung erfolgreich oder fehlgeschlagen).

Die passwortbasierte Signatur befindet sich in einer externen Datei

Wenn sich eine passwortbasierte XML-Signatur in einer externen Datei befindet, muss bei der Überprüfung der Signatur die Signaturdatei in XMLSpy aktiv sein. Wenn Sie den Befehl **XML | XML-Signatur überprüfen** auswählen, erscheint ein Dialogfeld, in dem Sie aufgefordert werden, das Passwort einzugeben (*Abbildung unten*).

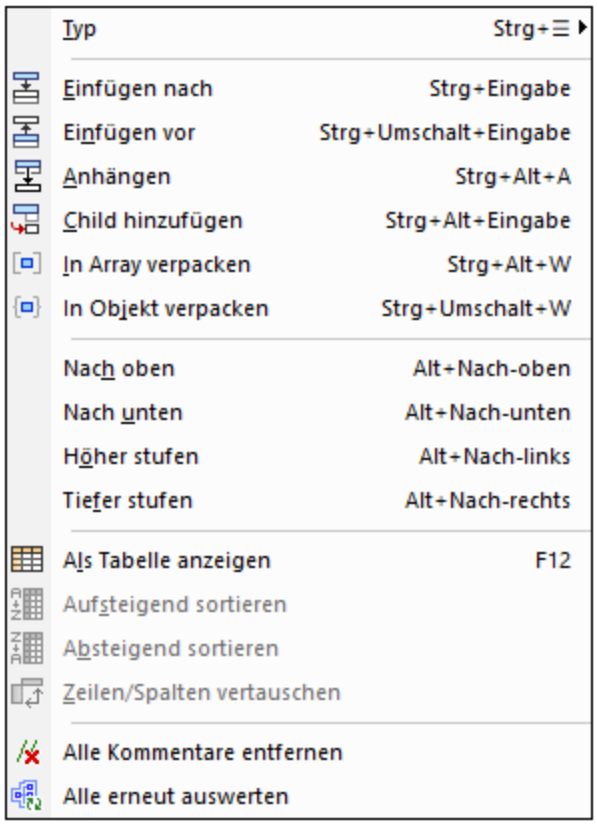


Geben Sie das Passwort ein, das fünf bis sechzehn Zeichen lang sein muss und klicken Sie auf **OK**. Die Überprüfung wird durchgeführt und das Ergebnis wird im Fenster "Meldungen" angezeigt (Überprüfung erfolgreich oder fehlgeschlagen).

© 2016-2022 Altova GmbH

28.5 Menü "JSON"

Das Menü **JSON** enthält Befehle, die bei der Arbeit mit JSON-Dokumenten verwendet werden. Die meisten der Befehle kommen beim Arbeiten in der JSON-Grid-Ansicht[690] zum Einsatz. Wenn ein Befehl an der aktuellen Cursorposition nicht verwendet werden kann, ist er deaktiviert.



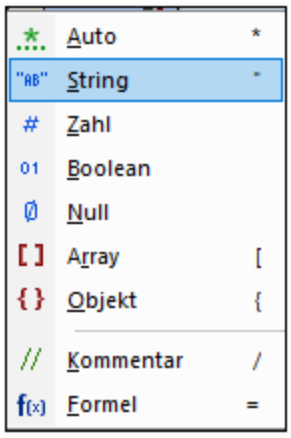
Die Befehle dieses Menüs werden in den Unterabschnitten dieses Abschnitts beschrieben:

- Typ[1367]
- Einfügen nach&vor, Anhängen, Child hinzufügen[1368]
- In Array/Objekt verpacken[1368]
- Verschieben[1369]
- Als Tabelle anzeigen[1369]
- Aufsteigend/Absteigend sortieren[1369]
- Zeilen/Spalten vertauschen[1369]
- Kommentare entfernen, Alle erneut auswerten[1370]

28.5.1 Typ

Mit dem Befehl **Typ** (**Strg + Menütaste**) wird ein Menü mit JSON-Datentypen aufgerufen *(Abbildung unten).* Wählen Sie einen Typ aus dem Menü aus, um ihn der/den aktuell ausgewählten Zelle(n) in der Grid-Ansicht

zuzuweisen. (Die **Menütaste** befindet sich normalerweise rechts unten auf der Tastatur, neben der **Strg**-Taste. Ihr Symbol ist:).



Nähere Informationen zum Bearbeiten von Typen finden Sie unter Bearbeiten von JSON-Dokumentinhalt[699].

28.5.2 Einfügen nach/vor, Anhängen, Child hinzufügen

Die Befehle **Einfügen nach**, **Einfügen vor**, **Anhängen** und **Child hinzufügen** sind aktiv, wenn eine Komponente in der Grid-Ansicht an der aktuell ausgewählten Stelle eingefügt, bzw. angehängt oder als Child hinzugefügt werden kann.

- Mit **Einfügen nach** wird eine Komponente desselben Typs wie die ausgewählte Komponente in eine Rasterzeile unterhalb der ausgewählten Komponente eingefügt.
- Mit **Einfügen vor** wird eine Komponente desselben Typs wie die ausgewählte Komponente in eine Rasterzeile oberhalb der ausgewählten Komponente eingefügt.
- Mit **Anhängen** wird eine Komponente desselben Typs wie die ausgewählte Komponente in einer Rasterzeile nach allen gleichrangigen Komponenten angehängt.
- Mit **Child hinzufügen** wird eine neue Child-Komponente als letztes Child hinzugefügt. Der Typ ist derselbe wie der des letzten Childs.

Nähere Informationen zum Bearbeiten der JSON-Dokumentstruktur finden Sie unter Bearbeiten der JSON-Dokumentstruktur[697].

28.5.3 In Array/Objekt verpacken

Mit den Befehlen **In Array verpacken** und **In Objekt verpacken** wird/werden die ausgewählte(n) Komponente(n) in ein Array bzw. Objekt verpackt. Nähere Informationen zum Bearbeiten der JSON-Dokumentstruktur finden Sie unter Bearbeiten der JSON-Dokumentstruktur[697].

© 2016-2022 Altova GmbH

28.5.4 Verschieben

Eine Komponente kann im Raster nach oben, unten, links oder rechts verschoben werden. In diesem Fall sind die entsprechende Befehle aktiv. Wählen Sie den entsprechenden Befehl aus, um die Komponente zu verschieben.

28.5.5 Als Tabelle anzeigen

Der Befehl **Als Tabelle anzeigen** ist aktiviert, wenn in der Grid-Ansicht eine sich wiederholende Komponente ausgewählt ist. Dies ist ein Ein/Aus-Befehl, mit dem Sie zwischen der Anzeige sich wiederholender Komponenten in der Standard-Grid-Ansicht[169] und der Tabellenansicht[312] wechseln können. In der Tabellenansicht können sich wiederholende Elemente als Tabelle, in der Zeilen die Instanzen und Spalten die Child-Nodes abbilden, angezeigt werden.

Nähere Informationen dazu finden Sie im Abschnitt Tabellenansicht[704].

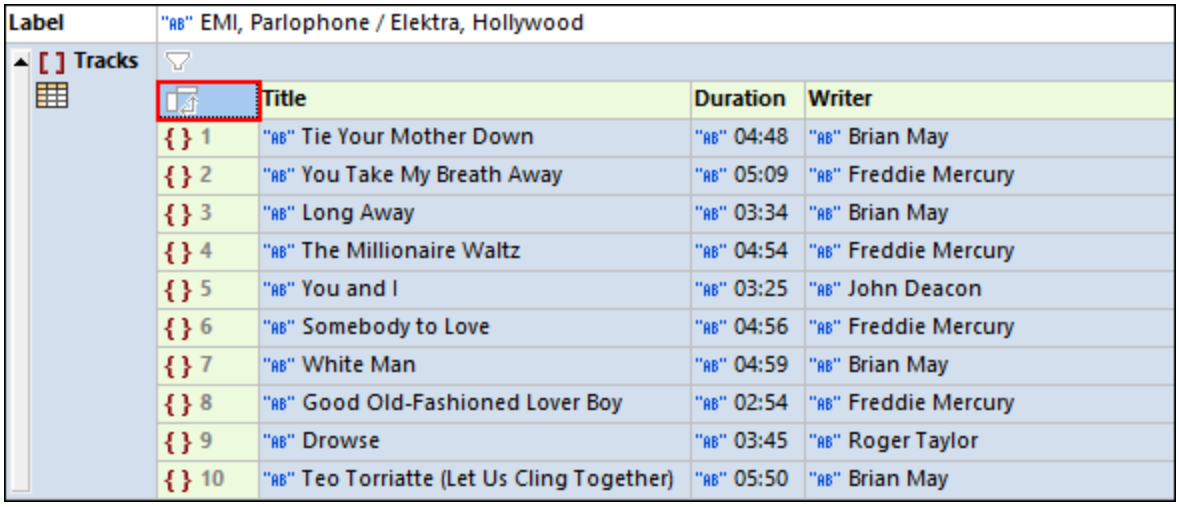
28.5.6 Aufsteigend/Absteigend sortieren

Die Befehle **Aufsteigend sortieren** und **Absteigend sortieren** sind in der Tabellenansicht[704] aktiviert, wenn in der Tabellenansicht einer Komponente eine Spalte ausgewählt ist. Um eine Spalte auszuwählen, klicken Sie auf ihre Überschrift. Die Sortierung erfolgt anhand der Werte in der Spalte.

Nähere Informationen dazu finden Sie im Abschnitt Tabellenansicht[704].

28.5.7 Zeilen/Spalten vertauschen

Der Befehl **Zeilen/Spalten vertauschen** ist in der Tabellenansicht[312] aktiviert, wenn die oberste linke Zelle einer Tabelle ausgewählt ist (*in der Abbildung unten rot markiert*). Mit dem Befehl werden Zeilen als Spalten angezeigt bzw. Spalten als Zeilen.



Label	"AB" EMI, Parlophone / Elektra, Hollywood		
[ ] Tracks			
	Title	Duration	Writer
{ } 1	"AB" Tie Your Mother Down	"AB" 04:48	"AB" Brian May
{ } 2	"AB" You Take My Breath Away	"AB" 05:09	"AB" Freddie Mercury
{ } 3	"AB" Long Away	"AB" 03:34	"AB" Brian May
{ } 4	"AB" The Millionaire Waltz	"AB" 04:54	"AB" Freddie Mercury
{ } 5	"AB" You and I	"AB" 03:25	"AB" John Deacon
{ } 6	"AB" Somebody to Love	"AB" 04:56	"AB" Freddie Mercury
{ } 7	"AB" White Man	"AB" 04:59	"AB" Brian May
{ } 8	"AB" Good Old-Fashioned Lover Boy	"AB" 02:54	"AB" Freddie Mercury
{ } 9	"AB" Drowse	"AB" 03:45	"AB" Roger Taylor
{ } 10	"AB" Teo Torriatte (Let Us Cling Together)	"AB" 05:50	"AB" Brian May

Nähere Informationen dazu finden Sie im Abschnitt Tabellenansicht[704].

28.5.8 Kommentare entfernen, Alle erneut auswerten

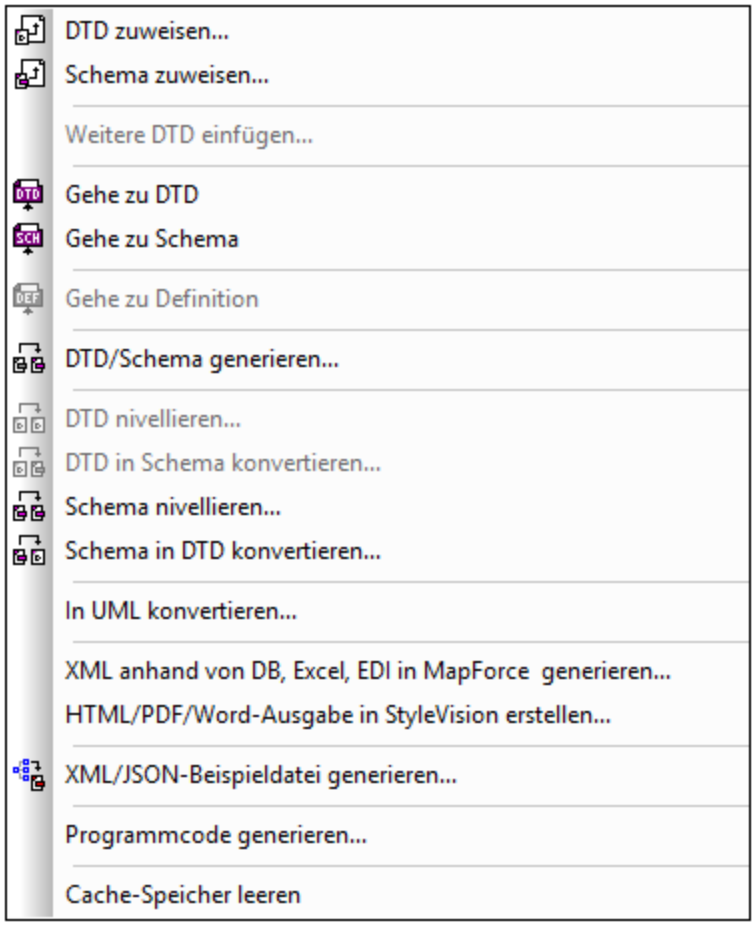
Mit dem Befehl **Kommentare entfernen** werden alle Kommentare entfernt. Der Befehl kann in der JSON-Grid-Ansicht verwendet werden.

Mit dem Befehl **Alle erneut auswerten** werden alle Filter[709] und Formeln[710] erneut ausgewertet. Dies ist nützlich, wenn im JSON-Dokument dynamische Daten aufgerufen werden, die sich im Laufe der Zeit ändern können (z.B. Wechselkurse). Der Befehl kann in der Text- und der JSON-Grid-Ansicht verwendet werden.

© 2016-2022 Altova GmbH

28.6 Menü "DTD/Schema"

Das **DTD/Schema**-Menü (*Abbildung unten*) enthält Befehle für das Arbeiten mit DTDs und XML-Schemas.



28.6.1 DTD zuweisen



Dieser Befehl ist aktiv, wenn eine XML-Datei aktiv ist. Sie weisen damit einem XML-Dokument eine DTD zu, wodurch Validierung und intelligentes Editieren ermöglicht wird. Wenn Sie diesen Befehl wählen, wird das Dialogfeld zum Zuweisen einer Datei geöffnet, in dem Sie die gewünschte DTD zuweisen können. Sie können eine Datei auch über eine globale Ressource, eine URL (klicken Sie dazu auf die Schaltfläche **Durchsuchen)** 1280 oder eines der offenen Fenster in XMLSpy auswählen (Klicken Sie dazu auf die Schaltfläche **Fenster**). Beachten Sie, dass Sie den Pfad der zugewiesenen Datei relativ machen können. Aktivieren Sie dazu das Kontrollkästchen "Pfad relativ zu.. machen". Nach erfolgter Zuweisung enthält Ihr

Dokument eine DOCTYPE-Deklaration, die die zugewiesene DTD referenziert. Die DOCTYPE-Deklaration sieht ungefähr folgendermaßen aus:

```
<!DOCTYPE main SYSTEM"http://link.xmlspy.com/spyweb.dtd">
```

Beachten Sie, dass eine DTD einer neuen XML-Datei zum Zeitpunkt der Erstellung zugewiesen werden kann.

28.6.2 Schema zuweisen



Dieser Befehl ist aktiv, wenn eine XML-Datei aktiv ist. Sie weisen damit einem XML-Dokument ein XML-Schema zu, wodurch Validierung und intelligentes Editieren ermöglicht wird. Wenn Sie diesen Befehl wählen, wird das Dialogfeld zum Zuweisen einer Datei geöffnet, in dem Sie das gewünschte XML-Schema zuweisen können. Sie können eine Datei auch über eine globale Ressource, eine URL (klicken Sie dazu auf die Schaltfläche **Durchsuchen)** 1280 oder eines der offenen Fenster in XMLSpy auswählen (Klicken Sie dazu auf die Schaltfläche **Fenster**). Beachten Sie, dass Sie den Pfad der zugewiesenen Datei relativ machen können. Aktivieren Sie dazu das Kontrollkästchen "Pfad relativ zu... machen". Nach erfolgter Zuweisung enthält Ihr Dokument eine XML-Schema-Zuweisung mit dem benötigten Namespace. Die Schema-Zuweisung sieht ungefähr folgendermaßen aus:

```
      xmlns="http://www.xmlspy.com/schemas/icon/orgchart"
   xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
   xsi:schemaLocation="http://www.xmlspy.com/schemas/icon/orgchart
      http://schema.xmlspy.com/schemas/icon/orgchart.xsd"
```

28.6.3 Weitere DTD einfügen

Mit diesem Befehl können Sie eine weitere Dokument Type Definition (DTD) oder extern geparste Entity in das interne Subset einer DTD, oder in jede andere DTD inkludieren. Dies geschieht durch Definition einer entsprechenden extern geparsten Entity-Deklaration und Verwendung dieser Entity in folgender Zeile:

```
<!ENTITY % navigation.dtd SYSTEM"S:\xml\navigation.dtd">
%navigation.dtd;
```

Mit diesem Befehl öffnen Sie das Dialogfeld zum Zuweisen einer Datei, in dem Sie angeben können, welche DTD in Ihre DTD inkludiert werden soll.

Bitte beachten Sie: Dieser Befehl ist nur in der Grid-Ansicht aktiv.

© 2016-2022 Altova GmbH

28.6.4 Gehe zu DTD



Mit diesem Befehl öffnen Sie die DTD, auf der das aktive XML-Dokument basiert. Wurde keine DTD zugewiesen, wird eine Fehlermeldung angezeigt.

28.6.5 Gehe zu Schema



Mit diesem Befehl öffnen Sie das XML-Schema, auf dem das aktive XML-Dokument basiert. Wurde kein XML-Schema zugewiesen, wird eine Fehlermeldung angezeigt.

28.6.6 Gehe zu Definition



Mit dem Befehl **DTD/Schema | Gehe zu Definition** wird die exakte Definition eines Elements oder Attributs in der zugehörigen DTD bzw. dem zugehörigen Schema-Dokument angezeigt

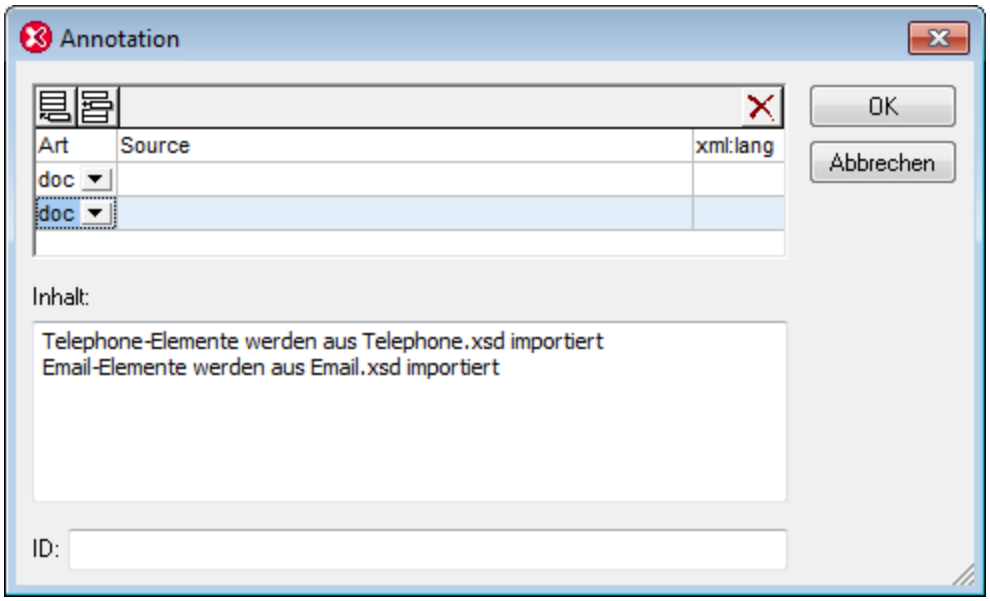
So zeigen Sie die Definition des Datenelements in der Grid Ansicht an

1. Klicken Sie auf das Datenelement.
2. Wählen Sie die Menüoption **DTD/Schema | Gehe zu Definition** oder klicken Sie auf das Symbol.

So zeigen Sie die Definition des Datenelements in der Schema-Ansicht an

- Mit Strg + Doppelklick auf das Datenelement, dessen Definition Sie ansehen möchten, oder
- Klicken Sie auf das Objekt und wählen Sie die Menüoption **DTD/Schema | Gehe zu Definition**, oder klicken Sie auf das Symbol.

In beiden Fällen wird die zugehörige DTD- oder Schemadatei geöffnet und die Definition des Datenelements wird markiert.

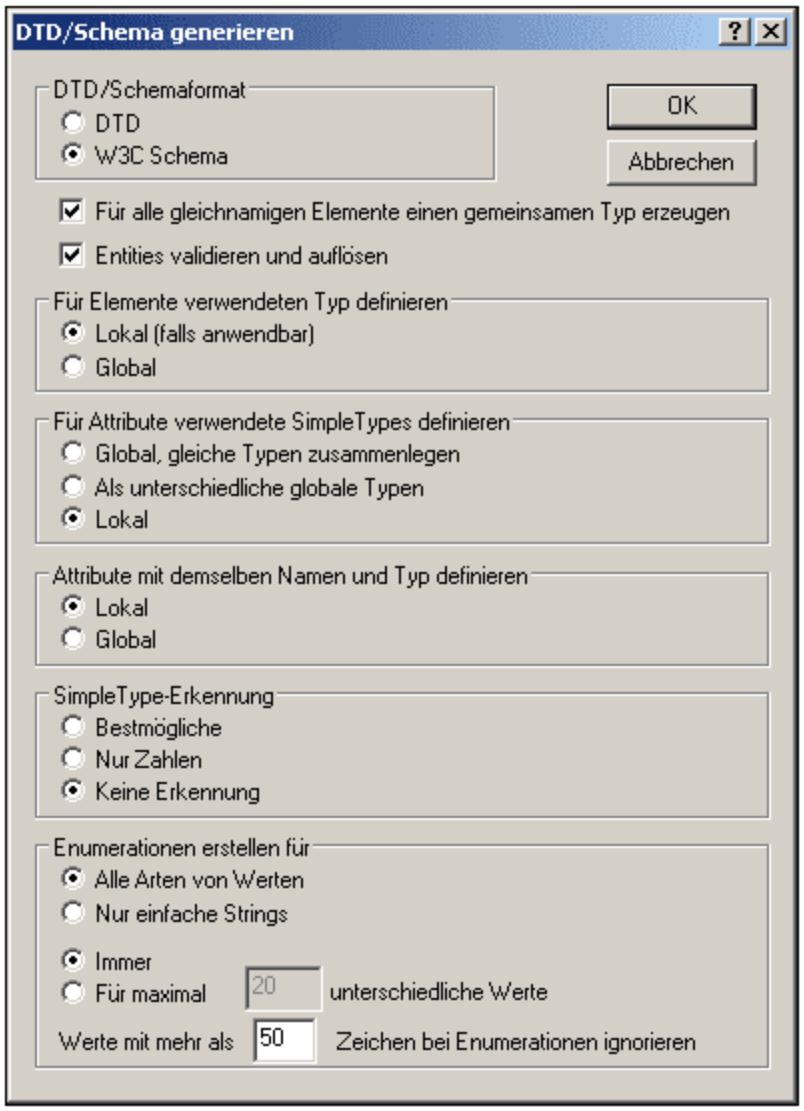


28.6.7 DTD/Schema generieren



Mit diesem Befehl wird eine neue DTD oder ein W3C XML-Schema von einem XML-Dokument (oder von einer Gruppe von Dokumenten im Projekt Fenster) erzeugt.
Verwenden Sie diese Funktion, wenn Sie ein neues Schema oder eine neue DTD anhand eines XML-Dokuments erzeugen möchten.

© 2016-2022 Altova GmbH



Bei der Generierung eines XML-Schemas stehen die folgenden Optionen zur Verfügung:

- ***Elemente***: Die Art der Elemente kann lokal oder global definiert werden (*für Elemente verwendeten Typ definieren*). Wenn Elemente denselben Namen haben, kann ein allgemeiner Typ definiert werden, der in der Definition dieser Elemente verwendet wird (*für alle gleichnamigen Elemente einen gemeinsamen Typ erstellen)*.
- ***Attribute***: Die simpleTypes von Attributen (*für Attribute verwendete simpleTypes definieren*) können definiert werden als: (i) allgemeine globale Typen; (ii) unterschiedliche globale Typen; (iii) lokale Typen. Attribute desselben Namens und Typs können entweder lokal oder global definiert werden.
- ***SimpleType-Erkennung***: Die Typerkennung (*SimpleType-Erkennung*) kann folgendermaßen festgelegt werden: (i) als bestmögliche; (ii) nur Zahlen-Datentypen werden erkannt; (iii) keine Datentyp-Erkennung. In diesem Fall werden alle Datentypen auf `xs:string` gesetzt.
- ***Entity-Auflösung***: Im XML-Dokument können Entities im Elementinhalt und in Attributwerten vorkommen. Es ist daher für Enumerationswerte von Bedeutung, ob diese Entities aufgelöst werden sollen oder nicht (*Entities validieren und auflösen*). Außerdem können sich einige Entities (v.a.

geparste Entities, die Markup-Tags enthalten) unterschiedlich auf das Content Model auswirken, je nachdem, ob sie aufgelöst werden oder nicht. Beachten Sie, dass das XML-Dokument auf seine korrekte XML-Syntax validiert wird, bevor das Schema generiert wird. Wenn das Dokument ungültig ist, wird die Schemagenerierung abgebrochen.

- ***Enumerationen***: Es können Enumerationen für alle Wertetypen oder nur für Stringwerte erstellt werden.

Bei Generierung einer DTD stehen die Optionen für die Entity-Auflösung und Enumerationen zur Verfügung.

Der Befehl **DTD/Schema generieren** wird normalerweise im aktiven Hauptfenster durchgeführt, allerdings können Sie ihn auch auf jede Datei, jeden Ordner oder jede Gruppe von Dateien im aktiven Projekt Fenster anwenden.

Wenn Elemente oder Attribute in mehr als einem Namespace vorhanden sind, generiert XMLSpy für jeden separaten Namespace ein eigenes XML-Schema. Es können daher auf der Festplatte mehrere Dateien erstellt werden.

28.6.8 DTD nivellieren

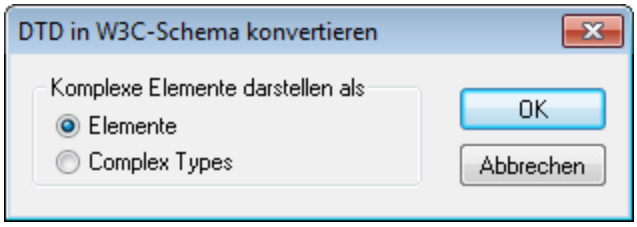
Der Befehl **DTD nivellieren** ist aktiv, wenn eine DTD das aktive Dokument ist. Mit dem Befehl wird eine neue, flache DTD erstellt, wobei Parameter-Entities entfernt werden und anhand einer Sammlung von Modulen eine einzige DTD erstellt wird. Mit `IGNORE` markierte Abschnitte werden unterdrückt und nicht verwendete Parameter Entities werden gelöscht.

Bei Aufruf des Befehls wird ein Speichern-Dialogfeld angezeigt, in dem Sie einen Ordner auswählen können, in dem die generierte DTD-Datei gespeichert wird. Klicken Sie auf **Speichern,** um die Konvertierung durchzuführen. Daraufhin wird die nivellierte DTD-Datei generiert und in XMLSpy geöffnet.

28.6.9 DTD in Schema konvertieren

Der Befehl **DTD in Schema konvertieren** ist aktiv, wenn eine DTD das aktive Dokument ist. Mit dem Befehl wird eine DTD in ein XML-Schema-Dokument (XSD) konvertiert.

Mit dem Befehl wird das Dialogfeld "DTD in W3C-Schema konvertieren" (*Abbildung unten*) aufgerufen, in dem Sie auswählen können, ob komplexe Elemente in Elemente oder complexTypes konvertiert werden sollen. Bei Klick auf **OK** werden Sie aufgefordert, einen Ordner auszuwählen, in dem die generierte XSD-Datei gespeichert werden soll. Klicken Sie auf **Speichern,** um die Konvertierung durchzuführen. Daraufhin wird die XSD-Datei generiert und in XMLSpy geöffnet.



 © 2016-2022 Altova GmbH

Bei der Konvertierung einer DTD in ein XML-Schema macht XMLSpy aufgrund der beschränkten Informationen, die zur Verfügung stehen, einige Annahmen. Beachten Sie vor allem, dass die Werte bestimmter DTD-Komponenten wörtlich interpretiert werden, anstatt dass ihre Semantik geparst wird, da das Programm nicht weiß, welche der vielen Möglichkeiten die richtige ist. Bearbeiten Sie die generierte Konvertierung in diesem Fall.

Das Resultat sollte jedenfalls immer genau überprüft und gegebenenfalls verbessert werden. Im Folgenden werden einige Bereiche angegeben, in denen Sie eventuell Korrekturen vornehmen sollten.

Definition von Datentypen für Attribute
DTDs unterstützen nur 10 Attribut-Datentypen, während in XML-Schemas z.B. mehr als 40 Datentypen plus davon abgeleitete Datentypen zulässig sind. Durch Verwendung von weniger Datentypen lässt sich ein generiertes XML-Schema unter Umständen verbessern. Beachten Sie, dass Datentyp-Informationen bei der Konvertierung eines XML-Schemas in eine DTD[1379] verloren gehen.

Namespaces
DTD erkennen keine Namespaces. Wenn daher in einer DTD Namespaces definiert werden, müssen sie in Form von Element- und Attributnamen hartkodiert werden. Bei der Konvertierung von einem Schema in ein anderes kann dies einige Probleme verursachen.

Entities
In XML-Schemas gibt es kein Äquivalent für die allgemeinen Entity-Deklarationen von DTDs. Wenn XMLSpy eine DTD in ein XML-Schema konvertiert, ignoriert es die Entity-Deklarationen.

Nicht geparste Datendeklarationen
DTDs und XML-Schemas verwenden bei der Behandlung nicht geparster Daten unterschiedliche Mechanismen. Dies wird im Folgenden näher erklärt.

DTDs verwenden den folgenden Mechanismus:

- Eine Notation wird aus einem Namen und einem Identifier bestehend deklariert, z.B.
  `<!NOTATION gif SYSTEM "image/gif">`
- Sie deklarieren die Entity, z.B.
  `<!ENTITY cover_img SYSTEM "graphics/cover_img.gif" NDATA gif>`
- Normalerweise gibt man zum betreffenden Attribut einen Attributtyp von ENTITY an, z.B.`<!ELEMENT img EMPTY>`
  `<!ATTLIST img format ENTITY #REQUIRED>`

In einem XML-Schema wird hierfür der folgende Mechanismus verwendet:

- Deklarieren Sie eine Notation. Dies funktioniert auf dieselbe Art wie bei der DTD.
  `<xs:notation name="gif" public="image/gif"/>`
  Beachten Sie, dass das Attribut `public` obligatorisch ist und den Identifier enthält. Ein optionales Attribut `system` enthält den System Identifier und ist normalerweise eine ausführbare Datei, die Ressourcen des Notationstyps verarbeiten kann.
- Man verknüpft die Notationsdeklaration über den NOTATION-Datentyp mit einem bestimmten Attributwert. Sie können den `NOTATION`-Datentyp jedoch nicht direkt verwenden, sondern müssen einen weiteren Datentyp aus dem NOTATION-Datentyp ableiten.

```
<xs:simpleType name="formatType">
  <xs:restriction base="xs:NOTATION">
    <xs:enumeration value="gif"/>
    <xs:enumeration value="jpeg"/>
```

```
    </xs:restriction>
  </xs:simpleType>
```

- Man verknüpft das Attribut mit dem vom `NOTATION`-Datentyp abgeleiteten Datentyp, z.B.

```
<xs:complexType name="imgType">
  <xs:attribute name="height"/>
  <xs:attribute name="width"/>
  <xs:attribute name="location"/>
  <xs:attribute name="format" type="formatType" use="required"/>
</xs:complexType>
<xs:element name="img" type="imgType"/>
```

Beim Konvertieren einer DTD in ein XML-Schema geht XMLSpy vor wie folgt:

- Die folgende Deklaration:

```
<!ATTLIST image format ENTITY #REQUIRED
...>
```

  wird in

```
<xs:attribute name="format" type="xs:ENTITY" use="required"/>
```

  konvertiert.

- und die folgende Deklaration

```
<!NOTATION gif SYSTEM "image/gif">
```

  wird in

```
<xs:notation name="gif" system="image/gif"/>
```

  konvertiert.

Nehmen Sie daher die folgenden Änderungen vor:

1. In Notationen wie `<xs:notation name="gif" system="image/gif"/>` ersetzen Sie `system` durch `public` und fügen einen optionalen System Identifier hinzu, falls erforderlich.
2. Leiten Sie aus dem `NOTATION` Datentyp einen Datentyp ab, wie oben für `formatType` beschrieben.
3. Verknüpfen Sie den abgeleiteten Datentyp mit dem entsprechenden Attribut.

Anmerkung: Gemäß der XML-Schemaspezifikation, ist es nicht notwendig - bzw. je nach Sichtweise, nicht möglich - eine externe Entity zu deklarieren.

28.6.10 Schema nivellieren

Der Befehl **Schema nivellieren** ist aktiv, wenn ein XML-Schema das aktive Dokument ist. Mit dem Befehl wird eine neue, flache XSD-Datei erstellt, wobei (i) die Komponenten aller inkludierten Schemas als globale Komponenten des aktiven Schemas hinzugefügt werden und (ii) die inkludierten Schemas gelöscht werden.

Der Befehl wird zum Befehl Schema nivellieren[1407] der Schema-Design-Ansicht umgeleitet. Da der Befehl Schema nivellieren[1407] nur in der Schema-Ansicht zur Verfügung steht, werden Sie gefragt, ob Sie in die

Schema-Ansicht wechseln möchten. Nähere Informationen dazu finden Sie unter dem Befehl Schema nivellieren[1407].

28.6.11 Schema in DTD konvertieren

Der Befehl **Schema in DTD konvertieren** ist aktiv, wenn ein XML-Schema das aktive Dokument ist. Mit dem Befehl wird ein XML-Schema-Dokument (XSD) in eine DTD konvertiert.

Bei Aufruf des Befehls wird ein Speichern-Dialogfeld angezeigt, in dem Sie einen Ordner auswählen können, in dem die generierte DTD-Datei gespeichert wird. Klicken Sie auf **Speichern,** um die Konvertierung durchzuführen. Daraufhin wird die nivellierte DTD-Datei generiert und in XMLSpy geöffnet.

Beachten Sie die folgenden Punkte:

1. Bei der Konvertierung eines XML-Schemas in eine DTD werden die im XML-Schema verwendeten Namespace-Präfixe - nicht die Namespace URIs oder die Namespace-Deklarationen - in die Namen der entsprechenden Elemente und Attribute in der DTD übernommen.

2. Da XML-Parser Namespaces beim Validieren eines Dokuments gegen eine DTD ignorieren, werden die Namespace-Deklarationen selbst nicht konvertiert.
3. Die Attribute `elementFormDefault` und `attributeFormDefault` des Elements `xs:schema` bestimmen, von welchen Elementen und Attributen die Präfixe im Konvertierungsvorgang übernommen werden. Ist der Wert auf `unqualified` gesetzt, werden nur die Präfixe von global deklarierten Elementen bzw. Attributen im Konvertierungsprozess übernommen. Ist der Wert auf `qualified` gesetzt, werden die Präfixe aller Element- und Attributnamen in den Konvertierungsprozess eingeschlossen.
4. Präfixe werden in den entsprechenden Stringwert plus Doppelpunkt konvertiert. Elemente und Attribute in Default Namespaces werden in Elemente und Attribute konvertiert, deren Namen mit: `default_NS_X` beginnt, wobei `X` eine Ganzzahl ist (beginnend mit 1 bis zu einem Maximalwert, der der Anzahl der im XML-Schema verwendeten Default Namespaces entspricht).
5. In der DTD sind Elementnamen aus Parameter-Entities zusammengesetzt. Auf diese Art können Sie das Präfix in der DTD jederzeit ändern, falls das Präfix im XML-Dokument einmal geändert werden muss. Parameter-Entity-Definitionen können entweder in der DTD selbst oder durch Außerkraftsetzung der Parameter-Entity-Definitionen im internen DTD Subset des XML-Dokuments geändert werden.

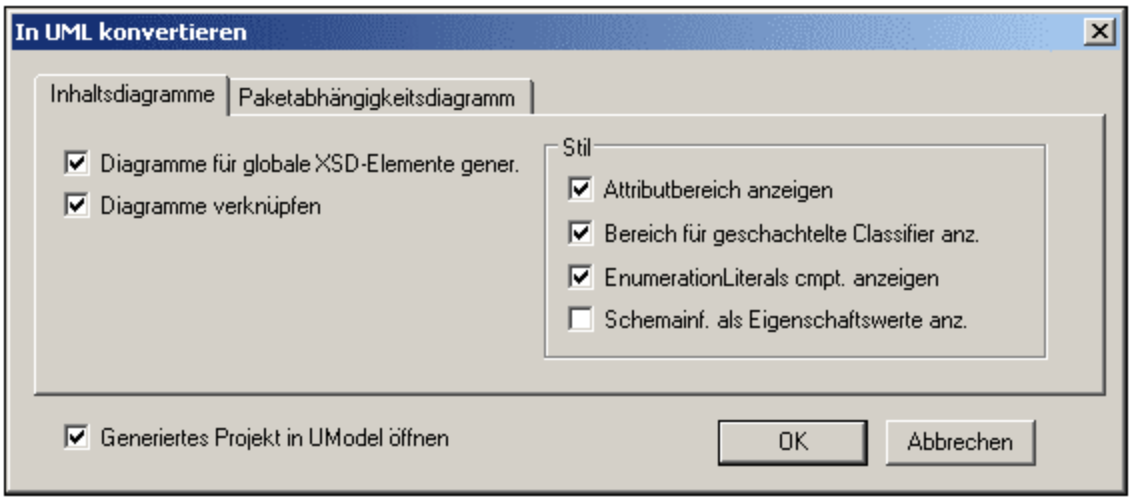
Anmerkung: Namespaces haben keinen semantischen Wert in einer DTD. Namespace-Präfixe, die aus dem XML-Schema übernommen werden, werden einfach nur Teil des Namens des in der DTD definierten Elements oder Attributs.

28.6.12 Im UML konvertieren

Mit dem Befehl **DTD/Schema | In UML konvertieren** wird ein W3C XML-Schema in ein Altova UModel-Projekt (`.ump`) Dokument (in Folgenden UModel-Projekt genannt) konvertiert. UMP ist das native Format von Altova UModel, der UML-Modellierungsapplikation von Altova. UMP-Dateien können in Altova UModel angezeigt und editiert werden.

Um ein Schema in UML zu konvertieren, gehen Sie folgendermaßen vor:

1. Klicken Sie, während das Schema geöffnet ist, auf den Befehl **In UML konvertieren**. Daraufhin erscheint das Dialogfeld "In UML konvertieren" (*Abbildung unten*).



2. Wählen Sie auf dem Register "Inhaltsdiagramme" die Option "Diagramm für globale XSD-Elemente generieren". Daraufhin wird im UModel-Projekt ein Content Model-Diagramm für jede globale Komponente erstellt.
3. Wählen Sie die erforderlichen Optionen aus den im Dialogfeld verfügbaren aus. Diese Optionen werden weiter unten beschrieben.
4. Wenn Sie das erstellte Projekt sofort in UModel ansehen möchten, wählen Sie die Option zum Öffnen des Projekts in UModel. Lassen Sie die Option andernfalls deaktiviert.
5. Klicken Sie auf **OK**.
6. Suchen Sie im Dialogfeld "Speichern unter", das nun angezeigt wird, den Zielordner, geben Sie anschließend den Namen der UMP-Datei an und klicken Sie auf **Speichern.**

Optionen für das Konvertieren in UML

Im Dialogfeld "In UML konvertieren" stehen die folgenden Optionen zur Verfügung.

Auf dem Register ***Inhaltsdiagramme***:

- *Diagramme verknüpfen* erstellt in jedem Diagramm einen Link zum Eintrag dieser globalen Komponente in der Modell-Strukturansicht, so dass Sie die Komponente in der Schemahierarchie schnell finden.
- Im Bereich *Stil* können Sie über die Optionen zur Anzeige der Bereiche verschiedene Bereiche ein oder ausblenden.

Auf dem Register ***Paketabhängigkeitsdiagramm***:

- Die Option *Diagramm generieren* legt fest, ob ein Paketabhängigkeitsdiagramm generiert werden soll. Ein Paketabhängigkeitsdiagramm bietet eine Übersicht über das gesamte Paket und zeigt die Beziehungen der Paketkomponenten zueinander. Beachten Sie, dass die anderen Optionen auf diesem Register nur aktiviert sind, wenn die Option "Diagramm generieren" aktiviert ist.
- Bei Auswahl der Option *Paket mit Diagramm verknüpfen* wird ein Link vom Paketdiagramm zur Modell-Strukturansicht erstellt.
- Für das Layout des Paketabhängigkeitsdiagramms stehen vier Optionen zur Verfügung: (i) nicht strukturiertes Layout (die Option "Automatisches Layout" ist deaktiviert); (ii) hierarchisches Layout (die Optionen "Automatisches Layout" und "hierarchisch" sind aktiviert); (iii) block (die Optionen

 © 2016-2022 Altova GmbH

"Automatisches Layout" und "Block" sind aktiviert); (iv) zentriert (die Optionen "Automatisches Layout" und "Zentriert" sind aktiviert). Das Layout kann durch Editieren des Diagramms in UModel geändert werden..

Anmerkung: Die Funktion "In UML konvertieren" unterstützt nur W3C XML-Schemas.

28.6.13 XML anhand von DB, Excel, EDI in MapForce generieren

Startet Altova MapForce, wenn die Applikation installiert ist. In MapForce können Sie ein Schema auf eine andere DTD, ein anderes XML-Schema oder eine Datenbank mappen und XML generieren.

28.6.14 HTML/PDF/Word-Ausgabe mit StyleVision erstellen

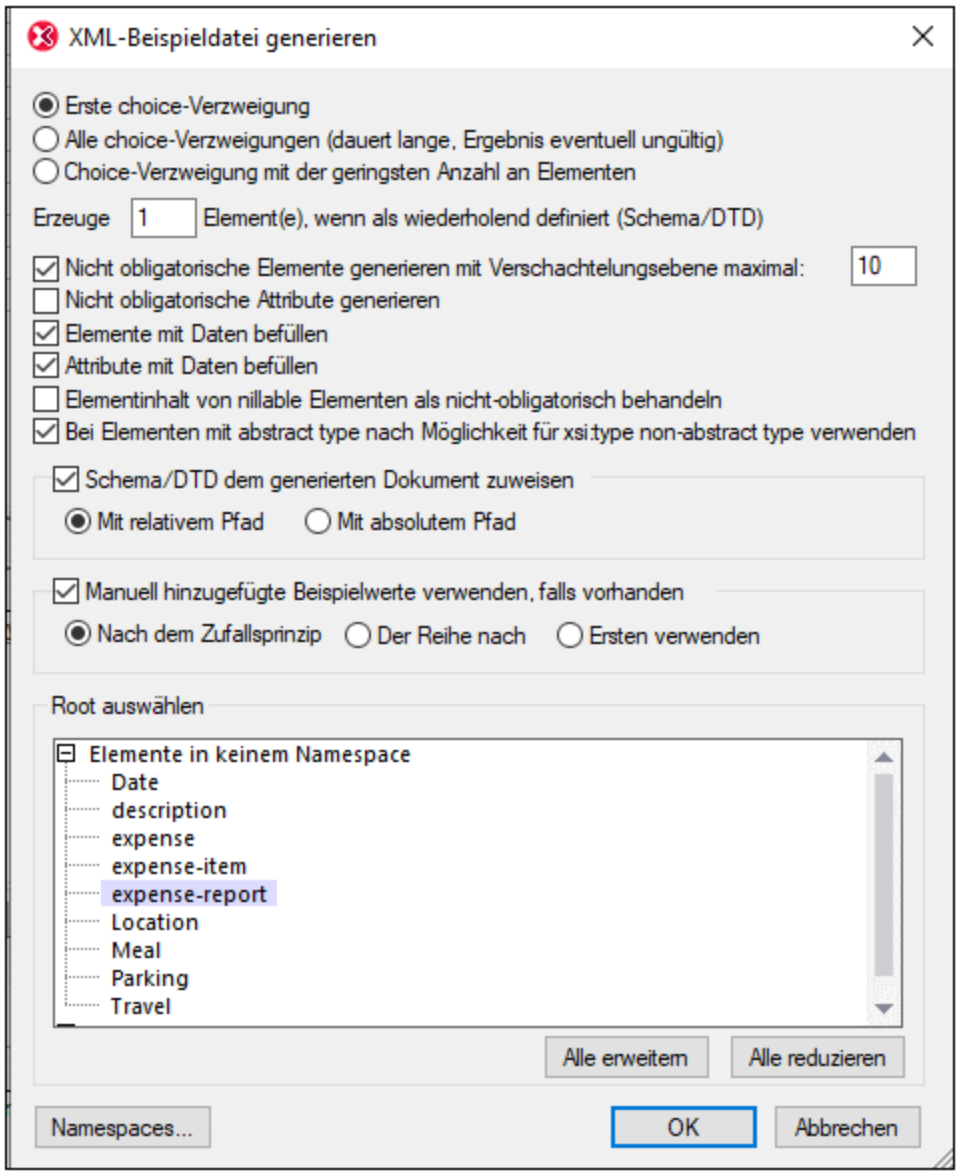
Startet Altova StyleVision, wenn die Applikation installiert ist. In StyleVision können Sie Stylesheets für die Ausgabe von Dokumenten im HTML-, PDF- und RTF-Format erstellen.

28.6.15 XML/JSON-Beispieldatei generieren

Der Befehl **XML/JSON-Beispieldatei generieren** ist in der Textansicht, der Grid-Ansicht und der Schema-Ansicht aktiv und generiert eine XML- oder JSON-Instanz auf Basis der gerade aktiven Schema-Datei. Wenn es sich bei der aktiven Datei um eine DTD oder ein XML-Schema handelt, so wird eine XML-Instanzdatei generiert. Wenn es sich bei der aktiven Datei um ein JSON-Schema handelt, so wird eine JSON-Instanzdatei generiert.

Generieren von XML-Beispieldateien

Wenn eine DTD oder ein XML-Schema aktiv ist, können Sie auf Basis des Schemas eine XML-Beispielinstanz generieren. Bei Auswahl des Befehls **XML/JSON-Beispieldatei generieren** wird das Dialogfeld "XML-Beispieldatei generieren" (*Abbildung unten*) angezeigt, in dem Sie die Optionen für die Generierung der Beispielinstanzdatei festlegen können.



Elemente von choice-Gruppen

Eine Choice-Gruppe ist eine Gruppe von Elementen, aus der nur eines verwendet werden darf. Wenn z.B. ein Element namens `items` laut Definition eine `choice`-Gruppe, bestehend aus den drei Elementen: `cd`, `dvd`, `book` hat, dann kann `items` korrekterweise jedes dieser drei Elemente als Child-Element haben (wobei die Maximalanzahl der zulässigen Elemente im `maxOccurs` Attribut des Elements definiert ist).

Im Dialogfeld "XML-Beispieldatei generieren" können Sie auswählen, ob (i) die erste Verzweigung (das erste Element) der `choice`-Gruppe, (ii) alle Verzweigungen oder (iii) die Verzweigung mit der geringsten Anzahl von Nachfahrenelementen generiert werden soll. Beachten Sie, dass bei Auswahl von *Alle Verzweigungen* ein ungültiges Dokument erzeugt werden könnte, da nur eine Verzweigung aus einer `choice`-Gruppe zulässig ist.

Wenn eine der Verzweigungen der `choice`-Gruppe sich wiederholen darf (d.h., wenn sie einen `maxOccurs` Wert hat, der größer als `1` ist), so wird im ersten Textfeld des Dialogfeld definiert, wie viele der sich wiederholenden Elemente bis zu einer Maximalanzahl von `99` generiert werden sollen. Wenn das `maxOccurs` Attribut der

 © 2016-2022 Altova GmbH

`choice` Gruppe als `unbounded` oder als hohe Zahl definiert ist und im Dialogfeld "XML-Beispieldatei generieren" die Option *Alle Verzweigungen* ausgewählt ist, so wird der `maxOccurs` Wert der `choice`-Gruppe auch durch die Anzahl der im ersten Textfeld des Dialogfelds definierten sich wiederholenden Elemente beschränkt.

Nicht obligatorische Elemente generieren
Beim Aktivieren dieser Option werden alle obligatorischen und optionalen Elemente, die im Schema definiert sind, generiert. Wenn diese Option aktiviert ist, können Sie die gewünschte Verschachtelungsebene definieren. Bei einer tieferen Verschachtelung können nicht obligatorische Elemente bis zur gewünschten Verschachtelungstiefe generiert werden.

Nicht obligatorische Attribute generieren
Beim Aktivieren dieser Option werden sowohl die obligatorischen als auch die optionalen Attribute, die im Schema definiert sind, generiert.

Erzeuge "X" Element(e), wenn als wiederholend definiert (Schema/DTD)
Bei Aktivieren dieser Option wird die im Textfeld eingegebene Anzahl von sich wiederholenden Elementen generiert. Dies wird auf alle Elemente inklusive der Elemente in `choice`-Gruppen angewendet.

Elemente und Attribute mit Daten befüllen
Bei Aktivieren dieser Option werden die Datentyp-Werte der jeweiligen Elemente/Attribute eingefügt. Wenn ein Element z.B. als Datentyp `string` definiert ist, erhält das Element einen Dummy-Wert `string`.

Nillable Elemente und abstrakte Typen
Der Inhalt von nillable Elemente kann als nicht obligatorisch behandelt werden und Elemente mit einem abstrakten Typ können einen nicht abstrakten Typ für ihr `xsi:type Attribut` verwenden.

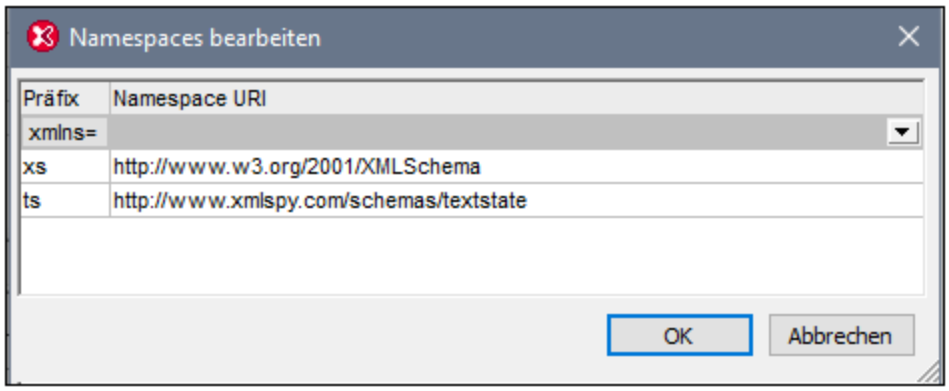
Schemazuweisung für die generierte XML-Datei
Das für die Generierung der XML-Datei verwendete Schema kann der generierten XML-Datei mit einem relativen oder einem absoluten Pfad zugewiesen werden.

Manuell hinzugefügte Beispielwerte verwenden, falls vorhanden
Wenn der Schemakomponente Beispielwerte zugewiesen wurden, so werden diese als Wert bzw. Inhalt dieser Komponente verwendet. Für einzelne Komponenten werden Beispielwerte auf dem Register "Beispiele" über die Eingabehilfe "Facets" 235 zugewiesen. Sie können festlegen, welcher Wert aus den verfügbaren Beispielwerten für die Generierung einer einzigen Datei ausgewählt werden soll:

- Auswahl nach dem Zufallsprinzip.
- Auswahl der einzelnen Beispielwerte der einzelnen Instanzen der Komponente der Reihe nach. Bei jeder Dateigenerierung wird wieder von vorne begonnen.
- immer der erste Wert.

Root-Element
Wenn das Schema mehr als ein globales Element enthält, werden diese aufgelistet und Sie können das für die XML-Beispieldatei benötigte Root-Element aus der Liste auswählen.

Namespaces
Klicken Sie auf die Schaltfläche **Namespaces**, um das Dialogfeld "Namespaces bearbeiten" (*Abbildung unten*) aufzurufen. In diesem Dialogfeld werden die im Schema definierten Namespaces sowie alle Standard-XML-Schema-Namespaces, die in der XML-Beispieldatei erforderlich sind, angezeigt.

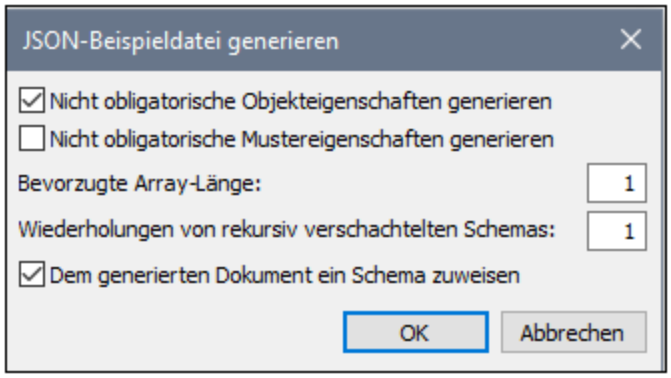


Sie können Folgendes bearbeiten:

- das Namespace-Präfix, das an einen der Dokument-Namespaces gebunden ist. Die in diesem Dialogfeld definierten Namespace-Präfixe werden (in der generierten XML-Datei) verwendet, um Nodes, die sich im entsprechenden Namespace befindet, mit einem Präfix zu versehen. So sehen Sie in der Abbildung etwa, dass Nodes im Namespace `http://www.xmlspy.com/schemas/textstate` in der Beispieldatei mit dem Präfix `ts:` versehen werden.
- Sie können einen der Namespaces des Dokuments als Standard-Namespace (`xmlns=`) definieren, indem Sie in der Auswahlliste `xmlns=` den gewünschten Namespace auswählen. Nodes in dem als Standard-Namespace ausgewählten Namespace werden dadurch ohne ein Namespace-Präfix generiert.

Generieren von JSON-Beispieldateien

Wenn ein JSON-Schema aktiv ist, können Sie auf Basis des JSON-Schemas eine JSON-Beispielinstanz generieren. Bei Auswahl des Befehls wird das Dialogfeld "JSON-Beispieldatei generieren" (*Abbildung unten*) angezeigt, in dem Sie die Optionen für die Generierung der Beispielinstanzdatei festlegen können.

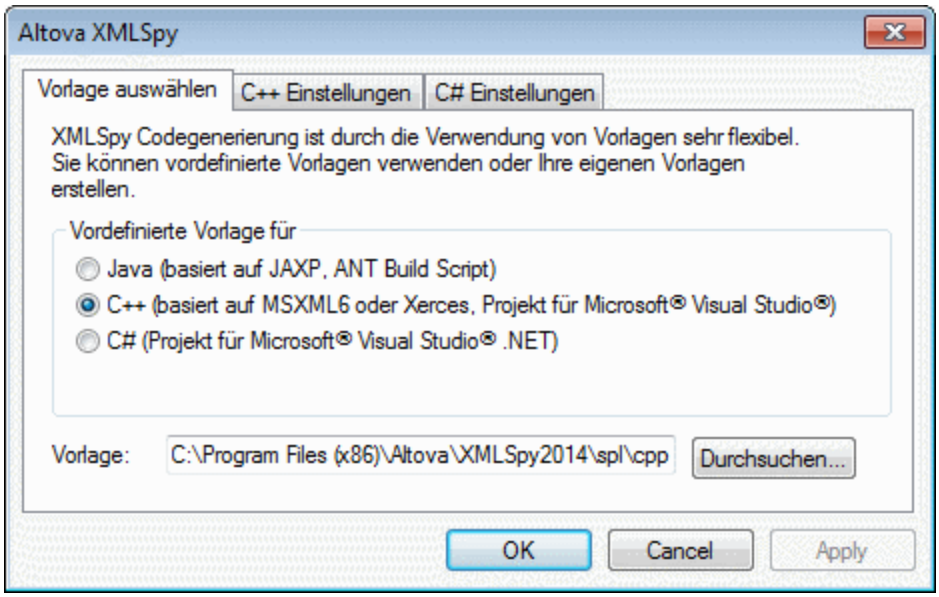


Sie können auswählen, ob nicht obligatorische Objekteigenschaften[737] oder nicht obligatorische Mustereigenschaften[737] generiert werden sollen. Sie können die Länge von Arrays[748] und die Wiederholungen

von rekursiven Definitionen auswählen. Außerdem können Sie festlegen, ob das aktive JSON-Schema der generierten JSON-Beispieldatei automatisch zugewiesen werden soll[765] oder nicht.

28.6.16 Programmcode generieren

Mit diesem Menüpunkt werden automatisch Java-, C++- oder C#-Klassendateien aus Definitionen im aktiven Schema (DTD oder XML-Schema) generiert.



Weitere Informationen dazu erhalten Sie in der Code Generator Dokumentation[1156].

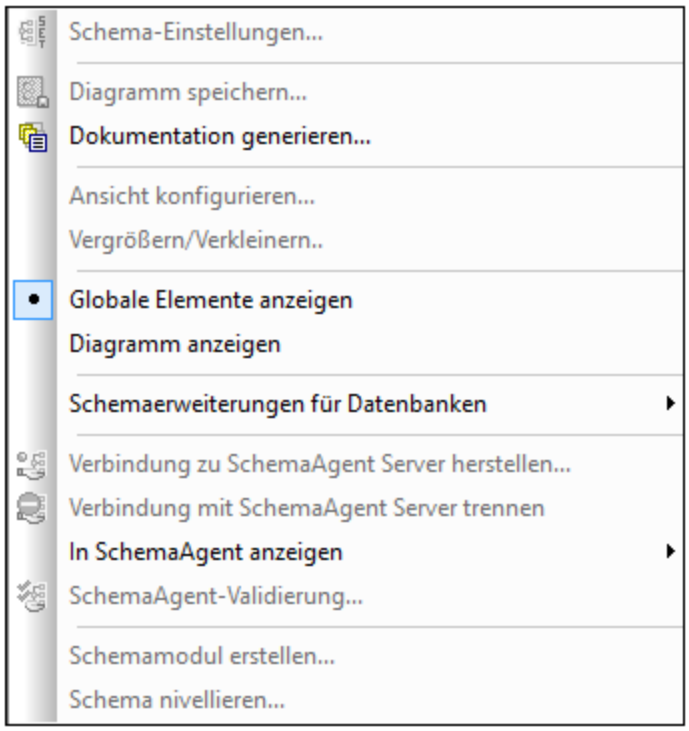
28.6.17 Cache-Speicher leeren

Mit diesem Befehl werden alle zwischengespeicherten Schema- und DTD-Dokumente aus dem Speicher gelöscht. Um Validierung und intelligentes Editieren zu beschleunigen, werden in XMLSpy alle zuletzt verwendeten Schemadokumente und externen geparsten Entities im Zwischenspeicher gespeichert. Informationen aus diesen Dokumenten werden auch mittels des Befehls Gehe zu Definition[1373] angezeigt.

Verwenden Sie diesen Befehl, wenn Sie nicht genügend Speicher in Ihrem System haben, oder wenn Sie zuletzt mit sehr vielen Dokumenten gearbeitet haben, die auf verschiedenen Schemas basieren.

28.7 Menü "Schema-Design"

Im Schema-Design-Menü können Sie die Schema-Design-Ansicht von XMLSpy konfigurieren. In dieser Ansicht können Sie XML-Schemas über eine grafische Benutzeroberfläche erstellen. Das Menü steht dann zur Verfügung, wenn ein XML-Schemadokument in der Schema-Ansicht aktiv ist.



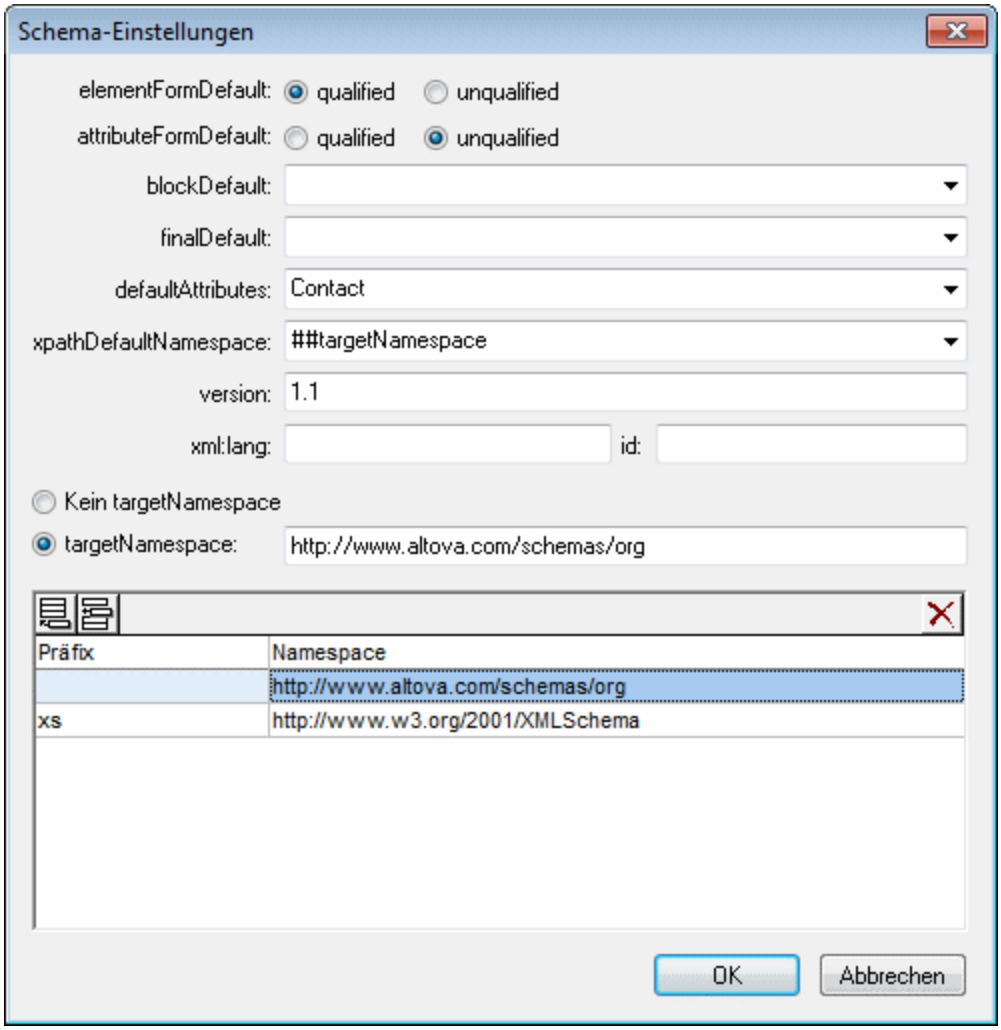
Die Befehle in diesem Menü werden in diesem Abschnitt beschrieben.

28.7.1 Schema-Einstellungen



Der Befehl **Schema-Design | Schema-Einstellungen** ist in der Schema-Ansicht aktiv. Mit diesem Befehl definieren Sie globale Einstellungen für das aktive Schema. Dabei handelt es sich um Attribute für das XML-Schema-Element `xs:schema`.

© 2016-2022 Altova GmbH



Mit den im obigen Dialogfeld "Schema-Einstellungen" (Dialogfeld, wie es im XSD 1.1-Modus angezeigt wird) definierten Einstellungen wird das folgende `xs:schema` Element erstellt.

```
<xs:schema xmlns="http://www.altova.com/schemas/org"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       targetNamespace="http://www.altova.com/schemas/org"
       elementFormDefault="qualified"
       attributeFormDefault="unqualified"
       xpathDefaultNamespace="##targetNamespace"
       version="1.1"
       defaultAttributes="Contact">
```

Anmerkung: Was im Dialogfeld "Schema-Einstellungen" angezeigt wird, hängt davon ab, welcher XSD-Modus aktiv ist. Wenn XSD 1.0 der aktive Modus ist, werden XSD 1.1 Attribute im Dialogfeld nicht angezeigt.

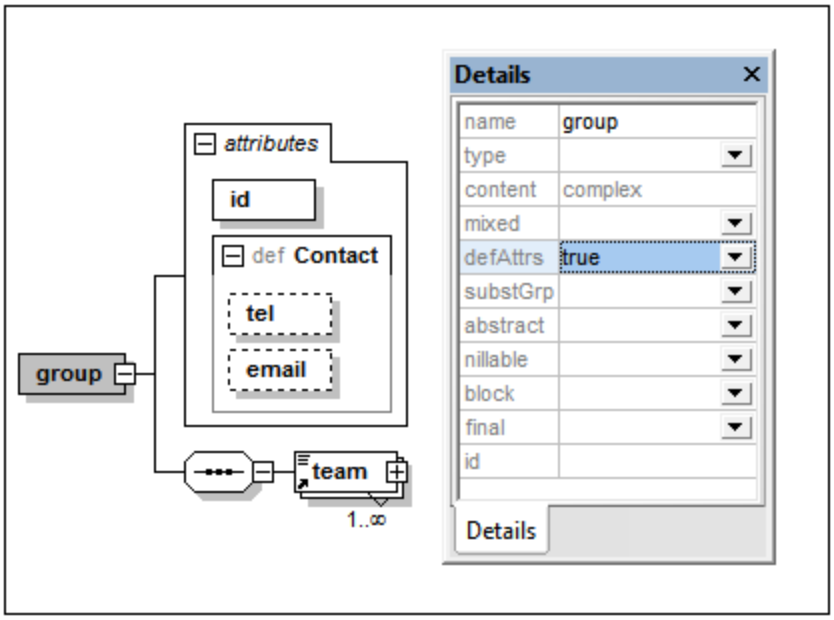
Die Standardattribute `defaultAttributes` und `xpathDefaultNamespace` sind XML Schema 1.1 Features und stehen nur im XSD 1.1-Modus[173] zur Verfügung. Die anderen Attribute stehen in XSD 1.0 und XSD 1.1 zur Verfügung.

Das Attribut version

Das Attribut "version" ist die Dokumentversion. Es handelt sich hierbei nicht um die XSD-Version des Dokuments[173].

Das Attribut defaultAttributes

Mit dem **`defaultAttributes`** Attribut können Sie eine Attributgruppe als Standardattributgruppe für alle complexTypes im Schema auswählen. Die Standardattributgruppe wird im Content Model dieser complexTypes angezeigt. Die Attributgruppe `Contact` in der Abbildung unten ist die Standard-Attributgruppe (*siehe auch Abbildung oben, wo dies definiert ist*) und steht automatisch im Element `group` zur Verfügung. Um die Attributgruppe zu deaktivieren, setzen Sie das Attribut `defaultAttributesApply` des complexType auf `false`. Sie können diese Einstellung auch über die Eigenschaft *defAttrs* in der Eingabehilfe "Details" des complexType vornehmen (*siehe Abbildung unten*).



Das Attribut xpathDefaultNamespace

Das `xpathDefaultNamespace` Attribut definiert den Standard-Namespace für Elemente in XPath-Ausdrücken, die im Schema verwendet werden. Wenn diese Einstellung im Dialogfeld "Schema-Einstellungen" definiert ist, wird das Attribut auf das oberste `xs:schema` Element in der Hierarchie angewendet. Daher gilt die Deklaration für das gesamte Dokument. Sie können die Deklaration in `xs:schema` durch Deklarationen zu Elementen außer Kraft setzen, in denen das Attribut zulässig ist.

- `xs:assert` und `xs:assertion`
- `xs:alternative`

© 2016-2022 Altova GmbH

- `xs:selector` und `xs:field` (in Identity Constraints)

Sie können den XPath-Standard-Namespace in der Eingabehilfe "Details" der oben aufgelisteten Elemente ändern.

Das `xpathDefaultNamespace` Attribut kann einen von drei zulässigen Werten haben:

- `##targetNamespace`: Der XPath-Standard-Namespace ist derselbe wie der Target Namespace des Schemas
- `##defaultNamespace`: Der XPath-Standard-Namespaceist ist derselbe wie der Standard-Namespace des Schemas
- `##local`: Es gibt keinen XPath-Standard-Namespace

Wenn im Dokument kein XPath-Standard-Namespace deklariert ist, befinden sich in XPath-Ausdrücken verwendete Elemente ohne Präfix in keinem Namespace. Die XPath-Standard-Namespace-Deklaration gilt nicht für Attribute.

28.7.2 Diagramm speichern...



Mit diesem Befehl wird das Diagramm des im Hauptfenster angezeigten Content Model im PNG-Format unter dem gewünschten Pfad gespeichert.

28.7.3 Dokumentation generieren



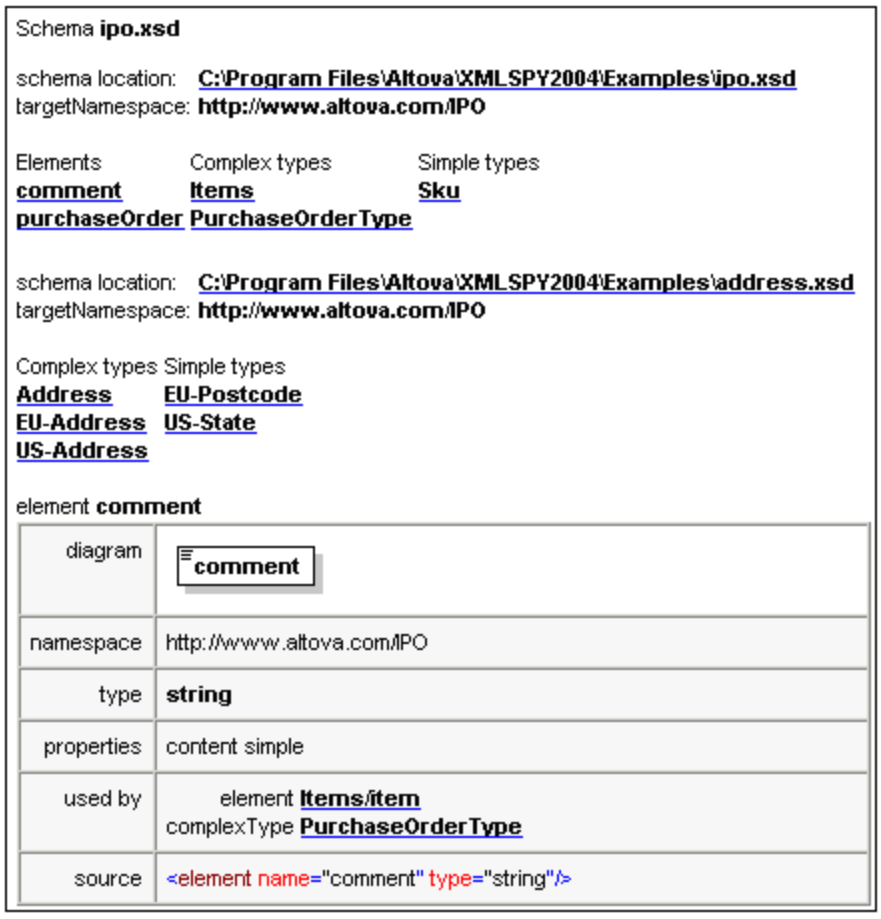
Mit diesem Befehl wird eine detaillierte Dokumentation zu Ihrem XML- oder JSON-Schema (*siehe Abbildung unten*) im HTML-, MS Word-, RTF-, oder PDF-Format erzeugt. Diese Dokumentation kann nach Wunsch geändert und verwendet werden; Sie benötigen dafür keine Genehmigung von Altova. Die Dokumentation wird für diejenigen Komponenten generiert, die Sie im Dialogfeld "(JSON)-Schema-Dokumentation" auswählen (Aufruf des Dialogfelds mit dem Befehl "Dokumentation generieren"). Verwandte Elemente (Child-Elemente, ComplexTypes, usw.) werden in der Regel in der Bildschirmausgabe mit Hyperlinks versehen, sodass Sie zwischen den einzelnen Komponenten navigieren können. Komponenten mit einem Content Model weisen auch Links zur jeweiligen Content Model-Definition auf. Beachten Sie, dass Schema-Dokumentation auch für **inkludierte und importierte Schemakomponenten** generiert wird. Die verschiedenen Optionen für die Generierung von Dokumentation zu XML-Schemas werden im Abschnitt "Dokumentationsoptionen[1391]" beschrieben. Die Optionen zur Generierung von JSON-Schemas sind im Abschnitt Generieren von JSON-Schema-Dokumentation[762] beschrieben.

Beachten Sie, dass die Dokumentationsoptionen[1391] zusätzlich zu den im Dialogfeld Schema-Ansicht konfigurieren[1395] definierten Einstellungen angewendet werden.

Anmerkung: Um Dokumentation im Format MS Word zu generieren, müssen Sie MS Word (Version 2000 oder höher) installiert haben.

Sie können für das generierte Dokument entweder das festgelegte Standard-Design von XMLSpy verwenden oder für das Design ein StyleVision SPS verwenden. Die Verwendung eines StyleVision SPS hat den Vorteil, dass Sie das Design der generierten Dokumentation anpassen sowie als zusätzliches Ausgabeformat PDF generieren können. Wie man mit einem SPS arbeitet, wird in im Abschnitt "Benutzerdefiniertes Design 1393" erklärt.

Anmerkung: Um ein SPS zum Generieren von Schema-Dokumentation zu verwenden muss StyleVision auf Ihrem Computer installiert sein.


Schema **ipo.xsd**

schema location: **C:\Program Files\Altova\XMLSPY2004\Examples\ipo.xsd**
targetNamespace: **http://www.altova.com/IPO**

Elements	Complex types	Simple types
comment	**Items**	**Sku**
purchaseOrder	**PurchaseOrderType**	

schema location: **C:\Program Files\Altova\XMLSPY2004\Examples\address.xsd**
targetNamespace: **http://www.altova.com/IPO**

Complex types	Simple types
Address	**EU-Postcode**
EU-Address	**US-State**
US-Address	

element **comment**

diagram	comment
namespace	http://www.altova.com/IPO
type	**string**
properties	content simple
used by	element **Items/item** complexType **PurchaseOrderType**
source	<element name="comment" type="string"/>

In der oben gezeigten Abbildung sehen Sie die generierte Schema-Dokumentation mit einem Index (alle verwandten Schemas mit ihren globalen Komponenten nach Komponententyp geordnet) am Beginn des Dokuments.

Anmerkung: Bei der Generierung von Dokumentation für W3C-Schemadokumente verwendet XMLSpy applikationsinterne Versionen dieser Dokumente. Aus diesem Grund werden andere Ordner für diese Applikationen nicht berücksichtigt und Redefinitionen und andere Änderungen am Schema werden in der Dokumentation nicht berücksichtigt.

© 2016-2022 Altova GmbH

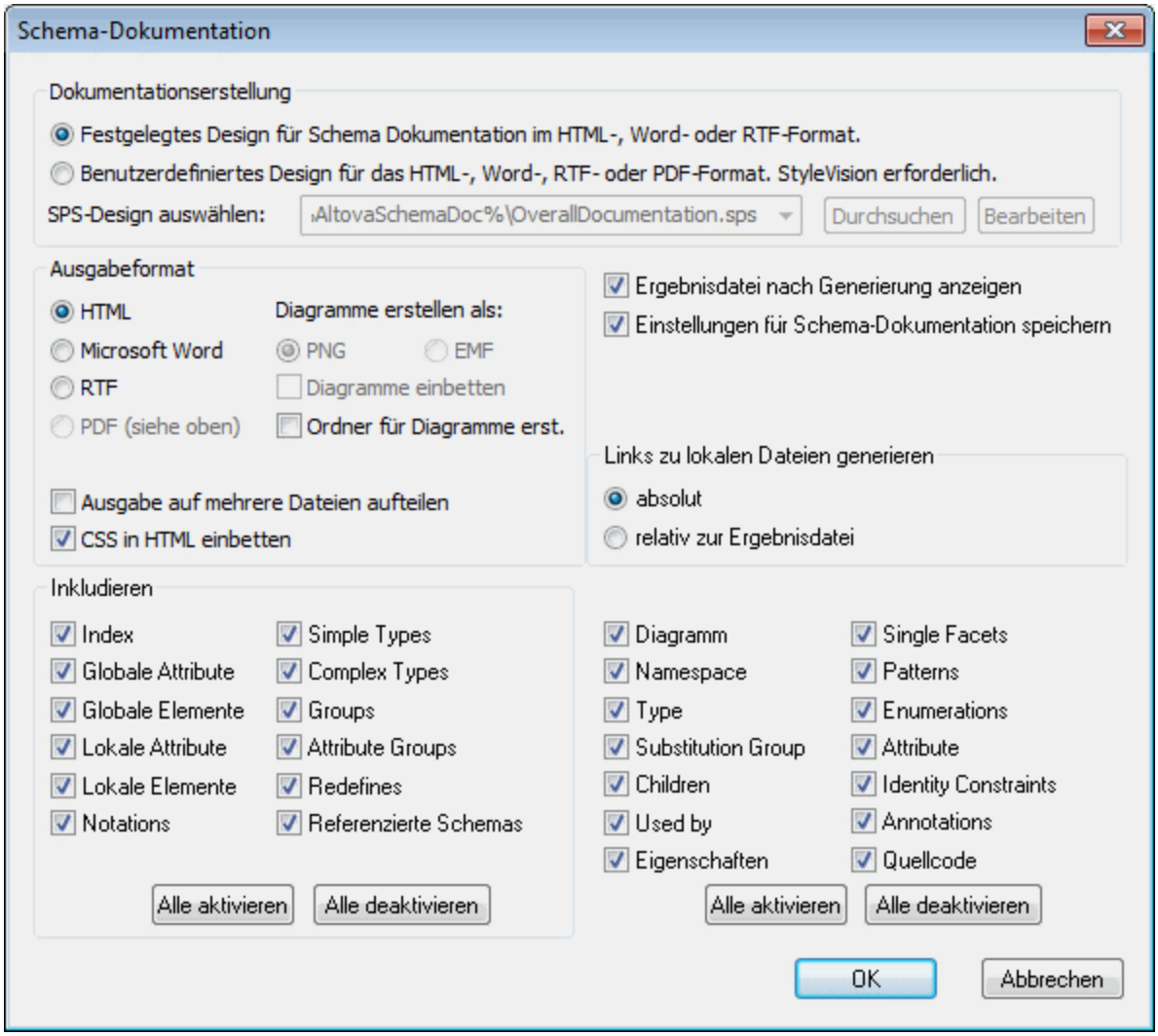
28.7.3.1 Dokumentationsoptionen

Mit dem Befehl **Schema-Design | Dokumentation generieren** wird detaillierte Dokumentation zum aktiven Schema, nämlich einem XML- oder JSON-Schema, generiert. Dieser Abschnitt enthält eine Beschreibung zur Generierung von XML-Schema-Dokumentation. Das Verfahren zur Generierung von JSON-Schema-Dokumentation ist ähnlich. Nähere Informationen zum Generieren von JSON-Schema-Dokumentation und eine Beschreibung der Dokumentationseinstellungen finden Sie im Abschnitt Generieren von JSON-Schema-Dokumentation 762.

Generieren von XML-Schema-Dokumentation

Wenn ein XML-Schema-Dokument aktiv ist und Sie den Befehl **Schema-Design | Dokumentation generieren** auswählen, wird das Dialogfeld "Schema-Dokumentation" geöffnet (*siehe Abbildung unten*), in dem Sie die Optionen für die Dokumentation auswählen können.

Im Bereich "Dokumentationserstellung" des Dialogfelds können Sie wählen, ob Sie für die generierte Dokumentation das festgelegte XMLSpy-Design oder ein in einem StyleVision SPS erstelltes angepasstes Design verwenden möchten. Wählen Sie die gewünschte Option. Beachten Sie, dass PDF-Ausgabe nur für mit einem StyleVision SPS erstellte Dokumentation und nicht für mit einem festgelegten Design generierte Dokumentation zur Verfügung steht. Wie Sie mit einem benutzerdefinierten Design arbeiten, wird im Abschnitt "Benutzerdefiniertes Design" 1393 beschrieben.



Die anderen Optionen im Dialogfeld Schema-Dokumentation werden im Anschluss erklärt:

- Das gewünschte Format wird im Bereich "Ausgabeformat" definiert: entweder HTML, Microsoft Word, RTF oder PDF. (Das Ausgabeformat PDF steht nur zur Verfügung wenn Sie ein StyleVision SPS zur Generierung der Dokumentation verwenden.) Wenn Sie auf OK klicken, werden Sie aufgefordert, den Namen der Ausgabedatei und den Ausgabepfad dafür anzugeben.
- Microsoft Word-Dokumente werden mit der Dateierweiterung `.doc` erstellt, wenn sie mit Hilfe eines festgelegten Designs generiert werden, und mit der Dateierweiterung `.docx`, wenn sie mit Hilfe eines StyleVision SPS generiert werden.
- Die Dokumentation kann entweder als eine einzige Datei generiert oder in mehrere Dateien aufgeteilt werden. Wenn mehrere Dateien generiert werden, entspricht jede Datei einer Komponente. Welche Komponenten in der Ausgabe enthalten sein sollen, wird mittels der Kotrollkästchen im Bereich "Inkludieren" festgelegt. In festgelegten Designs werden automatisch Links zwischen mehreren Dokumenten erstellt.
- Für die HTML-Ausgabe können die CSS-Stildefinitionen entweder in einer separaten CSS-Datei gespeichert oder in der HTML-Datei (im Element `<head>`) eingebettet werden. Wenn eine separate CSS-Datei erstellt wird, erhält sie denselben Namen wie die HTML-Datei, jedoch mit der Erweiterung

 © 2016-2022 Altova GmbH

`.css`. Aktivieren oder deaktivieren Sie das Kontrollkästchen *CSS in HTML einbetten* um die gewünschte Option festzulegen.

- Die Option "Diagramme einbetten" ist für die Ausgabeoptionen MS Word, RTF und PDF aktiv. Wenn diese Option aktiviert ist, werden Diagramme entweder im PNG- oder im EMF-Format in die Ergebnisdatei eingebettet. Andernfalls werden die Diagramme als PNG- oder EMF-Dateien generiert und in der Ergebnisdatei über Objektlinks angezeigt.
- Wenn es sich bei der Ausgabedatei um eine HTML-Datei handelt, werden alle Diagramme als dokumentexterne PNG-Dateien generiert. Wenn das Kontrollkästchen *Ordner für Diagramme erstellen* aktiviert ist, wird im selben Ordner wie die HTML-Datei ein Ordner erstellt, in dem die PNG-Dateien gespeichert werden. Dieser Ordner erhält einen Namen im Format *`HTMLDateiname`*`_diagramme`. Wenn das Kontrollkästchen *Ordner für Diagramme erstellen* deaktiviert ist, werden die PNG-Dateien im selben Ordner wie die HTML-Datei gespeichert.
- Links zu lokalen Dateien (wie z.B. Diagrammbilddateien und externe CSS-Dateien) können relativ oder absolut sein. Wählen Sie im Bereich *Links zu lokalen Dateien generieren* je nach Wunsch das gewünschte Optionsfeld aus.
- Im Bereich **Inkludieren** können Sie auswählen, welche Arten von Datenelementen in der Dokumentation inkludiert werden sollen. Jedes Datenelement der ausgewählten Typen wird in der geniererten Dokumentation angezeigt. Wenn z.B. das Kontrollkästchen von *Lokale Attribute* aktiviert ist, so wird die Beschreibung jedes einzelnen lokalen Attributs als separater Eintrag angezeigt. Bei Auswahl der Option "Index" werden alle verwandten Schemas mit ihren globalen Komponenten nach Komponententyp geordnet am Anfang der Datei aufgelistet. Über die Schaltflächen **Alle aktivieren** und **Alle deaktivieren** können Sie alle Optionen im Bereich einfach aktivieren bzw. deaktivieren. Beachten Sie, dass die Option **Inkludieren** keine Auswirkung auf die Anzeige eines Datenelementtyps innerhalb der grafischen Definitionen hat. Diese Anzeige ist von den Einstellungen abhängig, die Sie im Dialogfeld Schema-Ansicht konfigurieren[1395] vornehmen. Wenn Sie also die Attribute in der grafischen Darstellung eines Schema-Datenelements ausblenden möchten, so deaktivieren Sie die Option "Attribute" im Dialogfeld Schema-Ansicht konfigurieren[1395].
- Im Bereich **Details** werden die Einzelheiten aufgelistet, die für die einzelnen Komponenten inkludiert werden können. Wählen Sie die gewünschten Informationen aus. Über die Schaltflächen **Alle aktivieren** und **Alle deaktivieren** können Sie alle Optionen im Bereich einfach aktivieren bzw. deaktivieren.
- Die Option "Ergebnisdatei nach der Generierung anzeigen" ist für alle Ausgabeoptionen aktiv. Wenn diese Option aktiviert ist, wird die erzeugte Datei in der Browser-Ansicht (HTML-Ausgabe), MS Word (MS Word Ausgabe) und den Standardapplikationen für `.rtf`-Dateien (RTF-Ausgabe) und `.pdf`-Dateien (PDF-Ausgabe) angezeigt.

Parameterwerte

Wenn das StyleVision SPS einen oder mehrere Parameterdefinitionen enthält, wird beim Klicken auf **OK** ein Dialogfeld aufgerufen, in dem alle Parameter aufgelistet werden, die im SPS definiert sind. Sie können in diesem Dialogfeld Parameterwerte eingeben um die Standardwerte zu überschreiben, die im SPS zugeordnet worden sind.

28.7.3.2 Benutzerdefiniertes Design

Sie können anstatt des festgelegten Standard XMLSpy Designs ein angepasstes Design für die Schema-Dokumentation erstellen. Das angepasste Design wird in einem StyleVision SPS erstellt, das eine Design-Vorlage für das Ausgabedokument darstellt.

Erstellen des SPS

Ein StyleVision Power Stylesheet (oder SPS) wird mit Hilfe des StyleVision Produkts von Altova erstellt. Einem SPS zur Generierung von Schema-Dokumentation muss ein XML-Schema zugrunde liegen, das die Struktur der Schema-Dokumentation festlegt. Dieses Schema heißt `SchemaDocumentation.xsd` und wird mit Ihrem XMLSpy Paket mitgeliefert. Es wird im Verzeichnis `C:\Dokumente und Einstellungen\<Benutzername>\Eigene Dateien\Altova\XMLSpy2023\Documentation\Schema`. gespeichert.

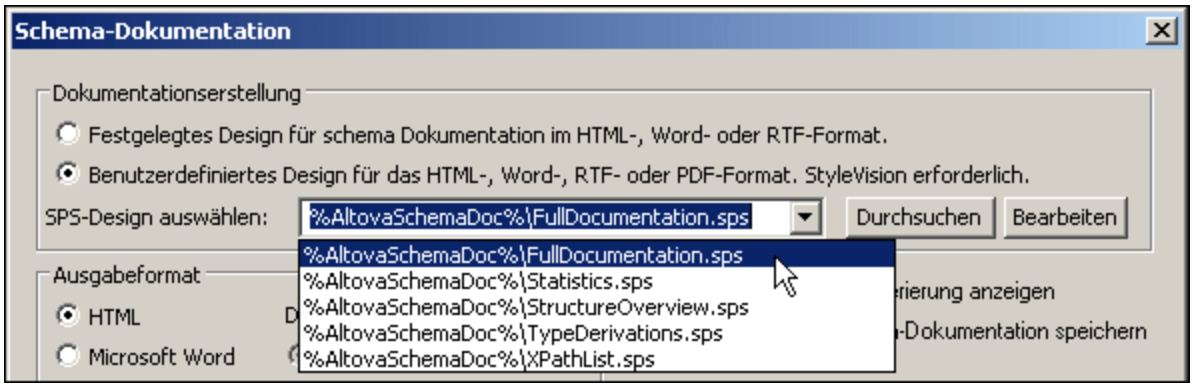
Beim Erstellen des SPS Designs in StyleVision werden Nodes aus dem Schema `SchemaDocumentation.xsd` in die Design-Vorlage eingefügt und Formatvorlagen und Eigenschaften zugewiesen. Zusätzliche Komponenten wie Links, Tabellen und Bilder können ebenfalls zum SPS hinzugefügt werden. Auf diese Weise kann die gesamte Ausgabe im SPS gestaltet werden. Wie man ein SPS Design in StyleVision erstellt, wird im Benutzerhandbuch von StyleVision im Detail beschrieben.

Die Verwendung eines SPS zur Generierung von Schema-Dokumentation hat den Vorteil, dass Sie die vollständige Kontrolle über das Design der Schema-Dokumentation haben. Beachten Sie auch, dass die PDF-Ausgabe der Schema-Dokumentation nur dann verfügbar ist, wenn ein benutzerdefiniertes SPS verwendet wird; PDF-Ausgabe steht nicht zur Verfügung, wenn das festgelegte XMLSpy Design verwendet wird.

Festlegen des für die Schema-Dokumentation verwendete SPS

Nachdem ein SPS erstellt worden ist, kann es zum Generieren von Schema-Dokumentation verwendet werden. Das SPS, das Sie für die Generierung der Schema-Dokumentation verwenden möchten, wird im Dialogfeld "Schema-Dokumentation" (aufgerufen über den Befehl **Schema-Design | Dokumentation generieren...**) ausgewählt. Wählen Sie im Bereich "Dokumentationserstellung" dieses Dialogfelds (*siehe Abbildung unten*) die Optionsschaltfläche *Beutzerdefiniertes Design für das HTML-, Word-, RTF- oder PDF-Format* aus. Sie können dann auf die Schaltfläche **Durchsuchen** klicken und das gewünschte SPS im Dateisystem auswählen. Klicken Sie im Dialogfeld auf **OK** und geben Sie im Dialogfeld "Speichern", das erscheint, das Verzeichnis und den Namen der Ausgabedatei ein.

Anmerkung: Der Pfad zum Schema, auf dem die SPS-Datei basiert, (`WSDLDocumentation.xsd` or `WSDL20Documentation.xsd` (siehe oben) muss im SPS korrekt angegeben sein:



Die folgenden editierbaren SPS-Designs für die Schema-Dokumentation werden mit XMLSpy geliefert. Sie befinden sich im Unterverzeichnis `Altova\XMLSpy2023\Documentation\Schema\` des Verzeichnisses "(Eigene) Dokumente" 34:

- `OverallDocumentation.sps`, mit dem die vollständige Dokumentation über das Schema generiert wird
- `Statistics.sps`, das die Anzahl der globalen und lokalen Elemente, Attribute und Attributgruppen, sowie Simple Types und Complex Types für das Hauptschema und für jede Schemadatei separat auflistet
- `StructureOverview.sps`, mit der eine Struktur der globalen Elemente und Complex Types bis zu einer einstellbaren Tiefe ausgegeben wird
- `TypeDerivations.sps`, das Simple Types und Complex Types sowie alle ihre direkt und indirekt abgeleiteten Typen in Baumform auflistet
- `XPathGenerator.sps`, das alle möglichen XPath-Statements bis zu einer einstellbaren Tiefe generiert

Diese Dateien sowie alle anderen SPS-Dateien, die Sie kürzlich aufgerufen haben, stehen in der Dropdown-Liste der Option Benutzerdefiniertes Design zur Verfügung (*siehe Abbildung oben*).

Durch Klicken auf die Schaltfläche **Bearbeiten** im Bereich "Dokumentationserstellung" wird StyleVision gestartet und das ausgewählte SPS in einem StyleVision Fenster geöffnet. Um eine Vorschau auf das resultierende Dokument in StyleVision zu erhalten, benötigen Sie eine XML-Arbeitsdatei. Den oben aufgelisteten SPS-Designs wurde bereits eine XML-Beispieldatei namens `Sample.xml` zugewiesen, die sich im folgenden Unterordner des Verzeichnisses "(Eigene) Dokumente" 34 befindet:

```
Altova\XMLSpy2023\Documentation\Schema\SampleData
```

Anmerkung: Um ein SPS zum Generieren von Schema-Dokumentation verwenden zu können, muss StyleVision auf Ihrem Computer installiert sein.

28.7.4 Ansicht konfigurieren

Mit diesem Befehl, der in der Content Model-Ansicht aktiv ist, können Sie die Anzeige des Content Model konfigurieren. Wenn Sie auf den Befehl klicken, wird rechts unten im XMLSpy-Fenster das Dialogfeld "Schema-Ansicht konfigurieren" geöffnet. Hier sehen Sie bereits bei der Eingabe in das Dialogfeld, wie sich Ihre Einstellungen auswirken. Sobald Sie auf OK klicken, werden die Einstellungen aktiv und für das Content Model aller XML-Schemadateien, die in der Folge geöffnet werden, übernommen. Diese Einstellungen werden auch für die erzeugte Schema-Dokumentation 1389 und die Druckausgabe übernommen. Wenn Sie also die Attribute in der grafischen Darstellung eines Schema-Datenelements ausblenden möchten, so deaktivieren Sie die Option "Attribute" im Dialogfeld Schema-Ansicht konfigurieren 1395.

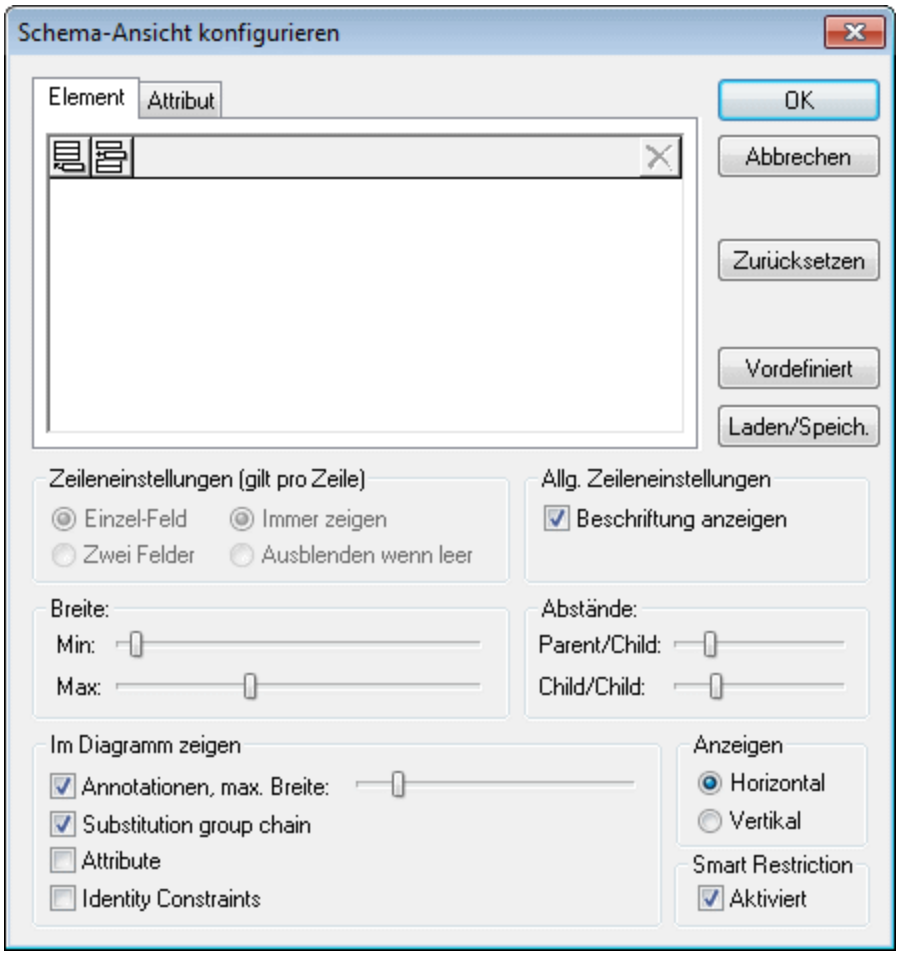
Anmerkung: Eine Beschreibung zum Konfigurieren der JSON-Schema-Design-Ansicht finden Sie im Abschnitt Konfigurieren der JSON-Schema-Design-Ansicht 761.

Definieren von Eigenschaftsdeskriptorzeilen für das Content Model

Sie können festlegen, welche Element- und Attributeigenschaften in der Content Model-Ansicht angezeigt werden sollen. Diese Eigenschaften werden als Grid-Zeilen in Komponentenkästchen angezeigt. Um Eigenschaftsdeskriptorzeilen zu definieren, gehen Sie vor wie folgt:

1. Wählen Sie **Schema-Design | Ansicht konfigurieren**. Daraufhin wird das Dialogfeld "Schema-Ansicht konfigurieren" angezeigt.
2. Klicken Sie auf dem **Element**- oder **Attribut**-Register auf das Symbol "Anhängen"  oder "Einfügen" , um eine Eigenschaftsdeskriptorzeile hinzuzufügen. Die Zeile wird im Dialogfeld und zu den Elementkästchen in der Content Model-Ansicht hinzugefügt.

3. Wählen Sie aus der Dropdown-Liste die gewünschte Eigenschaft aus. *Siehe Abbildung unten.*
4. Wiederholen Sie Schritt 2 und 3 für alle benötigten Eigenschaften.



Die Content Model-Ansicht wird aktualisiert und zeigt die definierten Eigenschaftsdeskriptorzeilen für alle Elemente an, für die sie existieren.

Anmerkung: Bei Attributen wird die von Ihnen definierte Konfiguration nur angezeigt, wenn Attribute (im Gegensatz zur Anzeige der Attribute in einem Bereich unterhalb der Content Model-Ansicht) im Diagramm angezeigt werden. Die konfigurierte Ansicht wird auf alle Content Model-Ansichten angewandt, die nach Definition der Ansicht, geöffnet werden.

Löschen einer Eigenschaftsdeskriptorzeile aus der Content Model-Ansicht

Um einzelne Eigenschaftsdeskriptorzeilen im Dialogfeld "Schema-Ansicht konfigurieren" zu löschen, wählen Sie die betreffende Zeile aus und klicken Sie auf das Symbol "Löschen" .

Einstellungen zur Konfiguration der Content Model-Ansicht

Die Content Model-Ansicht kann mit Hilfe von Einstellungen im Dialogfeld " Schema-Ansicht konfigurieren" konfiguriert werden. Oben wurde beschrieben, wie Sie definieren, welche Eigenschaftsdeskriptorzeilen in der Content Model-Ansicht angezeigt werden. Im Folgenden werden die anderen Einstellungen beschrieben:

© 2016-2022 Altova GmbH

Zeileneinstellungen (gilt pro Zeile)
Sie können festlegen, ob eine Eigenschaftsdeskriptorzeile Einzel- oder Doppel-Inhalte enthalten soll und ob Zeilen für jedes einzelne Element oder nur für Elemente, die diese Eigenschaft enthalten, angezeigt werden sollen. Aktivieren Sie das entsprechende Optionsfeld für die gewünschte Einstellung. Beachten Sie, dass diese beiden Einstellungen separat für einzelne Zeilen definiert werden können (markieren Sie die gewünschte Zeile und legen Sie die Einstellung fest).

Allgemeine Zeileneinstellungen
Mit dieser Option blenden Sie die Zeilenbeschreibungen (d.h. den Namen der Eigenschaft) ein bzw. aus.

Breite
Mit Hilfe dieser Schieberegler können Sie die Mindest- und Maximalgröße der Elementrechtecke in der Content Model-Ansicht einstellen. Ändern Sie die Größe, wenn der beschreibende Text nicht zur Gänze sichtbar ist oder wenn Sie die Anzeige standardisieren möchten.

Abstände
Mit den Schiebern können Sie den horizontalen und vertikalen Abstand zwischen den Elemente am Bildschirm verändern.

Im Diagramm anzeigen
Das Kontrollkästchen Annotationen erlaubt es, den Annotationstext ein- oder auszublenden und die Textbreite des Annotationstextes zu verändern. Weiters können Sie die Anzeige der Substitutionsgruppen ein- oder ausblenden. Wenn Sie diese Kontrollkästchen aktivieren, werden die Attribute und Identity Constraints im Content Model-Diagramm angezeigt; andernfalls werden Sie am unteren Rand des Content Model-Fensters als Register angezeigt.

Zeichenrichtung
Diese Optionen definieren die Orientierung des Elementdiagramms am Bildschirm. Wählen Sie zwischen horizontal und vertikal.

Editieren des Content Model im Diagramm selbst
Sie können Elementparameter- und Werte direkt im Inhaltsmodell ändern. Doppelklicken Sie auf die Eigenschaft, die Sie ändern wollen, und geben Sie einen neuen Wert ein. Wenn mehrere Werte zur Auswahl stehen, öffnet sich das Drop-down Listenfeld, ansonsten geben Sie den gewünschten Wert ein und bestätigen Sie mit der Eingabetaste.

Schaltflächen im Dialogfeld "Schema-Ansicht konfigurieren"
- Die Schaltfläche **Laden/Speichern** lädt und speichert die von Ihnen getätigten Einstellungen.
- Die Schaltfläche **Vordefiniert** setzt die Konfiguration der Ansicht auf die Werkseinstellungen zurück.
- Die Schaltfläche **Zurücksetzen** löscht alle Listenfeld-Einträge.

Aktivieren von Smart Restrictions

Um Smart Restrictions[244] zu aktivieren, aktivieren Sie das Kontrollkästchen für Smart Restrictions.

28.7.5 Vergrößern/Verkleinern

Mit diesem Befehl stellen Sie den Zoomfaktor der Ansicht des Content Model ein. Diese Funktion eignet sich zum Verkleinern eines großen Content Model, sodass das gesamte Content Model im Hauptfenster zu sehen ist. Der Zoomfaktor beträgt zwischen 10 % und 200 % der tatsächlichen Größe.



Ziehen Sie den Schieber um den Zoomfaktor zu verändern, oder geben Sie im Textfeld den gewünschten Zoomfaktor ein.

28.7.6 Globale Elemente anzeigen

Mit diesem Befehl wechseln Sie von der Content Model-Ansicht[190] zur Schema-Übersicht[177], um alle globalen Komponenten im Schema anzuzeigen. Es ist entweder dieser Befehl oder der Befehl "Diagramm anzeigen" aktiv. Der jeweils aktive Befehl wird durch ein Häkchen auf der linken Seite angezeigt. (*siehe Abbildung unten).*



Alternativ dazu können Sie auch auf das Symbol **Globale Elemente anzeigen**  am oberen Rand des Content Model klicken, um zur Schema-Übersicht zurückzukehren.

28.7.7 Diagramm anzeigen

Mit diesem Befehl zeigen Sie die Content Model-Ansicht[190] der ausgewählten globalen Komponente an - falls die ausgewählte globale Komponente ein Content Model hat. Globale Komponenten mit einem Content Model (ComplexTypes, Elemente und Element-Gruppen) werden mit dem Symbol  links davon gekennzeichnet. Es ist entweder dieser Befehl oder der Befehl "Globale Elemente anzeigen" aktiv. Der jeweils aktive Befehl wird durch ein Häkchen auf der linken Seite angezeigt. (*siehe Abbildung unten).*



Alternativ dazu haben Sie folgende Möglichkeiten, um zur Content Model-Ansicht zu wechseln:

- Klicken Sie auf das Symbol  neben der Komponente, die Sie anzeigen möchten, oder

 © 2016-2022 Altova GmbH

- Doppelklicken Sie auf den Namen der Komponente im Komponenten-Navigator (oberste rechte Eingabehilfe).

28.7.8 Schemaerweiterungen für Datenbanken

Dieser Menübefehl enthält ein Untermenü mit den Befehlen für Oracle und MS SQL Server-Schemaerweiterungen.

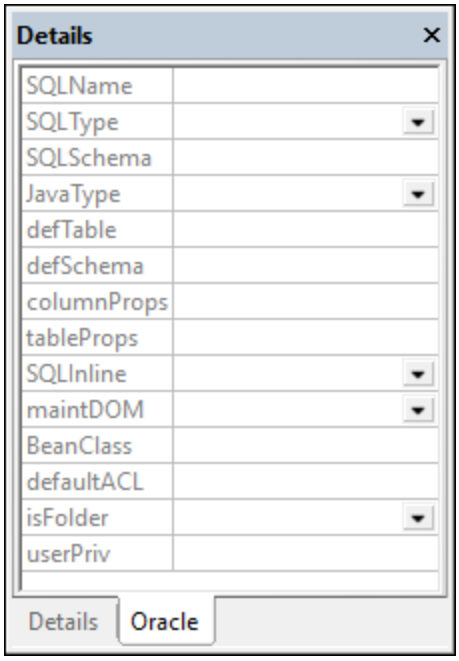
- Oracle Schema-Extensions aktivieren 1399
- Oracle Schema-Einstellungen 1400
- Microsoft SQL Server Schema Extensions aktivieren 1401
- Named Schema Relationships 1402
- Unnamed Element Relationships 1402

28.7.8.1 Oracle Schema Extensions aktivieren

XMLSpy unterstützt Oracle Schema Extensions zur Verwendung in Oracle 9i Project XDB. Mit Hilfe dieser Extensions können Sie definieren und konfigurieren, wie Oracle 9i Project XDB XML-Dokumente intern speichert. Diese XML-Dokumente können dann über SQL-Abfragen und Legacy Tools aufgerufen werden. Nähere Informationen dazu finden Sie auf der Oracle Website.

Bei Auswahl des Befehls **Oracle Schema Extensions aktivieren** geschieht Folgendes:

- Der XDB Namespace wird am `schema`-Element deklariert: `xmlns:xdb="http://xmlns.oracle.com/xdb"`.
- In der Eingabehilfe "Details" wird ein Register "Oracle" angelegt, über das Sie Attribute - einschließlich XDB-spezifischer Attribute - zu Schema-Elementen wie `xsd:complexType` und `xsd:element` hinzufügen können.

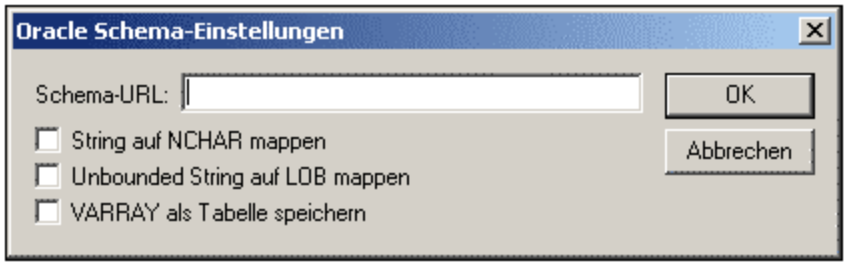


Oracle Extensions können für complexTypes, Elemente und Attribute definiert werden. Verwenden Sie die Eingabehilfe genau wie in XMLSpy.

Anmerkung: Sie können diesen Menübefehl aktivieren oder deaktivieren, d.h. Sie können Extensions ein- oder ausschalten. Bei aktivierten Oracle Extensions wird links neben dem Befehl ein Häkchen angezeigt. Bei Deaktivierung der Extensions werden die XDB Namespace-Deklaration und alle XDB Extensions in der Datei gelöscht. Sie werden durch eine Meldung darauf hingewiesen, dass diese Informationen unwiderruflich gelöscht werden.

28.7.8.2 Oracle Schema-Einstellungen

Mit diesem Befehl können Sie globale Einstellungen für Oracle Schemaerweiterungen vornehmen.



Diese Menüoption kann nur aufgerufen werden, wenn Sie zuvor Oracle Schema Extensions aktiviert haben (Menüoption "Oracle Schema Extensions aktivieren[1399]").

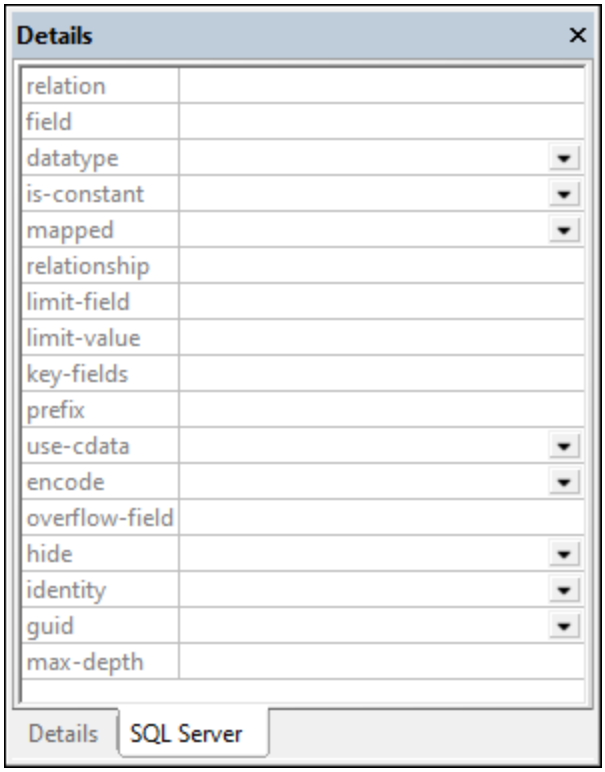
 © 2016-2022 Altova GmbH

28.7.8.3 Microsoft SQL Server Schema Extensions aktivieren

XMLSpy unterstützt Microsoft SQL Server 2000 Schema Extensions zur Verwendung mit Microsoft SQL Server. Durch Verwendung dieser Extensions können Sie definieren und konfigurieren, wie Microsoft SQL Server Ihre XML-Dokumente intern speichert. Diese XML-Dokumente können dann über SQL-Abfragen und Legacy Tools aufgerufen werden. Weitere Informationen zu diesem Thema finden Sie auf der Microsoft Website.

Bei Auswahl des Befehls **Microsoft SQL Server Extensions aktivieren** geschieht Folgendes:

- Der SQL Server Namespace wird am `schema` Element deklariert: `xmlns:sql="urn:schemas-microsoft-com:mapping-schema"`.
- In der Eingabehilfe "Details" wird ein Register "SQL Server" angelegt, über das Sie Attribute zu Schema-Elementen wie `xsd:element` hinzufügen können.



Wenn SQL Server Extensions für eine Schema-Komponente definiert werden können, steht das Register "SQL Server" in der Eingabelhilfe "Details" zur Verfügung, wenn die Komponente ausgewählt ist. Verwenden Sie die Eingabehilfe genau wie in XMLSpy.

Anmerkung: Sie können diesen Menübefehl aktivieren oder deaktivieren, d.h. Sie können Extensions ein- oder ausschalten. Bei aktivierten Oracle Extensions wird links neben dem Befehl ein Häkchen angezeigt. Bei Deaktivierung von SQL Server Extensions werden die SQL Server Namespace-Deklaration und alle SQL Extensions in der Datei gelöscht. Sie werden durch eine Meldung darauf hingewiesen, dass diese Informationen unwiderruflich gelöscht werden.

28.7.8.4 Named Schema Relationships

Mit diesem Befehl können Sie "named relationships" definieren, die benötigt werden, um das Erstellen von Dokumenthierarchien zu erlauben.
Diese Menüoption kann nur aufgerufen werden, wenn Sie zuvor SQL Server Schema Extensions aktiviert haben (Menüoption "SQL Server Schema Extensions aktivieren").

So erstellen Sie Named Schema Relationships:

1. Klicken Sie auf das Symbol "Einfügen"  oder "Anhängen" , um im Dialogfeld eine neue Zeile einzufügen.
2. Klicken Sie in das neue Feld, um den Namen der Beziehung einzugeben.
3. Mit OK bestätigen Sie Ihre Eingaben.



Es wird ein SQL Relationship Element erzeugt, das unter der Namespace Deklaration eingefügt wird.

Bitte beachten Sie: Klicken Sie auf das Symbol "Löschen"  um eine Zeile aus dem Dialogfeld zu löschen.

28.7.8.5 Unnamed Element Relationships

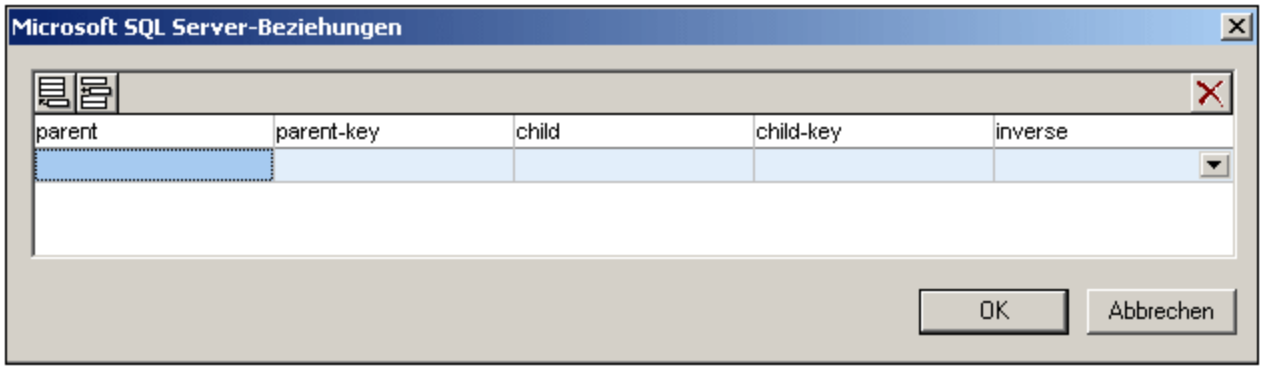
Mit diesem Befehl können Sie Unnamed Relationships definieren, um Informationen für das Erstellen von Dokumenthierarchien bereitstellen zu können. Diese Menüoption kann nur aufgerufen werden, wenn Sie zuvor SQL Server Schema Extensions aktiviert haben (Menüoption **Microsoft SQL Server Schema Extensions aktivieren**).

So erstellen Sie Unnamed Schema Relationships:

1. Klicken Sie auf das Symbol "Einfügen"  oder "Anhängen" , um im Dialogfeld eine neue Zeile einzufügen.
2. Klicken Sie in das Feld, um den entsprechenden Namen einzugeben.

© 2016-2022 Altova GmbH

3. Mit OK bestätigen Sie Ihre Eingaben.



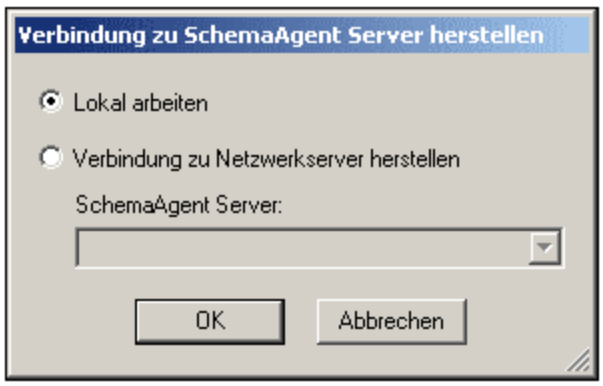
Hierdurch wird ein SQL Relationship Element für das ausgewählte Schemaelement erzeugt.

Bitte beachten Sie: Klicken Sie auf das Symbol "Löschen"  um eine Zeile aus dem Dialogfeld zu löschen.

28.7.9 Verbindung zu SchemaAgent Server herstellen



Dieser Befehl ist aktiv, wenn ein XML-Schema-Dokument aktiv ist. Sie können damit eine Verbindung zu einem SchemaAgent Server herstellen. Dies ist jedoch nur möglich, wenn auf Ihrem Rechner ein lizenzierter Altova SchemaAgent 2023 installiert ist. Wenn Sie auf diesen Befehl klicken, wird das Dialogfeld "Verbindung zu SchemaAgent herstellen" (*siehe Screenshot unten*) angezeigt:



Sie können entweder mit dem lokalen Server arbeiten (dem SchemaAgent, Server, der mit Altova SchemaAgent 2023 mitgeliefert wird). oder mit einem Netzwerk-Server (dem Altova SchemaAgent Server 2023, der kostenlos erhältlich ist). Wenn Sie die Option "Lokal arbeiten" auswählen, wird der lokale Server von SchemaAgent gestartet, wenn Sie auf OK klicken und es wird eine Verbindung mit diesem Server hergestellt.

Wenn Sie die Option "Verbindung zu Netzwerk-Server herstellen" wählen, muss der ausgewählte SchemaAgent Server ausgeführt werden, damit eine Verbindung hergestellt werden kann.

Wenn Sie eine Verbindung zu SchemaAgent Server hergestellt haben, fungiert XMLSpy als SchemaAgent Client. Sie haben darin leistungsstarke Funktionen zum Editieren und Verwalten von Schemas zur Verfügung. Nähere Informationen zu SchemaAgent, zur Installation von SchemaAgent Server und zum Herstellen einer Verbindung mit SchemaAgent Server finden Sie im Abschnitt "DTD und XML-Schema" dieses Benutzerhandbuchs unter SchemaAgent 470. Nähere Informationen über die Installation der beiden Produkte und die Verwendung finden Sie im SchemaAgent-Benutzerhandbuch zu diesen Produkten.

Nachdem Sie eine Verbindung mit SchemaAgent Server hergestellt haben, erscheint in der Leiste am oberen Rand des Hauptfensters eine Meldung mit Informationen über die Verbindung. Sie haben nun uneingeschränkten Zugriff auf alle Schemas und Schema-Komponenten im/in den Suchpfad(en) (Ordner(n)), die für den SchemaAgent Server definiert wurden, mit dem XMLSpy verbunden ist.

Bitte beachten Sie: Damit eine Verbindung zustande kommt, müssen Sie den Altova SchemaAgent Client auf Ihrem Rechner installiert haben und eine gültige Lizenz dafür besitzen.

28.7.10 Verbindung mit SchemaAgent Server trennen



Dieser Befehl steht zur Verfügung, wenn eine Verbindung zu einem SchemaAgent Server zustande gekommen ist. Bei Auswahl dieses Befehls wird die Verbindung zwischen XMLSpy und dem SchemaAgent Server getrennt.

28.7.11 In SchemaAgent anzeigen

Mit diesem Befehl werden das aktive Schema und optional damit verknüpfte Schemas im Altova-Produkt SchemaAgent 2023 angezeigt. (Dieses Produkt muss auf demselben Rechner wie XMLSpy installiert sein, wenn Sie die Funktionen von SchemaAgent verwenden möchten). Das/die Schema(s) wird/werden in einem neuen SchemaAgent Design in SchemaAgent geöffnet.

Wenn Sie den Mauszeiger über den Menübefehl **In SchemaAgent anzeigen** halten, erscheint ein Untermenü, das Optionen enthält, welche Schemas in SchemaAgent angezeigt werden sollen. Nähere Informationen zu diesen Optionen finden Sie in dieser Dokumentation im Abschnitt "DTD und XML-Schema" unter SchemaAgent 479.

© 2016-2022 Altova GmbH

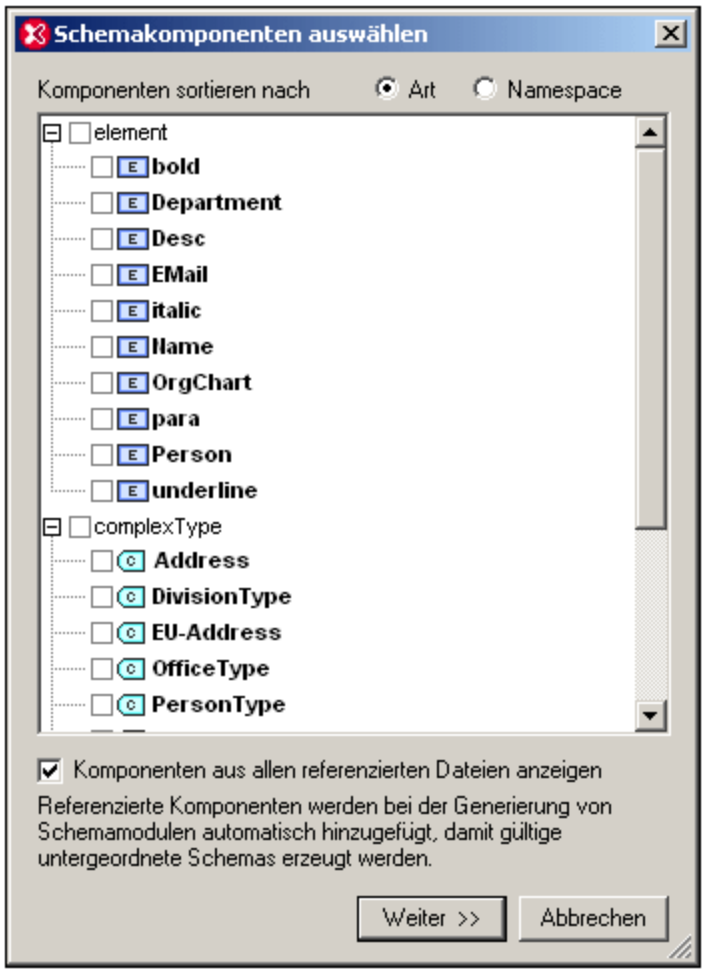
28.7.12 SchemaAgent-Validierung



Mit diesem Befehl können Sie das aktuell aktive Schema und die damit verwandten Schemas validieren. Eine Beschreibung zu dieser Funktion finden Sie in dieser Dokumentation im Abschnitt "Schema-Ansicht" unter SchemaAgent-Validierung[479].

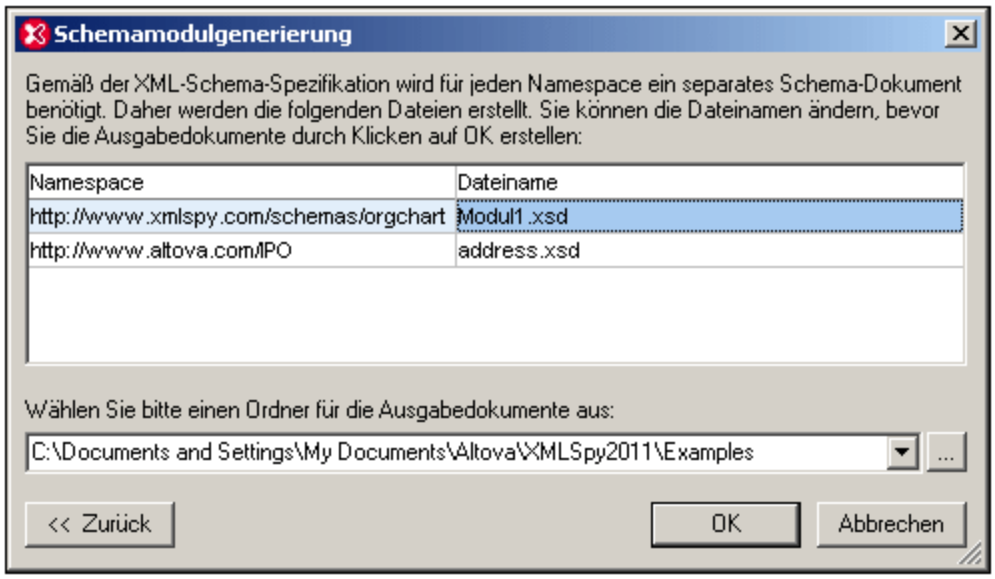
28.7.13 Schemamodul erstellen

Mit dem Befehle **Schemamodul erstellen** wird das Dialogfeld "Schemakomponenten auswählen" (*Abbildung unten*) aufgerufen. Wählen Sie im Dialogfeld die Komponente bzw. Komponenten aus, die als einzelne Schemamodule angelegt werden sollen und klicken Sie auf **Weiter**. (Beachten Sie das Kontrollkästchen unterhalb des Fensters, über das auch Komponenten aus allen referenzierten Dateien für die Auswahl aufgelistet werden können.)



Geben Sie im Dialogfeld "Schemamodulgenerierung" (*Abbildung unten*), das nun angezeigt wird, den/die Namen ein, den/die die einzelnen Dateien der Schemamodulpakete erhalten sollen. Außerdem müssen Sie den Ordner angeben, in dem die neuen Schemamoduldateien gespeichert werden sollen. Ein Schemamodulpaket könnte mehrere Dateien enthalten, wenn es sich bei einer oder mehreren der erstellten Komponenten im Originalschema um importierte Komponenten handelt. Für jeden Namespace im Schemamodul wird eine eigene Schemadatei angelegt. Die Dateinamen, die im Dialogfeld angezeigt werden, sind standardmäßig die Namen der Originaldateien. Da Sie die Originaldateien aber nicht überschreiben dürfen, verwenden Sie neue Dateinamen, wenn Sie die Dateien im selben Ordner wie die Originaldateien speichern möchten.

© 2016-2022 Altova GmbH

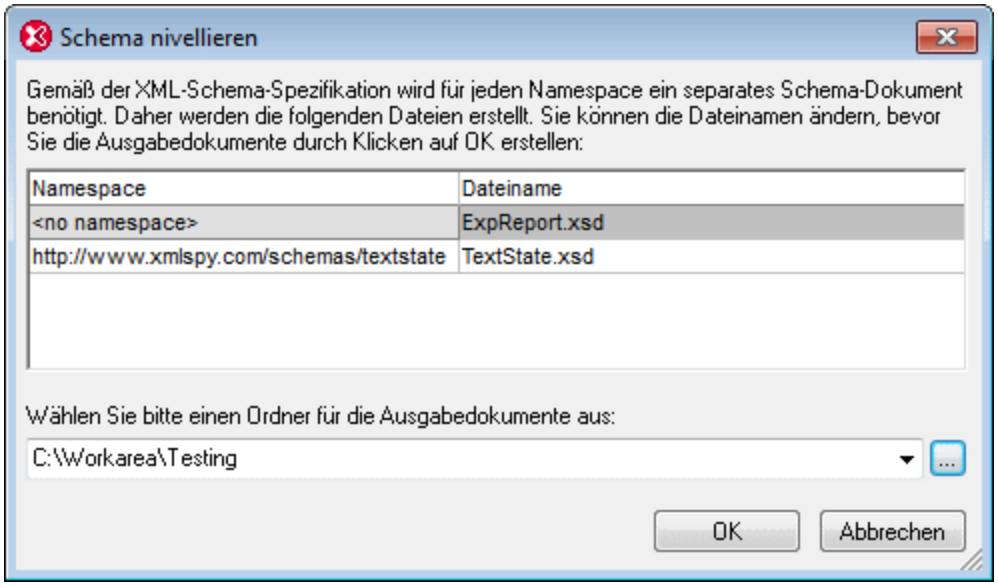


Wenn Sie auf **OK** klicken, wird die Schemamoduldatei mit dem Namespace, der dem der aktiven Datei entspricht, in der Schema-Ansicht geöffnet. Alle anderen Dateien im Paket werden erstellt, aber nicht in der Schema-Ansicht geöffnet.

28.7.14 Schema nivellieren

Beim Nivellieren des aktiven Schemas in der Schema-Ansicht (i) werden die Komponenten aller inkludierten Schemas als globale Komponenten des aktiven Schemas hinzugefügt und (ii) die inkludierten Schemas gelöscht.

Um das aktive Schema zu nivellieren, wählen Sie den Befehl **Schema-Design | Schema nivellieren**. Daraufhin wird das Dialogfeld "Schema nivellieren" (*Abbildung unten*) angezeigt, in dem die Namen separater Dateien - eine für jeden Namespace, der im nivellierten Schema enthalten sein wird - aufgelistet sind. Diese Standardnamen sind mit den Namen der Originaldateien identisch. Da die Originaldateien aber nicht überschrieben werden dürfen, müssen die Dateinamen geändert werden, wenn Sie die Dateien im selben Ordner wie die aktive Datei speichern möchten. Sie können zu einem Ordner navigieren, in dem das nivellierte Schema und die damit verknüpften Dateien gespeichert werden sollen.



Wenn Sie auf **OK** klicken, wird die nivellierte Schema-Datei in der Schema-Ansicht geöffnet.

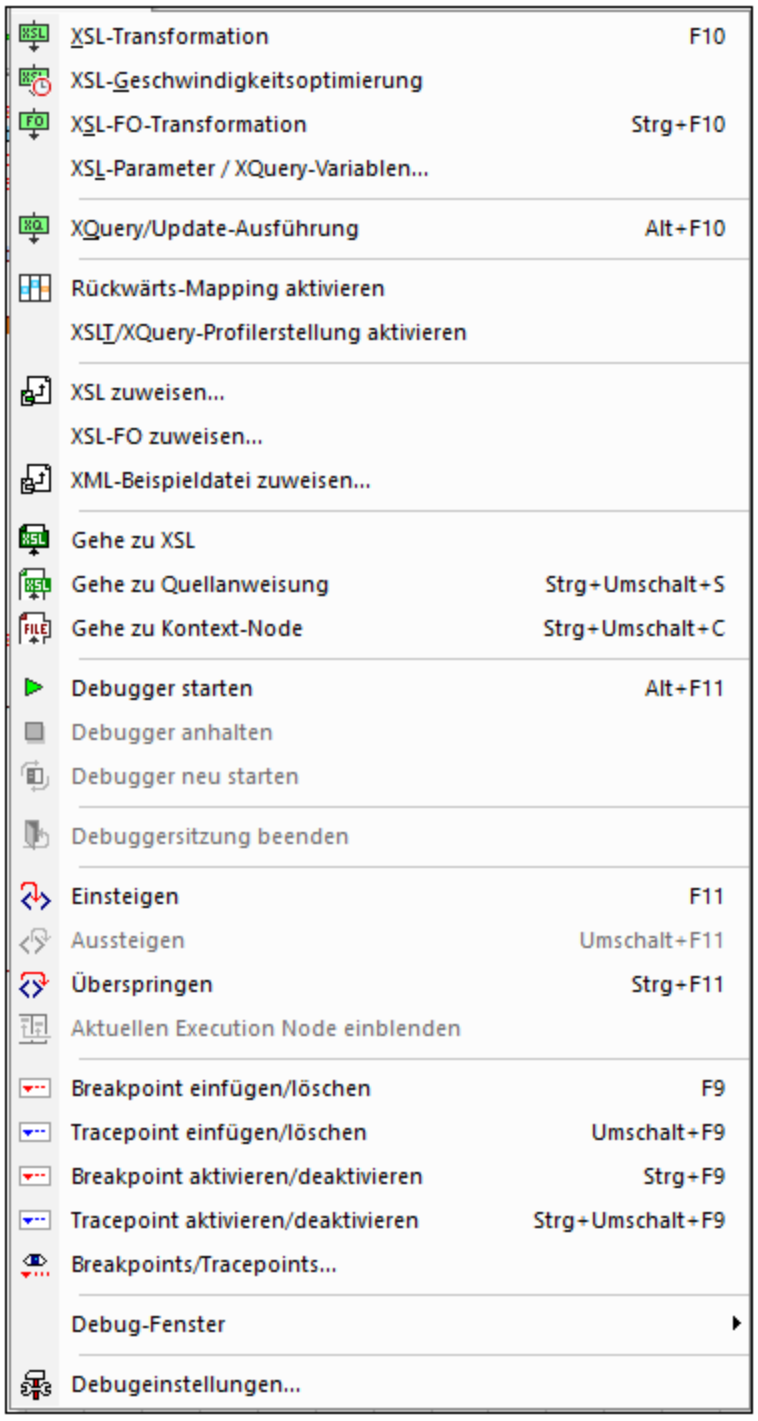
© 2016-2022 Altova GmbH

28.8 Menü "XSL/XQuery"

Mit Hilfe der XSL Transformationssprache können Sie festlegen, wie ein XML-Dokument in andere XML-Dokumente oder Textdateien konvertiert werden soll. Eine Art von XML-Dokument, die mittels eines XSLT-Dokuments generiert wird, ist ein FO-Dokument, das anschließend weiter in ein PDF-Ausgabedokument konvertiert wird.
XMLSpy enthält integrierte XSLT-Prozessoren (für XSLT 1.0, XSLT 2.0 und XSLT 3.0) und kann mit einem FO-Prozessor auf Ihrem System verknüpft werden, um XML-Dateien zu transformieren und verschiedene Arten von Ausgabedokumenten zu generieren. Der Pfad zum FO-Prozessor muss im Dialogfeld "Optionen" im Abschnitt XSL (Extras | Optionen[1633]) angegeben werden, um direkt innerhalb der XMLSpy Benutzeroberfläche verwendet werden zu können.

Des Weiteren verfügt XMLSpy über einen integrierten XQuery-Prozessor, mit Hilfe dessen Sie XQuery-Dokumente (mit oder ohne Referenz auf ein XML-Dokument) ausführen können.

Die Befehle für die oben beschriebenen Transformationen finden Sie im Menü **XSL/XQuery**. Zusätzlich dazu enthält dieses Menü auch Befehle für die Arbeit mit dem Altova XSLT/XQuery Debugger.



 © 2016-2022 Altova GmbH

28.8.1 XSL-Transformation



F10

Mit diesem Befehl führen Sie mittels eines zugewiesenen XML-Dokuments eine **XSL-Transformation** durch. Die Transformationen kann entweder mit Hilfe des entsprechenden Altova XSLT-Prozessors (Altova XSLT 1.0-Prozessor für XSLT 1.0 Stylesheets; Altova XSLT 2.0-Prozessor für XSLT 2.0 Stylesheets; Altova XSLT 3.0-Prozessor für XSLT 3.0 Stylesheets), mit Hilfe des MSXML-Moduls von Microsoft oder eines externen XSLT-Prozessors durchgeführt werden. Der in Verbindung mit diesem Befehl verwendete Prozessor, wird im Dialogfeld "Optionen" (**Extras | Optionen**) auf dem Register XSL[1633] festgelegt.

Sollte Ihr XML-Dokument eine Referenz zu einem XSLT Stylesheet enthalten, wird dieses Stylesheet automatisch für die XSLT Transformation verwendet. (Ein XSLT Stylesheet kann über den Befehl XSL zuweisen[1420] zugewiesen werden), Wenn das XML-Dokument Teil eines Projekts ist, können Sie im Dialogfeld "Projekteigenschaften[1344]" ein XSLT Stylesheet auch für einzelne Ordner definieren. Klicken Sie mit der rechten Maustaste auf den/die Projektordner oder die zu transformierenden Datei(en) und wählen Sie XSL-Transformation. Wurde einer XML-Datei kein XSLT Stylesheet zugewiesen, werden Sie aufgefordert das XSLT Stylesheet anzugeben. Sie können eine Datei auch über eine globale Ressource, eine URL (klicken Sie dazu auf die Schaltfläche **Durchsuchen)**[1280] oder eines der offenen Fenster in XMLSpy auswählen (Klicken Sie dazu auf die Schaltfläche **Fenster**).

Automatisieren der Validierung mit RaptorXML 2023

Altova RaptorXML ist eine Applikation zur XML-Validierung, XSLT-Transformation und XQuery-Ausführung. Sie kann über die Befehlszeile, über eine COM-Schnittstelle, in Java-Programmen und in .NET-Applikationen aufgerufen werden. So können mit RaptorXML z.B. Aufgaben wie die XSLT-Transformation automatisiert werden. Sie können z.B. eine Batch-Datei erstellen, die RaptorXML aufruft, um damit eine Reihe von Dokumenten zu transformieren. Nähere Informationen dazu finden Sie in der Dokumentation zu RaptorXML.

Transformationen in ZIP-Dateien

Damit die Ausgabe in Form einer ZIP-Datei, z.B. einer Open Office XML-Datei (OOXML) wie `.docx` erfolgt, muss im Dateipfad der Ausgabedatei das ZIP-Protokoll definiert werden. Zum Beispiel:

```
dateiname.zip|zip/dateiname.xxx

dateiname.docx|zip/dateiname.xxx
```

Anmerkung: Vor Ausführung der Transformation muss eventuell die Verzeichnisstruktur erstellt werden. Wenn Sie Dateien für ein Open Office XML-Archiv generieren, müssen Sie die Archivdateien zippen, um OOXML-Datei der obersten Ebene zu erstellen (z.B. `.docx`).

28.8.2 XSL-Geschwindigkeitsoptimierung



Der Befehl **XSL-Geschwindigkeitsoptimierung** ist aktiv, wenn ein XSLT- oder XML-Dokument aktiv ist. Der Befehl startet die XSL-Geschwindigkeitsoptimierung, bei der analysiert wird, ob anhand des verwendeten XSLT Stylesheets schnellere Transformationen durchgeführt werden könnten. Bei der Optimierung wird das XSLT Stylesheet an einer Gruppe von XML-Daten (einem oder mehreren XML-Dokumenten) ausgeführt. Dabei wird die Leistung des Stylesheet analysiert. Anhand dieser Analyse wird eine Optimierungstrategie ermittelt, die im XSLT Stylesheet (in Form einer Verarbeitungsanweisung am Ende des Stylesheet) gespeichert werden kann. Mit Hilfe des optimierten Stylesheet können Transformationen in Zukunft schneller durchgeführt werden.

Je nachdem, ob das XSLT oder das XML-Dokument aktiv ist, werden Sie aufgefordert, ein XML-Dokument bzw. ein XSLT Stylesheet auszuwählen. Wenn Sie auf **OK** klicken, wird die Analyse gestartet. Wenn das XSLT- bzw. XML-Dokument bereits eine XML-1421 bzw. XSLT-Zuweisung1420 im Dokument hat, wird dieser Schritt übersprungen und die Analyse beginnt sofort. Nähere Informationen zur Verwendung der Optimierungsfunktion finden Sie im Abschnitt XSL-Geschwindigkeitsoptimierung512. Die Einstellungen für die Optimierung können im Dialogeld "Optionen" auf dem Register "XSL-Geschwindigkeitsoptimierung"1633 (**Extras | Optionen**) vorgenommen werden.

28.8.3 XSL-FO-Transformation



Strg+F10

FO ist ein XML-Format zur Beschreibung von seitenweise formatierten Dokumenten. Ein FO-Prozessor wie z.B. der FOP-Prozessor von Apache XML Projects verarbeitet eine FO-Datei und generiert anhand dieser Datei ein PDF-Dokument. Die Generierung eines PDF-Dokuments erfolgt also in zwei Schritten.

1. Das XML-Dokument wird mittels eines XSLT Stylesheet in ein FO-Dokument transformiert.
2. Aus dem FO-Dokument wird von einem FO-Prozessor ein PDF-Dokument (oder ein anderes Ausgabeformat) generiert.

Mit dem Befehl **XSL:FO-Transformation** wird ein XML-Dokument oder ein FO-Dokument ins PDF-Format transformiert.

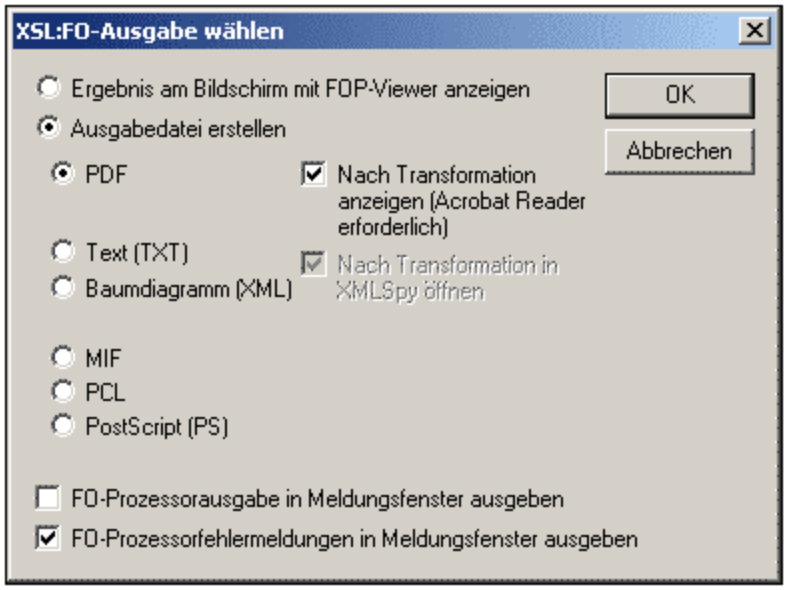
- Wird die **XSL:FO-Transformation** an einem XML-Quelldokument durchgeführt, werden beide oben beschriebenen Schritte nacheinander durchgeführt. Wenn das für die Transformation in FO benötigte XSLT Stylesheet im XML-Dokument nicht referenziert ist, werden Sie aufgefordert, eines für die Transformation zuzuweisen. Sie können eine Datei auch über eine globale Ressource, eine URL (klicken Sie dazu auf die Schaltfläche **Durchsuchen)**1280 oder eines der offenen Fenster in XMLSpy auswählen (Klicken Sie dazu auf die Schaltfläche **Fenster**). Die Transformation von XML in XSL-FO wird durch den XSLT-Prozessor durchgeführt, der im Dialogfeld "Optionen" (**Extras | Optionen**) auf dem Register XSL1633 definiert wurde. Standardmäßig ist der integrierte XSLT-Prozessor von XMLSpy als XSLT-Prozessor ausgewählt. Das erzeugte FO-Dokument wird von dem FO-Prozessor, der im Dialogfeld "Optionen" (**Extras | Optionen**), Register XSL1633 definiert wurde, direkt verarbeitet.

© 2016-2022 Altova GmbH

- Wird der Befehl **XSL:FO-Transformation** an einem FO-Dokument ausgeführt, wird das Dokument mit dem FO-Prozessor, der im Dialogfeld "Optionen" (**Extras | Optionen**), Register XSL[1633] definiert wurde, verarbeitet.

XSL:FO-Transformationsausgabe

Wenn Sie den Befehl **XSL:FO-Transformation** aufrufen, wird das Dialogfeld "XSL:FO-Ausgabe wählen" angezeigt (*siehe Abbildung unten*). (Wenn dem aktiven XML-Dokument kein XSLT Stylesheet zugewiesen wurde, werden Sie zuerst nach der XSLT-Datei gefragt).



Sie können das vom FO-Prozessor erzeugte Dokument mit Hilfe des FOP-Viewers direkt am Bildschirm anzeigen oder Sie können eine Ausgabedatei in einem der folgenden Formate erzeugen: PDF, Text, als XML-Baumdiagramm, MIF PCL oder PostScript. Sie können auch Meldungen von FO-Prozessor aktivieren, damit (i) die Standard-Ausgabemeldung des Prozessors im Fenster "Meldungen" angezeigt wird und (ii) die Fehlermeldungen des Prozessors im Fenster "Meldungen" angezeigt werden. Um eine der beiden Optionen zu aktivieren, klicken Sie auf das entsprechende Kontrollkästchen um unteren Rand des Dialogfelds.

Hinweis:

- Falls Sie die Option zur Installation des FOP-Prozessors des Apache XML-Projekts nicht deaktiviert haben, wurde dieser im Ordner `C:\ProgramData\Altova\SharedBetweenVersions` installiert. In diesem Fall wurde der Pfad zum Prozessor automatisch auf dem XSL-Register[1633] des Dialogfelds "Optionen" (**Extras | Optionen**) als zu verwendender FO-Prozessor eingegeben. Sie können den Pfad zu jedem beliebigen FO-Prozessor, den Sie verwenden möchten, definieren.
- Sie können den Befehl XSL:FO-Transformation nicht nur an der aktiven Datei im Hauptfenster anwenden, sondern auch an jeder Datei/jedem beliebigen Ordner die/den Sie im Projektordner auswählen. Klicken Sie dazu mit der rechten Maustaste und wählen Sie den Befehl **XSL:FO-Transformation**. Es wird das XSLT Stylesheet verwendet, das dem ausgewählten Projektordner zugewiesen wurde.

28.8.4 XSL-Parameter / XQuery-Variablen

Mit dem Befehl **XSL-Parameter/XQuery-Variablen** öffnen Sie das Dialogfeld "XSLT-Eingabeparameter/externe XQuey-Variablen" (*siehe Screenshot*). Sie können den Namen und die dazugehörigen Werte von einem oder mehreren Parametern eingeben, die Sie an das XSLT Stylesheet übergeben möchten, oder den Namen und die dazugehörigen Werte von einer oder mehreren XQuery-Variablen, die an das XQuery-Dokument übergeben werden sollen. Diese Parameter werden in XMLSpy folgendermaßen verwendet:

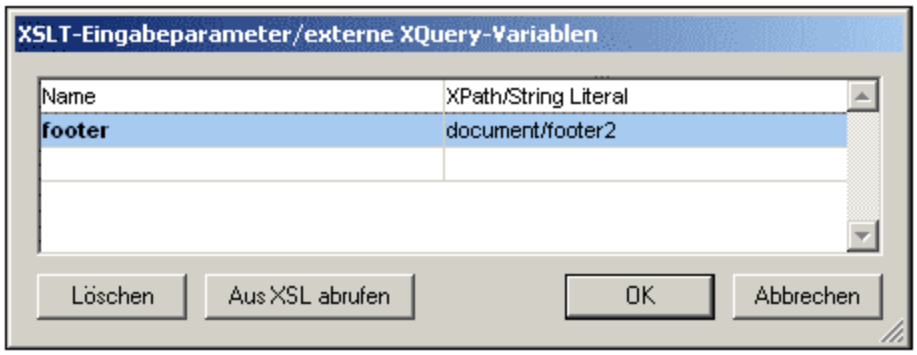
- Wenn ein XML-Dokument über den Befehl **XSL-Transformation** aus dem Menü "XSL/XQuery" transformiert wird, werden die derzeit im Dialogfeld gespeicherten Parameterwerte an das ausgewählte XSLT-Dokument übergeben und für die Transformation verwendet.
- Wenn ein XQuery-Dokument über den Befehl **XQuery-Ausführung** aus dem Menü "XSL/XQuery" verarbeitet wird, werden die aktuell im Dialogfeld gespeicherten externen XQuery-Variablen zur Ausführung an das XQuery-Dokument übergeben.

Anmerkung: Parameter oder Variablen, die Sie im Dialogfeld "XSLT-Eingabeparameter/externe XQuery-Variablen" eingeben, werden nur an den integrierten Altova XSLT-Prozessor übergeben. Wenn Sie daher MSXML oder einen anderen externen Prozessor verwenden, den Sie konfiguriert haben, werden diese Parameter nicht an diesen Prozessor übergeben.

Anmerkung: Es handelt sich nicht um einen Fehler, wenn ein XSLT-Parameter oder eine externe XQuery-Variable im Dialogfeld "XSLT-Eingabeparameter/externe XQuery-Variablen" definiert, aber im XSLT/XQuery-Dokument oder der Transformation nicht verwendet wird.

Verwendung von XSLT-Parametern

Bei dem Wert, den Sie für den Parameter eingeben, kann es sich um einen XPath-Ausdruck ohne Anführungszeichen oder einen von Anführungszeichen eingeschlossenen Textstring handeln. Wenn das aktive Dokument ein XSLT-Dokument ist, ist die Schaltfläche **Aus XSL abrufen** aktiv. Wenn Sie darauf klicken, werden die im XSLT-Dokument deklarierten Parameter zusammen mit ihren Standardwerten in das Dialogfeld eingefügt. Auf diese Art können Sie deklarierte Parameter schnell inkludieren und deren Standardwerte gegebenenfalls ändern.



Anmerkung: Sobald Sie im Dialogfeld die Parameterwerte eingegeben haben, werden diese für alle zukünftigen Transformationen verwendet, bis sie ausdrücklich gelöscht werden oder die Applikation neu gestartet wird. Parameter, die im Dialogfeld eingegeben wurden, werden für diese Sitzung auf Applikationsebene definiert und werden von diesem Zeitpunkt an bei jeder Transformation, die über die IDE durchgeführt wird, an das jeweilige XSLT-Dokument übergeben. Dies bedeutet:

© 2016-2022 Altova GmbH

- Parameter sind nicht mit einem bestimmten Dokument verknüpft und
- Alle Parameter, die im Dialogfeld eingegeben wurden, werden gelöscht, sobald die Applikation (XMLSpy) geschlossen wird.

Beispiel für die Verwendung von XSLT-Parametern

Im folgenden Beispiel haben wir ein XML-Dokument, das die Namen von Ländern und deren jeweiliger Hauptstadt enthält:

```
<document>
   <countries>
      <country name="USA" capital="Washington DC"/>
      <country name="UK" capital="London"/>
      <country name="France" capital="Paris"/>
      <country name="Russia" capital="Moscow"/>
      <country name="China" capital="Beijing"/>
   </countries>
</document>
```

Das folgende XSLT-Dokument generiert ein XML-Dokument, in dem ein Land aus der XML-Datei zusammen mit der Hauptstadt des jeweiligen Landes angezeigt wird. Das Land wird durch Auswahl seines Names als Wert des Parameters namens **country** (*unten gelb markiert*) ausgewählt.

```
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
   <xsl:param name="country" select="'USA'"/>
   <xsl:template match="countries">
      <xsl:for-each select="country[@name=$country]">
         <country>
            <name><xsl:value-of select="$country"/></name>
            <capital><xsl:value-of select="@capital"/></capital>
         </country>
      </xsl:for-each>
   </xsl:template>
</xsl:stylesheet>
```

Wenn dieses XSLT-Dokument an oben aufgelisteten XML-Dokument ausgeführt wird, erhalten Sie folgendes Ergebnis:

```
<country><name>USA</name><capital>Washington DC</capital></country>
```

Wenn Sie nun im Dialogfeld "XSLT-Eingabeparameter/Externe XQuery-Variablen" einen Parameter namens **country** erstellen und ihm einen Wert geben (*siehe Abbildung oben*), wird dieser Wert an den Parameter **country** im XSLT-Stylesheet für die Transformation übergeben. Auf diese Art können Sie zur Laufzeit verschiedene Werte an verschiedene Parameter übergeben.

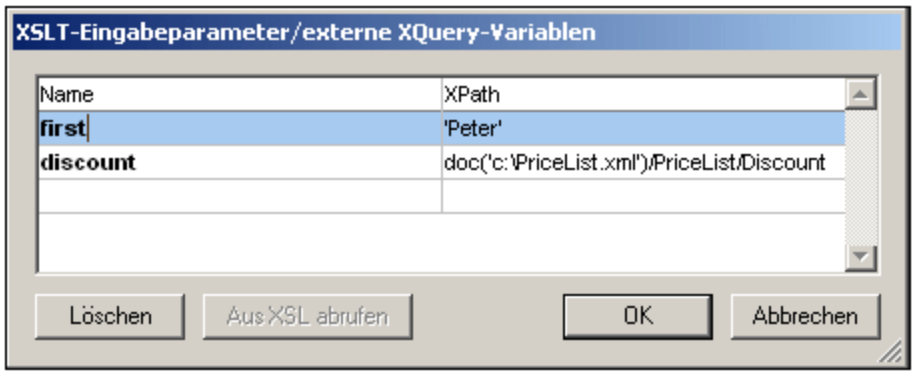
Anmerkung:

- Wenn Sie den Befehl **XSL:FO-Transformation** (**XSL/XQuery | XSL:FO-Transformation**) verwenden, werden Parameter, die im Dialogfeld "XSLT-Eingabeparameter/externe XQuery-Variablen" eingegeben werden, **nicht** an das Stylesheet übergeben. Damit diese Parameter in der PDF-Ausgabe verwendet werden, führen Sie mittels des Befehls XSLT-Transformation **(XSL/XQuery | XSL-Transformation)**

zuerst eine XSL-FO-Transformation durch und anschließend mit Hilfe des Befehls **XSL:FO-Transformation** (**XSL/XQuery | XSL:FO-Transformation**) eine FO-PDF-Transformation.
- Falls Sie einen anderen als den integrierten Altova XSLT-Prozessor verwenden, werden Parameter, die Sie über das Dialogfeld "XSLT-Eingabeparameter" eingeben, nicht an den externen Prozessor übergeben.

Verwendung von externen XQuery-Variablen

Als Wert, den Sie für eine externe XQuery-Variable eingeben, können Sie entweder einen XPath-Ausdruck ohne Anführungszeichen oder einen Textstring innerhalb von Anführungszeichen eingeben. Der Datentyp der externen Variable wird in der Variablendeklaration im XQuery-Dokument angegeben.



Anmerkung: Sobald Sie im Dialogfeld eine Reihe von externen XQuery-Variablen eingegeben haben, werden diese für alle zukünftigen Ausführungen verwendet, bis sie ausdrücklich gelöscht werden oder die Applikation neu gestartet wird. Variablen, die im Dialogfeld eingegeben wurden, werden auf Applikationsebene definiert und werden von diesem Zeitpunkt an bei jeder Transformation, die über die IDE durchgeführt wird, an das jeweilige XSLT-Dokument übergeben. Dies bedeutet:

- Variablen sind nicht mit einem bestimmten Dokument verknüpft und
- Alle Variablen, die im Dialogfeld eingegeben wurden, werden gelöscht, sobald die Applikation (XMLSpy) geschlossen wird.

Anwendungsbeispiel für externe XQuery-Variablen

Im folgenden Beispiel wird im XQuery-Dokument eine Variable `$first` deklariert und anschließend in der Return Clause des FLWOR-Ausdrucks verwendet:

```
xquery version "1.0";
declare variable $first as xs:string external;
let $last := "Jones"
return concat($first, " ", $last )
```

Diese XQuery gibt `Peter Jones` zurück, wenn der Wert der externen Variable (der im Dialogfeld "XSLT-Eingabeparameter/externe XQuery-Variablen" eingegeben wurde) `Peter` ist. Beachten Sie Folgendes:

- Das externe Schlüsselwort in der Variablendeklaration im XQuery-Dokument gibt an, dass es sich bei dieser Variable um eine externe Variable handelt.

© 2016-2022 Altova GmbH

- Die Definition des statischen Typs der Variable ist optional. Wenn der Datentyp für die Variable nicht in der Variablendeklaration angegeben wurde, wird dem Variablenwert der Typ `xs:untypedAtomic` zugewiesen.
- Wenn im XQuery-Dokument eine externe Variable deklariert wurde, aber keine externe Variable dieses Namens an das XQuery-Dokument übergeben wird, wird eine Fehlermeldung ausgegeben.
- Wenn eine externe Variable deklariert und im Dialogfeld "XSLT-Eingabeparameter/externe XQuery-Variablen" eingegeben wurde, wird diese als Gültigkeitsbereich für das auszuführende XQuery-Dokument verwendet. Wird eine neue Variable dieses Namens im XQuery-Dokument deklariert, setzt diese die externe Variable für den Gültigkeitsbereich vorübergehend außer Kraft. Das unten gezeigte XQuery-Dokument gibt z.B. `Paul Jones` zurück, obwohl die externe in-scope Variable `$first` den Wert `Peter` hat.

```
xquery version "1.0";
declare variable $first as xs:string external;
let $first := "Paul"
let $last := "Jones"
return concat($first, " ", $last )
```

28.8.5 XQuery/Update-Ausführung



Mit diesem Befehl wird ein XQuery (1.0/3.1) oder XQuery Update (1.0/3.0)-Dokument ausgeführt. Je nachdem, ob es sich bei der ausgewählten Datei um eine XQuery- oder eine XQuery Upate-Datei handel, wird entweder eine XQuery oder ein XQuery Update ausgeführt. XMLSpy erkennt die Art des Dokuments (XQuery oder XQuery Update) anhand der (im Dialogfeld "Optionen" im Abschnitt "Dateiarten" 1606 definierten) Dateiartzuordnung 1606.

Der XQuery-Prozessor für die Ausführung (1.0 oder 3.1) wird automatisch auf Basis der Versionsdeklaration im Dokument ausgewählt. Falls im Dokument keine Versionsdeklaration vorhanden ist, wird die im Dialogfeld "Optionen" im Abschnitt "XQuery" 1636 definierte Standardversion verwendet. Der Befehl **XQuery/Update-Ausführung** steht zur Verfügung, wenn eine XQuery-, XQuery Update- oder XML-Datei aktiv ist. Bei Aufruf von einer XML-Datei aus erscheint ein Dialogfeld, in dem Sie aufgefordert werden, die XML-Datei mit einer XQuery-Datei zu verknüpfen. Sie können eine Datei auch über eine globale Ressource, eine URL (klicken Sie dazu auf die Schaltfläche **Durchsuchen)** 1280 oder eines der offenen Fenster in XMLSpy auswählen (Klicken Sie dazu auf die Schaltfläche **Fenster**).

Anmerkung: Der Befehl steht auch im Kontextmenü von Elementen im Fenster "Projekt" 118 zur Verfügung.

Automatisieren der Validierung mit RaptorXML 2023

Altova **RaptorXML** ist eine Applikation zur XML-Validierung, XSLT-Transformation und XQuery-Ausführung. Sie kann über die Befehlszeile, über eine COM-Schnittstelle, in Java-Programmen und in .NET-Applikationen aufgerufen werden. So können mit RaptorXML z.B. Aufgaben wie die XSLT-Transformation automatisiert

werden. Sie können z.B. eine Batch-Datei erstellen, die RaptorXML aufruft, um damit eine Reihe von Dokumenten zu transformieren. Nähere Informationen dazu finden Sie in der Dokumentation zu RaptorXML.

28.8.6 Rückwärts-Mapping aktivieren



Mit diesem Befehl, der auch in der Hauptsymbolleiste zur Verfügung steht, wird die Rückwärts-Mapping-Funktion aktiviert.

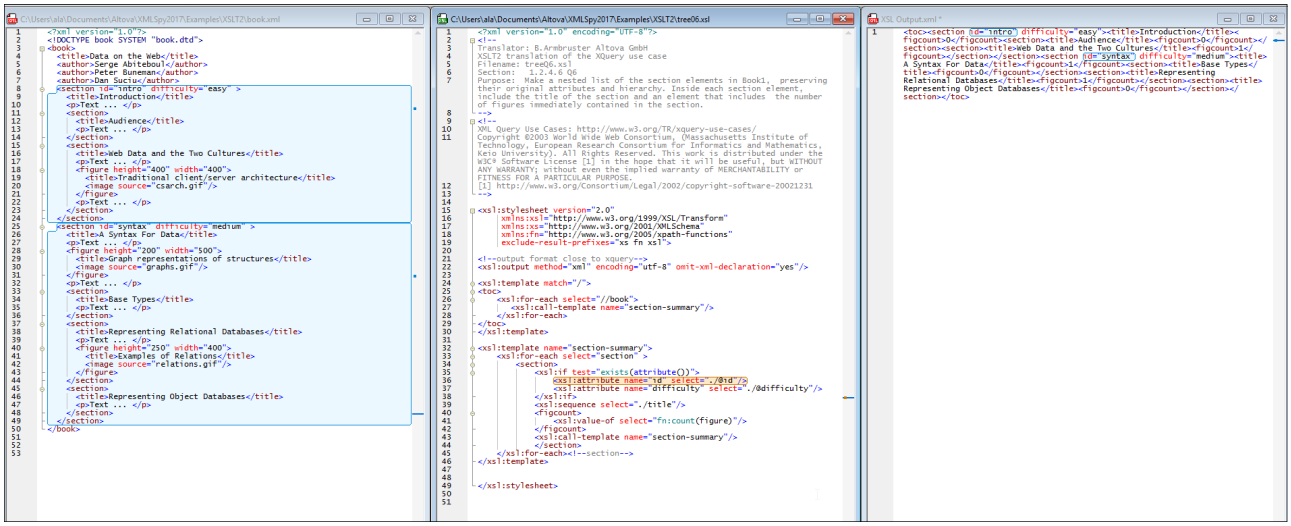
Wenn das Rückwärts-Mapping (mit diesem Befehl) aktiviert wurde, werden XSLT-Transformationen und XQuery-Ausführungen ausgeführt, sodass das Ergebnisdokument auf die XSLT+XML- bzw. XQuery+XML-Ausgangsdokumente rückgemappt werden kann. Das bedeutet, dass bei Klick auf einen Node im Ergebnisdokument die **XSLT-Anweisung** *und* die **XML-Quelldaten,** anhand welcher der jeweilige Ergebnis-Node generiert wurde, markiert werden (*siehe Abbildung unten*). Damit können Sie überprüfen, wie genau die verschiedenen Teile des Ergebnisdokuments bei der XSLT-Transformation oder XQuery-Ausführung erstellt werden. Sie können auch auf einen Node im XSLT/XQuery-Dokument oder im XML-Dokument klicken, um die entsprechenden Teile in den beiden anderen Dokumenten anzuzeigen.

Anmerkung: Ergebnisdokumente jeder Art mit Ausnahme von HTML werden in der Textansicht geöffnet. HTML-Ergebnisdokumente werden in der Browser-Ansicht geöffnet, Sie können jedoch in die Textansicht wechseln. Wenn das Ergebnisdokument in der Browser-Ansicht geöffnet wird, so steht das Rückwärts-Mapping nur durch Auswahl im Ergebnisdokument zur Verfügung. Das Rückwärts-Mapping steht nicht durch Auswählen im XML- oder XSLT/XQuery-Dokument zur Verfügung.

Anmerkung: Das Rückwärts-Mapping steht für Transformationen aus der Authentic-Ansicht[606] oder für diejenigen, die als Projekttransformationen[1344] ausgeführt werden, nicht zur Verfügung.

In der Abbildung unten sehen Sie das Rückwärts-Mapping einer XSLT-Transformation. Alle drei Dokumente - `XML+XSLT+Ergebnis` - werden in dieser Reihenfolge vertikal nebeneinander angeordnet. Die XSLT-Anweisung, die das Attribut **`section/@id`** im Ergebnisdokument generiert, wurde angeklickt. Dadurch werden alle anhand dieser Anweisung generierten Ergebnis-Nodes sowie die XML-Quelldaten, anhand welcher der Ergebnis-Node generiert wurde, markiert. Sie können auch auf Nodes im Ergebnisdokument oder im XML-Dokument klicken, um die dazugehörigen Nodes in den anderen beiden Dokumenten zu markieren.

© 2016-2022 Altova GmbH



Wenn Sie den Befehl "XSLT/XQuery Rückwärts-Mapping aktivieren" auswählen, wird ein Dialogfeld angezeigt, in dem Sie gefragt werden, ob die **Dokumentfenster nach der Transformation nebeneinander angeordnet werden sollen**. Wenn Sie dies bestätigen, werden die drei Dokumente, wie in der Abbildung oben gezeigt, vertikal nebeneinander angeordnet.

Beachten Sie die folgenden Punkte:

- Nur unter einem Pfad gespeicherte Dokumente werden angezeigt; temporäre Strukturen werden nicht angezeigt.
- In einigen Fällen, wie z.B. bei XQuery-Ausführungen, wird das Ergebnisdokument erstellt, ohne dass Daten aus einer XML-Quelle abgerufen werden. In diesen Fällen ist keine XML-Datei am Rückwärts-Mapping beteiligt und daher wird auch keine XML-Datei angezeigt.
- Wenn mehrere XML-Dateien als Datenquellen verwendet werden, so wird die erste bei der Transformation bzw. Ausführung gefundene angezeigt.
- Das Rückwärts-Mapping ist langsamer und speicherintensiver als Transformationen/Ausführungen ohne Rückwärts-Mapping. Dies sollten Sie besonders bei der Arbeit mit großen Dateien berücksichtigen.
- Das Kontextmenü des Ergebnisdokuments enthält (bei Anordnung nebeneinander oder nicht nebeneinander angeordnet) Befehle (**Gehe zu Kontext-Node** und **Gehe zu Quellanweisung**), über die Sie zu den entsprechenden Nodes im XML- bzw. XSLT/XQuery-Dokument gelangen.

Die Symbolleiste "Rückwärts-Mapping"

Die Symbolleiste "Rückwärts-Mapping" (*Abbildung unten*) enthält die folgenden Symbole:



- *HTML bei Mouse-over in Browser-Ansicht markieren:* Wenn das Ergebnisdokument in der Browser-Ansicht angezeigt wird, so steht das Rückwärts-Mapping nur zur Verfügung, indem Sie Inhalt im Ergebnisdokument auswählen. Es steht nicht zur Verfügung, wenn Sie entweder im XML- oder XSLT/XQuery-Dokument Inhalt auswählen. Sie können Inhalt in der Browser-Ansicht auf zwei verschiedene Arten für das Rückwärts-Mapping auswählen: (i) durch Klicken auf Inhalt in der Browser-Ansicht oder (ii) durch Platzieren der Maus über Inhalt. Verwenden Sie diesen Ein/Aus-Befehl, um zwischen diesen beiden Auswahlmethoden zu wechseln. Das Platzieren der Maus über Inhalt ist dann

sinnvoll, wenn es durch Anklicken von Inhalt in der Browser-Ansicht zu einer Änderung im Ergebnisdokument kommen kann (z.B. wenn Sie auf ein Optionsfeld oder eine Auswahlliste klicken).
- *Rückwärts-Mapping beenden:* Beendet das Rückwärts-Mapping.

Beenden des Rückwärts-Mappings

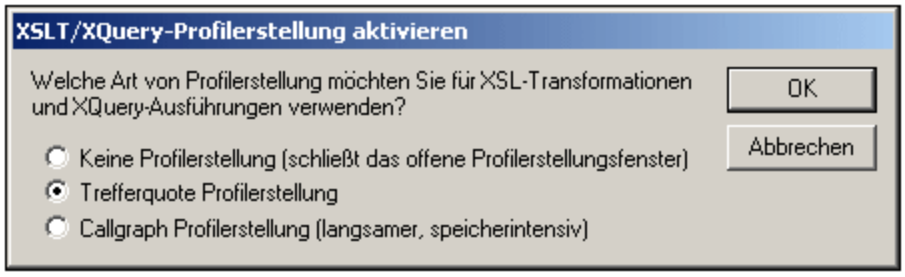
Während eine Rückwärts-Mapping-Sitzung aktiv ist, wird in der Symbolleiste "Rückwärts-Mapping" (*siehe oben)* ein "**Rückwärts-Mapping beenden**"-Symbol angezeigt. Klicken Sie darauf, um das Rückwärts-Mapping zu beenden. Rückwärts-Mappings sollten nur zu Diagnosezwecken verwendet werden. Wenn Sie eines der Dokumente bearbeiten möchten, sollten Sie das Rückwärts-Mapping am besten vorher beenden.

Textmarkierungsfarben

Die Farben des aktiven Rückwärts-Mappings (rückwärts gemappter Inhalt, der gerade ausgewählt ist) und nicht aktiven Rückwärts-Mappings (nicht ausgewählter rückwärts-gemappter Inhalt) können in den Textschriftarteinstellungen (**Extras | Optionen | Schriftarten und Farben**[1623]) in der Kategorie *Verschiedenes* eingestellt werden.

28.8.7 XSLT / XQuery-Profilerstellung aktivieren

Mit dem Befehl **XSLT / XQuery-Profilerstellung aktivieren** öffnen Sie das gleichnamige Dialogfeld. In diesem Dialogfeld können Sie den Profiler[567] aktivieren, ein Tool, das analysiert, wie lange die Ausführung von Anweisungen während einer XSLT-Transformation/XQuery-Ausführung dauert.



28.8.8 XSL zuweisen



Mit diesem Befehl weisen Sie einem XML-Dokument ein XSLT Stylesheet zu. Wenn Sie auf den Befehl klicken, wird ein Dialogfeld geöffnet, in dem Sie angeben können, welche XSLT-Datei zugewiesen werden soll. Sie können eine Datei auch über eine globale Ressource, eine URL (klicken Sie dazu auf die Schaltfläche **Durchsuchen)**[1280] oder eines der offenen Fenster in XMLSpy auswählen (Klicken Sie dazu auf die Schaltfläche **Fenster**).

Es wird eine `xml-stylesheet` Verarbeitungsanweisung in das XML-Dokument eingefügt:

 © 2016-2022 Altova GmbH

```
<?xml-stylesheet type="text/xsl" href="C:\workarea\recursion\recursion.xslt"?
   >
```

Beachten Sie, dass Sie den Pfad zur zugewiesenen Datei durch Aktivieren des Kontrollkästchens "Pfad relativ machen zu" relativ machen können.

28.8.9 XSL-FO zuweisen

Mit diesem Befehl weisen Sie einem XML-Dokument ein XSL Stylesheet für die FO-Transformation zu. Der Befehl öffnet ein Dialogfeld, in dem Sie angeben können, welche XSL oder XSLT Datei zugewiesen werden soll. Die benötigte Verarbeitungsanweisung wird in das XML-Dokument eingefügt:

Durch Aktivieren des Kontrollkästchens "*Pfad relativ machen zu*" können Sie den Pfad der zugewiesenen Datei relativ machen. Sie können eine Datei auch über eine globale Ressource, eine URL (klicken Sie dazu auf die Schaltfläche **Durchsuchen)**[1280] oder eines der offenen Fenster in XMLSpy auswählen (Klicken Sie dazu auf die Schaltfläche **Fenster**).

Bitte beachten Sie: Ein XML-Dokument kann zwei zugewiesene XSLT-Dateien haben: Eine Standard XSLT-Transformationsdatei, sowie eine zweite Datei für eine XSLT-Transformation in FO.

28.8.10 XML-Beispieldatei zuweisen



Mit diesem Befehl weisen Sie einem XSLT-Dokument eine **XML-Datei** zu. Der Befehl fügt eine Verarbeitungsanweisung ein, die eine XML-Datei spezifiziert, die mit dieser XSLT-Datei verarbeitet werden soll, wenn die XSL-Transformation an der XSLT-Datei ausgeführt wird:

```
<?altova_samplexml C:\workarea\html2xml\article.xml?>
```

Beachten Sie bitte, dass Sie den Pfad der zugewiesenen Datei relativ machen können. Aktivieren Sie dazu das Kontrollkästchen "Pfad relativ machen zu...". Sie können eine Datei auch über eine globale Ressource, eine URL (klicken Sie dazu auf die Schaltfläche **Durchsuchen)**[1280] oder eines der offenen Fenster in XMLSpy auswählen (Klicken Sie dazu auf die Schaltfläche **Fenster**).

28.8.11 Gehe zu XSL



Mit diesem Befehl öffnen Sie das zugehörige XSLT Dokument, falls Ihr XML-Dokument eine Processing Instruction zur Verarbeitung eines Stylesheet (also eine XSLT-Zuweisung) wie z.B. die folgende enthält:

```
<?xml-stylesheet type="text/xsl" href="Company.xsl"?>
```

Der Befehl **Gehe zu XSL** öffnet das XSLT-Dokument anschließend in XMLSpy.

28.8.12 Gehe zu Quellanweisung

Wenn Sie im Ergebnisdokument eines Rückwärts-Mapping[1418] auf **Gehe zu Quellanweisung** klicken, gelangen Sie zur Quellanweisung im XSLT- oder XQuery-Dokument, durch das der ausgewählte Node im Ergebnisdokument erzeugt wurde. Sie könnten die Anweisung anschließend bei Bedarf direkt bearbeiten.

Eine Beschreibung eines Rückwärts-Mappings finden Sie unter der Beschreibung des Menübefehls **XSL | Rückwärts-Mapping aktivieren**[1418].

28.8.13 Gehe zu Kontext-Node

Wenn Sie im Ergebnisdokument eines Rückwärts-Mapping[1418] einen Node auswählen, gelangen Sie bei Auswahl des Befehls **Gehe zu Kontext-Node** zum Daten-Node im XML-Dokument, durch den der Inhalt des ausgewählten Node im Ergebnisdokument erzeugt wurde. Sie könnten den Daten-Node anschließend bei Bedarf direkt bearbeiten.

Eine Beschreibung eines Rückwärts-Mappings finden Sie unter der Beschreibung des Menübefehls **XSL | Rückwärts-Mapping aktivieren**[1418].

28.8.14 Debugger starten



Alt+F11

Mit diesem Befehl starten Sie den Debugger oder fahren mit der Verarbeitung des XSLT/XQuery-Dokuments fort. Wenn Breakpoints gesetzt wurden, wird der Ablauf an dieser Stelle angehalten. Wenn Tracepoints gesetzt wurden, wird das Ergebnis dieser Anweisungen im Fenster "Trace" angezeigt, wenn der schließende Node der Anweisung mit dem Tracepoint erreicht wurde. Wenn die Debugger-Sitzung noch nicht gestartet wurde, wird diese mit diesem Befehl gestartet und am ersten Node angehalten, um diesen zu verarbeiten. Wenn die

© 2016-2022 Altova GmbH

Sitzung läuft, wird das XSLT/XQuery-Dokument bis zum Ende verarbeitet oder bis das Programm auf den nächsten Breakpoint stößt.

28.8.15 Debugger anhalten



Mit diesem Befehl stoppen Sie den Debugger. Dies ist nicht dasselbe, wie das Stoppen der Debugger-Sitzung, in der der Debugger gerade läuft. Dieser Befehl eignet sich, wenn Sie ein Dokument während einer Debugger-Sitzung bearbeiten oder andere Dateien in derselben Debugger-Sitzung verwenden möchten. Wenn Sie den Debugger angehalten haben, müssen Sie ihn erneut starten, um am Anfang des XSLT/XQuery-Dokuments zu beginnen.

28.8.16 Debugger neu starten



Mit diesem Befehl wird der XSLT Debugger mit den aktuell ausgewählten Dateien erneut gestartet und das Ausgabefenster wird gelöscht.

28.8.17 Debugger-Sitzung beenden



Mit diesem Befehl beenden Sie die Debugger-Sitzung und kehren zur normalen XMLSpy Ansicht zurück, die vor dem Start des Debuggers aktiv war. Ob die für die Debugger-Sitzung geöffneten Ausgabedokumente geöffnet bleiben oder nicht, hängt davon ab, welche Einstellung Sie im Dialogfeld XSLT/XQuery Debugger-Einstellungen[546] gewählt haben.

28.8.18 Einsteigen



F11

Mit diesem Befehl gehen Sie Schritt für Schritt durch alle Nodes und XPath-Ausdrücke des Stylesheets. Dieser Befehl dient auch zum erneuten Starten des Debuggers, nachdem dieser angehalten wurde.

28.8.19 Aussteigen



Umschalt+F11

Mit diesem Befehl steigen Sie aus dem aktuellen Node aus und gehen zum nächsten **Zwilling** des Parent-Node oder zum nächsten Node auf der nächsthöheren Ebene über der des Parent-Node.

28.8.20 Überspringen



Strg+F11

Mit diesem Befehl überspringen Sie den aktuellen Node und gehen zum nächsten Node auf derselben Ebene oder zur nächsthöheren Ebene über der des aktuellen Node. Dieser Befehl dient auch zum erneuten Starten des Debuggers, nachdem er angehalten wurde.

28.8.21 Aktuellen Execution Node einblenden



Dieser Befehl dient zum Anzeigen/Auswählen des aktuellen Execution Node im XSLT/XQuery-Dokument und des entsprechenden Context Node im XML-Dokument. Dies ist nützlich, wenn Sie auf andere Register geklickt haben, auf denen bestimmter Code im XSLT Stylesheet bzw. in der XML-Datei angezeigt oder markiert wird, und anschließend wieder zum Ausgangspunkt zurückkehren möchten.

28.8.22 Breakpoint einfügen/löschen



F9

Dieser Befehl dient zum Einfügen oder Löschen eines Breakpoint an der angeklickten Position. Sie können sowohl für Nodes im XSLT/XQuery-Dokument als auch im XML-Dokument Breakpoints innerhalb einer Zeile definieren, um festzulegen, wo die Verarbeitung unterbrochen werden soll. An der Stelle, an der Sie den Breakpoint setzen, erscheint oberhalb des Nodes eine strichlierte rote Linie. An schließenden Nodes können keine Breakpoints definiert werden, und Breakpoints an Attributen in XSLT-Dokumenten werden ignoriert.

© 2016-2022 Altova GmbH

Dieser Befehl steht auch zur Verfügung, wenn Sie mit der rechten Maustaste auf die gewünschte Stelle für den Breakpoint klicken.

28.8.23 Tracepoint einfügen/löschen



Umschalt+F9

Mit dem Befehl **XSL/XQuery | Tracepoint einfügen/löschen** wird ein Tracepoint an der aktuellen Cursorposition in einem XSLT/XQuery-Dokument eingefügt oder gelöscht. Bei Anweisungen mit einem Tracepoint wird der Wert der Anweisung beim Debuggen im Fenster "Trace" angezeigt, sobald der schließende Node dieser Anweisung erreicht wird. An der Stelle, an der Sie den Tracepoint setzen, erscheint oberhalb des Nodes eine strichlierte blaue Linie. An schließenden Nodes können keine Tracepoints definiert werden. Dieser Befehl steht auch zur Verfügung, wenn Sie mit der rechten Maustaste auf die gewünschte Stelle für den Breakpoint klicken.

28.8.24 Breakpoint aktivieren/deaktivieren



Strg+F9

Mit dem Befehl **XSL/XQuery | Breakpoint aktivieren/deaktivieren** schalten Sie bereits definierte Breakpoints ein oder aus. Die rote Breakpoint-Markierung wird grau angezeigt, sobald der Breakpoint deaktiviert wird. Der Debugger hält bei ausgeschalteten Breakpoints nicht an. Um einen Breakpoint ein- bzw. auszuschalten, setzen Sie den Cursor in den Node-Namen und klicken Sie auf **Breakpoint aktivieren/deaktivieren,** Dieser Befehl kann auch über das Kontextmenü aufgerufen werden, indem Sie mit der rechten Maustaste an die Stelle klicken, an der der Breakpoint aktiviert oder deaktiviert werden soll.

28.8.25 Tracepoint aktivieren/deaktivieren



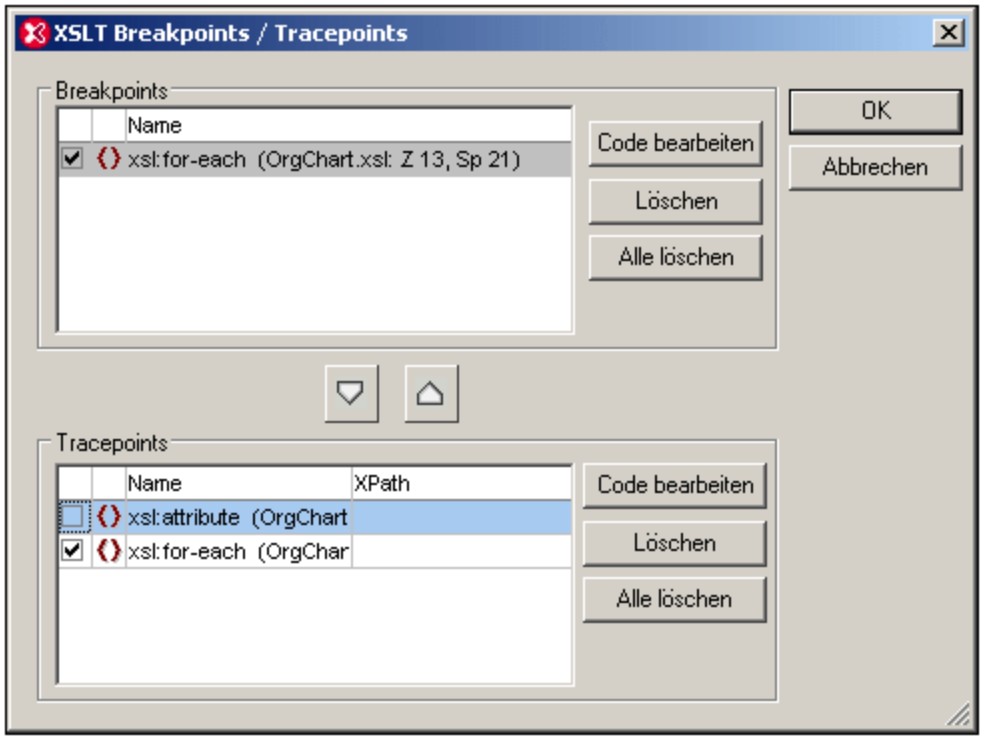
Strg+Umschalt+F9

Mit dem Befehl **XSL/XQuery | Tracepoint aktivieren/deaktivieren** schalten Sie bereits definierte Tracepoints ein oder aus. Die blaue Tracepoint-Markierung wird grau angezeigt, sobald der Tracepoint deaktiviert wird. Bei Anweisungen, bei denen der Tracepoint deaktiviert wurde, wird keine Ausgabe angezeigt. Um einen Tracepoint ein- bzw. auszuschalten, setzen Sie den Cursor in den Node-Namen und klicken Sie auf **Tracepoint aktivieren/deaktivieren,** Dieser Befehl kann auch über das Kontextmenü aufgerufen werden, indem Sie mit der rechten Maustaste an die Stelle klicken, an der der Tracepoint aktiviert oder deaktiviert werden soll.

28.8.26 Breakpoints/Tracepoints...



Mit dem Befehl **XSL/XQuery | Breakpoints/Tracepoints** öffnen Sie das Dialogfeld "XSLT/XQuery Breakpoints", in welchem eine Liste zuvor definierter Breakpoints und Tracepoints (einschließlich aller deaktivierten Breakpoints und Tracepoints) in allen Dateien in der aktuellen Debugger-Sitzung angezeigt wird.



Es stehen die folgenden Funktionalitäten zur Verfügung:

- Das Kontrollkästchen zeigt an, ob ein Breakpoint aktiv (Häkchen) oder deaktiviert ist.
- Es kann immer nur ein Breakpoint oder Tracepoint markiert werden.
- Um einen markierten Breakpoint/Tracepoint zu entfernen, klicken Sie auf die Schaltfläche **Entfernen**.
- Zum Entfernen aller Breakpoints oder Tracepoints klicken Sie auf die entsprechende Schaltfläche **Alle löschen**.
- Bei Auswahl der Schaltfläche **Code bearbeiten** gelangen Sie direkt zum Breakpoint/Tracepoint in der jeweiligen Datei.
- Klicken Sie auf ▽, um den markierten Breakpoint in das Fenster "Tracepoints" zu verschieben.
- Klicken Sie auf △, um den markierten Tracepoint in das Fenster "Breakpoints" zu verschieben.
- In der Spalte "XPath" im Fenster "Tracepoints" können Sie einen XPath-Ausdruck für jeden einzelnen Tracepoint definieren. Dadurch können Sie eine Bedingung definieren, unter der der Tracepoint angewendet werden soll.

© 2016-2022 Altova GmbH

28.8.27 Debug-Fenster

Wenn Sie den Cursor über den Befehl **Debug-Fenster** platzieren, erscheint ein Untermenü mit einer Liste der verschiedenen Informationsfenster des XSLT/XQuery-Debuggers. Wenn Sie eines der Informationsfenster aus dieser Liste auswählen, wird dieses Fenster auf der XSLT/XQuery Debugger-Oberfläche ein- bzw. ausgeblendet. Dieser Befehl kann nur während einer laufenden Debugger-Sitzung verwendet werden.

28.8.28 Debugeinstellungen



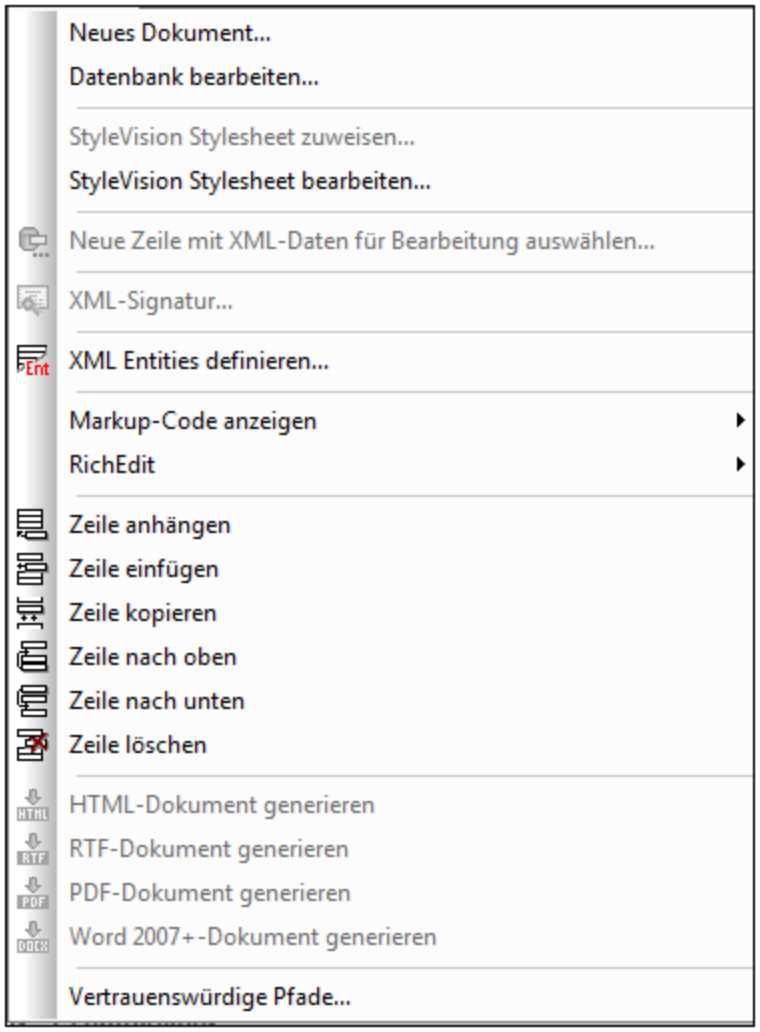
Mit diesem Befehl rufen Sie das Dialogfeld Debugeinstellungen[546] auf, in welchem Sie Benutzereinstellungen für den Debugger festlegen können. Nähere Informationen dazu finden Sie im Abschnitt XSLT/XQuery Debugger[540].

28.9 Menü "Authentic"

Über die Authentic-Ansicht können Sie XML-Dokumente editieren, **die auf StyleVision Power Stylesheets basieren, die mit StyleVision von Altova erstellt worden sind!** Diese Stylesheets enthalten alle Informationen, die benötigt werden, damit eine XML-Datei grafisch in der Authentic-Ansicht dargestellt werden kann. Zusätzlich zu diesen Anzeigeinformationen ermöglichen Ihnen StyleVision Power Stylesheets außerdem, Daten in eine XML-Datei zu schreiben. Diese Daten werden mit Hilfe aller XSLT-Stylesheet-Funktionen dynamisch verarbeitet und sofort in Form einer Authentic-Ansicht ausgegeben.

Des Weiteren können Sie StyleVision Power Stylesheets erstellen, um eine editierbare XML-Ansicht einer Datenbank anzuzeigen. Das StyleVision Power Stylesheet enthält Informationen, um eine Verbindung zu einer Datenbank herzustellen, die Datenbankdaten in der Authentic-Ansicht anzuzeigen und wieder zurück in die Datenbank zu schreiben.

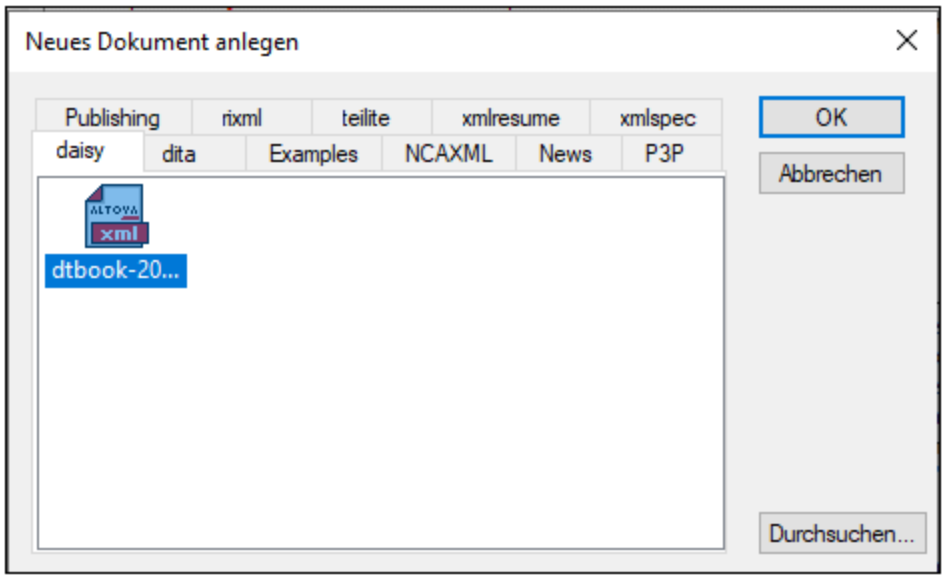
Das Menü "Authentic" enthält Befehle zum Editieren von XML-Dokumenten in der Authentic-Ansicht. Ein Tutorial zur Authentic-Ansicht finden Sie im Abschnitt Tutorials Authentic-Ansicht 608.



 © 2016-2022 Altova GmbH

28.9.1 Neues Dokument

Mit diesem Befehl öffnen Sie eine neue XML-Dokumentvorlage in der Authentic-Ansicht. Die XML-Dokumentvorlage basiert auf einem StyleVision Power Stylesheet (.sps-Datei) und wird durch Auswahl des StyleVision Power Stylesheet im Dialogfeld "Neues Dokument anlegen" (*Abbildung unten*) geöffnet. Nachdem Sie ein SPS ausgewählt und mit OK bestätigt haben, wird die für diese SPS-Datei definierte XML-Dokumentvorlage in der Authentic-Ansicht geöffnet.



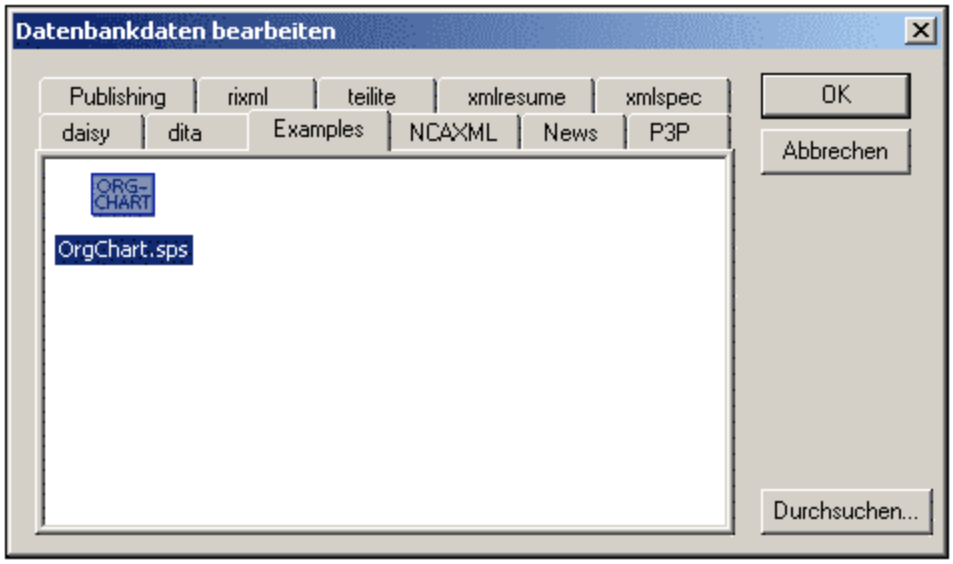
Im Dialogfeld "Neues Dokument anlegen" steht eine Reihe von XML-Dokumentvorlagen, die auf beliebten DTDs oder Schemas basieren, zur Auswahl zur Verfügung. Alternativ dazu können Sie eine benutzerdefinierte SPS-Datei, der eine XML-Vorlagendatei zugewiesen wurde, aussuchen. SPS-Dateien werden mit Altova StyleVision, einer Applikation zum Erstellen von XML-Dokumentvorlagen anhand einer DTD oder eines XML-Schemas, erstellt. Nachdem Sie die erforderliche SPS-Datei in StyleVision erstellt haben, wird der SPS-Datei (in StyleVision) eine XML-Datei als XML-Vorlagendatei zugewiesen. Die Daten in dieser XML-Datei liefern die Ausgangsdaten der neuen, in der Authentic-Ansicht von XMLSpy geöffneten Dokumentvorlage.

Die neue XML-Dokumentvorlage enthält daher die im SPS definierten Layout-Eigenschaften und die Daten der als XML-Vorlagendatei ausgewählten XML-Datei. Der Benutzer der Authentic -Ansicht kann die XML-Dokumentvorlage nun über eine grafische WYSIWYG-Oberfläche bearbeiten und als XML-Dokument speichern.

28.9.2 Datenbankdaten bearbeiten

Mit dem Befehl **Authentic | Datenbankdaten bearbeiten** öffnen Sie eine editierbare Ansicht einer Datenbank (DB) in der Authentic-Ansicht. Alle Informationen über die Herstellung einer Verbindung zur Datenbank, die Anzeige der DB und die Übernahme von Änderungen an der Datenbank über die Authentic-Ansicht befinden sich in einem StyleVision Power Stylesheet. Mit dem Befehl **Datenbankdaten bearbeiten** öffnen Sie ein StyleVision Power Stylesheet, stellen Sie eine Verbindung zur DB her und zeigen die DB-Daten (durch eine XML-Linse) in der Authentic-Ansicht an.

Bei Aufruf des Befehls **Datenbankdaten bearbeiten** erscheint das Dialogfeld "Datenbankdaten bearbeiten" (*Abbildung unten*).



Suchen Sie die gewünschte SPS-Datei und wählen Sie diese aus. Dadurch wird eine Verbindung zur DB hergestellt und eine editierbare Ansicht der Datenbank in der Authentic-Ansicht geöffnet. Das Design der in der Authentic-Ansicht angezeigten DB-Ansicht ist im StyleVision Power Stylesheet enthalten.

Anmerkung: Wenn Sie versuchen, ein StyleVision Power Stylesheet mit dem Befehl **Datenbankdaten bearbeiten** zu öffnen, das nicht auf einer DB basiert, oder wenn Sie versuchen, ein DB-basiertes StyleVision Power Stylesheet zu öffnen, das in einer StyleVision-Version vor der Version StyleVision 2005 erstellt wurde, erhalten Sie eine Fehlermeldung.

Anmerkung: StyleVision Power Stylesheets werden mit Altova StyleVision erstellt.

28.9.3 StyleVision Stylesheet zuweisen

Mit dem Befehl **StyleVision Stylesheet zuweisen** weisen Sie einem **XML-Dokument** ein StyleVision Power Stylesheet (SPS) zu, um die Ansicht und Bearbeitung in der Authentic-Ansicht zu ermöglichen. Das StyleVision Power Stylesheet, das der XML-Datei zugewiesen werden soll, muss auf demselben Schema basieren, auf dem auch die XML-Datei basiert.

So weisen Sie einer XML-Datei ein StyleVision Power Stylesheet zu:

1. Machen Sie die XML-Datei zur aktiven und wählen Sie den Befehl **Stylevision Stylesheet zuweisen...**.
2. Daraufhin wird ein Dialogfeld geöffnet, in dem Sie angeben können, welche StyleVision Power Stylesheet-Datei zugewiesen werden soll.
3. Klicken Sie auf OK, um die gewünschte SPS-Anweisung in Ihr XML-Dokument einzufügen. Beachten Sie bitte, dass Sie den Pfad der zugewiesenen Datei relativ machen können, indem Sie das Kontrollkästchen "Pfad relativ machen zu ..." aktivieren. Sie können eine Datei auch über eine globale Ressource, eine URL (klicken Sie dazu auf die Schaltfläche **Durchsuchen)** 1280 oder eines der offenen Fenster in XMLSpy auswählen (Klicken Sie dazu auf die Schaltfläche **Fenster**).

© 2016-2022 Altova GmbH

```
<?xml version="1.0" encoding="UTF-8"?>
<?altova_sps HTML-Orgchart.sps?>
```

Im obigen Beispiel ist der Name des StyleVision Power Stylesheet `HTML_Orgchart.sps`, und die Datei befindet sich im selben Verzeichnis wie die XML-Datei.

Anmerkung: In früheren Altova-Produktversionen wurde eine Verarbeitungsanweisung mit dem Ziel bzw. Namen `xmlspysps` verwendet. Eine Verarbeitungsanweisung sah daher etwa folgendermaßen aus: `<? xmlspysps HTML-Orgchart.sps?>`. Diese älteren Verarbeitungsanweisungen sind auch in aktuellen Altova-Produktversionen bei Authentic immer noch gültig.

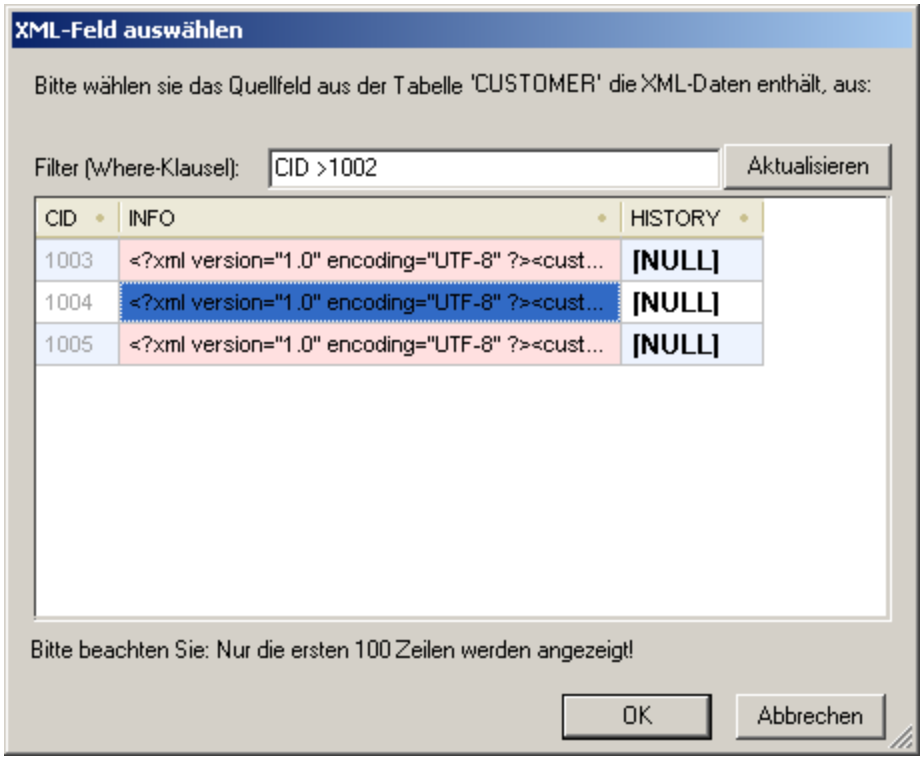
28.9.4 StyleVision Stylesheet bearbeiten

Der Befehl **StyleVision Stylesheet bearbeiten** steht nur in der Authentic-Ansicht, d.h. nur, wenn dem XML-Dokument ein StyleVision Power Stylesheet zugewiesen wurde, zur Verfügung. Der Befehl startet StyleVision, wo Sie das StyleVision Power Stylesheet sofort bearbeiten können.

28.9.5 Neue Zeile mit XML-Daten für Bearbeitung auswählen

Mit diesem Befehl können Sie eine neue Zeile in der entsprechenden Tabelle einer XML-Datenbank wie z.B. IBM DB2 auswählen. Diese Zeile wird nun in der Authentic-Ansicht angezeigt, kann hier bearbeitet und wieder in der Datenbank gespeichert werden.

Bei Verwendung einer XML-Datenbank als XML-Datenquelle handelt es sich bei den in der Authentic-Ansicht angezeigten XML-Daten um das XML-Dokument, das in einer der Zellen der XML-Datenspalte enthalten ist. Mit dem Befehl **Neue Zeile mit XML-Daten für Bearbeitung auswählen** können Sie ein XML-Dokument aus einer anderen Zelle (oder Zeile) dieser XML-Spalte auswählen. Bei Auswahl des Befehls **Neue Zeile...auswählen** wird das Dialogfeld "XML-Feld auswählen" aufgerufen (*Abbildung unten*), in dem die Tabelle, die die XML-Spalte enthält angezeigt wird.



Sie können für diese Tabelle einen Filter eingeben. Dabei sollte es sich um eine SQL `WHERE`-Klausel handeln (nur die Bedingung ohne das `WHERE`-Schlüsselwort wie z.B.: `CID>1002`). Klicken Sie auf **Aktualisieren**, um das Dialogfeld zu aktualisieren. In der Abbildung oben sehen Sie das Ergebnis einer gefilterten Ansicht. Wählen Sie als nächstes die Zelle, die das gewünschte XML-Dokument enthält aus, und klicken Sie auf **OK**. Das XML-Dokument in der ausgewählten Zelle (Zeile) wird in die Authentic-Ansicht geladen.

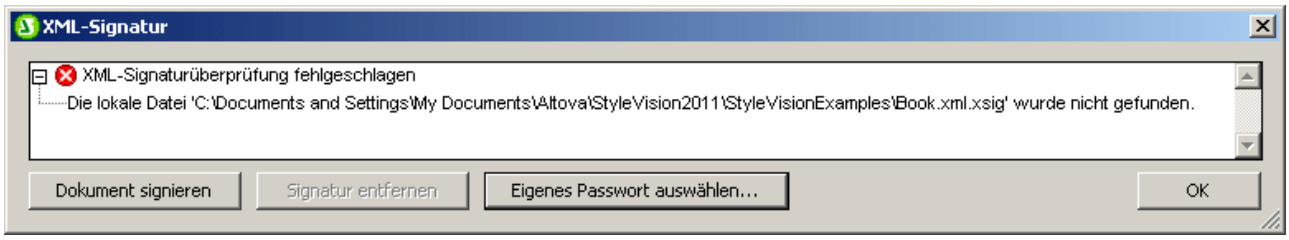
28.9.6 XML-Signatur

Der Befehl **XML-Signatur** steht in der Authentic-Ansicht zur Verfügung, wenn XML-Signaturen im dazugehörigen SPS aktiviert sind. Außerdem steht der Befehl **XML-Signatur** in Form der Schaltfläche "XML-Signatur"  in der Authentic-Symbolleiste zur Verfügung.

Überprüfung und eigenes Zertifikat/Passwort

Bei Auswahl des Befehls **XML-Signatur** wird mit der Überprüfung der Signatur begonnen. Wenn keine Signatur im Dokument vorhanden ist, erscheint im Dialogfeld "XML-Signatur" eine entsprechende Meldung (*siehe Abbildung unten*) und das Dialogfeld weist eine Schaltfläche auf, mit der der Benutzer der Authentic-Ansicht das Dokument signieren kann.

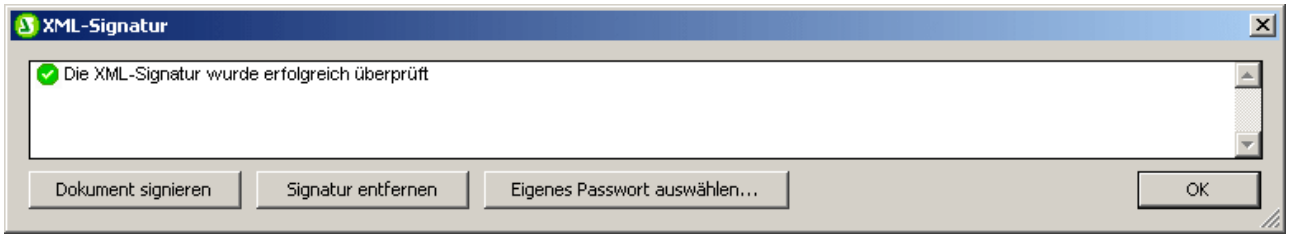
© 2016-2022 Altova GmbH



Wenn in diesem Dialogfeld die Schaltfläche **Eigenes Zertifikat auswählen** oder **Eigenes Passwort auswählen** vorhanden ist, so bedeutet das, dass der Benutzer der Authentic-Ansicht die Option hat, ein eigenes Zertifikat/Passwort auszuwählen. (Die Entscheidung, ob ein Zertifikat oder Passwort gewählt werden soll, hat der SPS-Designer zum Zeitpunkt der Konfiguration der Signatur getroffen. Die Signatur ist entweder zertifikatbasiert oder passwortbasiert.) Wenn eine dieser Schaltflächen im Dialogfeld vorhanden ist und der Benutzer der Authentic-Ansicht darauf klickt, kann er ein Zertifikat auswählen oder ein Passwort eingeben. Das vom Benutzer der Authentic-Ansicht gewählte Zertifikat/Passwort wird im Arbeitsspeicher gespeichert und gilt nur für die aktuelle Sitzung. Wenn das Dokument oder die Applikation geschlossen wird, nachdem ein Zertifikat oder Passwort ausgewählt wurde, wird die Zertifikat-/Passworteinstellung wieder auf die mit dem SPS gespeicherte Originaleinstellung zurückgesetzt.

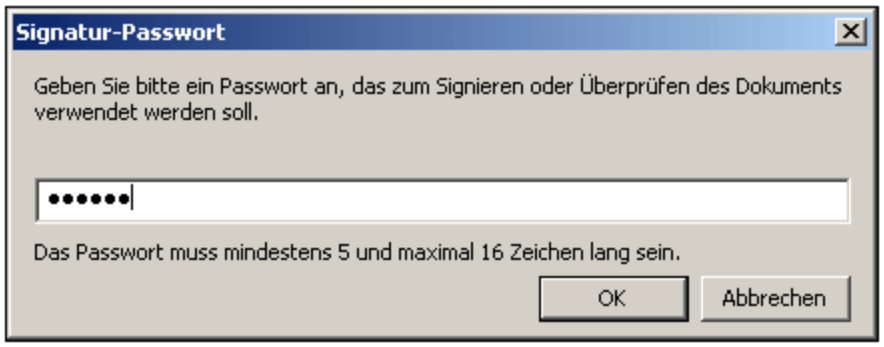
Informationen für die Überprüfung und Authentifizierung

Wenn die Überprüfung an einem signierten Dokument durchgeführt wird, sind zwei allgemeine Situationen möglich. Erstens: Wenn die Authentifizierungsinformationen (in der Signatur oder dem SPS) verfügbar sind, so wird die Überprüfung direkt durchgeführt und das Ergebnis wird angezeigt (*Abbildung unten*).



Die Authentifizierungsinformationen sind entweder die Schlüsselinformationen des Zertifikats oder das zum Signieren verwendete Passwort. Der SPS-Designer hat definiert, ob die Schlüsselinformationen des Zertifikats bei der Signierung des XML-Dokuments in der Signatur gespeichert werden sollen oder ob das Passwort im Fall einer passwortbasierten Signatur im SPS gespeichert werden soll. In beiden Fällen stehen die Authentifizierungsinformationen zur Verfügung. Infolgedessen wird die Überprüfung direkt durchgeführt, ohne dass Eingaben durch den Benutzer der Authentic-Ansicht erforderlich sind.

Die zweite mögliche Situation tritt ein, wenn die Authentifizierungsinformationen in der Signatur (Schlüsselinformationen des Zertifikats) oder im SPS (Passwort) nicht verfügbar sind. In dieser Situation wird der Benutzer der Authentic-Ansicht aufgefordert, die Authentifizierungsinformationen anzugeben, nämlich ein Passwort (*siehe Abbildung unten*) oder den Pfad des Zertifikats. Wenn Authentic gemäß dem SPS ein eigenes Passwort oder Zertifikat auswählen darf, aktivieren Sie dazu die Option **Eigenes Passwort (oder Zertifikat) auswählen**.



28.9.7 XML Entities definieren

Unabhängig davon, ob Ihr Dokument auf einer DTD oder einem XML-Schema basiert, können Sie für die <% AUTH-VIEW%>-Ansicht Entities definieren, die dann in der Eingabehilfe "Entities" und im Kontext-Untermenü **Entity einfügen** angezeigt werden. Wenn Sie in der Entities-Eingabehilfe auf eine Entity doppelklicken, wird diese an der Cursorposition eingefügt.

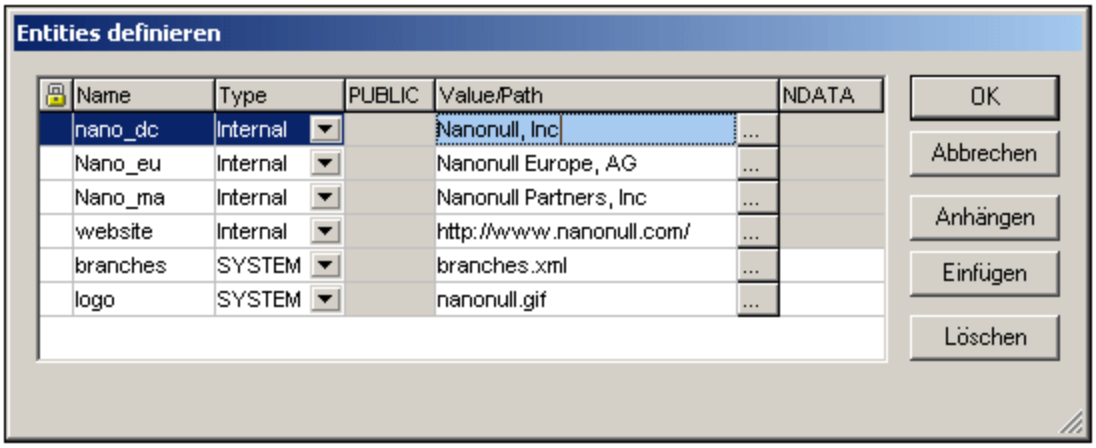
Eine Entity eignet sich v.a. dann, wenn Sie einen Textstring, ein XML-Fragment oder einige andere externe Ressourcen an mehreren Stellen in Ihrem Dokument verwenden möchten. Die Entity, bei der es sich im Grunde um eine Kurzbezeichnung für die benötigten Daten handelt, wird im Dialogfeld "Entities definieren" definiert. Nachdem Sie die Entity definiert haben, können Sie sie an mehreren Stellen in Ihrem Dokument einfügen. Dies spart Zeit und ist effizient, wenn Sie Änderungen am Dokument vornehmen möchten.

Sie können in Ihrem Dokument zwei grundlegende Arten von Entities verwenden: **geparste Entities**, wobei es sich um XML-Daten handelt (entweder einen Textstring oder ein Fragment aus einem XML-Dokument) oder **ungeparste Entities**, wobei es sich um nicht-XML-Daten handelt wie z.B. eine Binärdatei (normalerweise eine Grafik, Ton oder ein Multimedia-Objekt). Jede Entity hat einen Namen und einen Wert. Im Fall von geparsten Entities ist die Entity ein Platzhalter für die XML-Daten. Der Wert der Entity sind entweder die XML-Daten selbst oder eine URI, die auf eine `.xml`-Datei verweist, die die XML-Daten enthält. Bei ungeparsten Entities ist der Wert der Entity eine URI, die auf eine nicht-XML-Datei verweist.

So definieren Sie eine Entity:

1. Wählen Sie **Authentic | XML Entities definieren**. Dadurch wird das Dialogfeld "Entities definieren" geöffnet.

 © 2016-2022 Altova GmbH



2. Geben Sie den Namen Ihrer Entity im Feld **Name** ein. Dies ist der Name, der in der Entities-Eingabehilfe angezeigt werden wird.
3. Wählen Sie den Typ der Entity aus der Dropdown-Liste im Feld **Type** aus. Es stehen drei Typen zur Auswahl. Eine **interne** (internal) Entity ist eine Entity, für die der zu verwendende Text im XML-Dokument selbst gespeichert ist. Wenn Sie **PUBLIC** oder **SYSTEM** auswählen, wird angegeben, dass sich die Ressource außerhalb der XML-Datei befindet und mittels eines "Public Identifier" oder eines "System Identifier" gefunden wird. Ein System Identifier ist eine URI, die den Pfad der Ressource definiert. Ein Public Identifier ist ein pfadunabhängiger Identifier, der es manchen Prozessoren ermöglicht, die Ressource zu identifizieren. Wenn Sie sowohl einen Public als auch einen System Identifier angeben, wird der Public Identifier zum System Identifier aufgelöst und es wird der System Identifier verwendet.
4. Wenn Sie als Type PUBLIC ausgewählt haben, geben Sie den Public Identifier der Ressource im Feld PUBLIC ein. Wenn Sie Internal oder SYSTEM als Type ausgewählt haben, ist das Feld PUBLIC deaktiviert.
5. Im Feld **Value/Path** haben Sie folgende Eingabemöglichkeiten:

   - Wenn die Entity "Internal" ist, geben Sie den Textstring ein, der als Wert der Entity verwendet werden soll. Setzen Sie den Textstring nicht in Anführungszeichen. Alle Anführungszeichen werden als Teil des Textstrings behandelt.
   - Wenn der Entity-Typ SYSTEM ist, geben Sie die URI der Ressource ein oder wählen Sie mit Hilfe der **Durchsuchen**-Schaltfläche eine Ressource auf Ihrem lokalen Netzwerk aus. Wenn die Ressource geparste Daten enthält, muss es eine **.xml**-Datei sein (d.h. sie muss eine XML-Erweiterung haben). Als Alternative dazu kann die Ressource eine Binärdatei sein, z.B. eine GIF-Datei.
   - Wenn der Entity-Typ PUBLIC ist, müssen Sie zusätzlich einen System Identifier in dieses Feld eingeben.

6. Der Eintrag NDATA sagt dem Prozessor, dass diese Entity nicht geparst, sondern an den entsprechenden Prozessor gesendet werden soll. Das Feld NDATA sollte daher nur bei nicht geparsten Entities verwendet werden.

Dialogfeldfunktionen

Sie können Entities anhängen, einfügen und löschen, indem Sie auf die entsprechende Schaltfläche klicken. Des Weiteren können Sie Entities spaltenweise alphabetisch sortieren, indem Sie auf die Kopfzeile der jeweiligen Spalte klicken; bei einmaligem Klicken werden die Entities aufsteigend sortiert, bei zweimaligem Klicken absteigend. Die Größe des Dialogfelds und die Spaltenbreite lassen sich anpassen.

Sobald eine Entity im XML-Dokument verwendet wird, ist sie gesperrt und kann im Dialogfeld "Entities definieren" nicht mehr bearbeitet werden. Gesperrte Entities werden durch das Schlosssymbol in der ersten Spalte gekennzeichnet. Durch das Sperren einer Entity wird sichergestellt, dass das Dokument in Bezug auf Entities gültig ist. (Das Dokument wäre ungültig, wenn eine Entity referenziert aber nicht definiert ist.)

Doppelt vorhandene Entities werden markiert.

Einschränkungen

- Eine in einer anderen Entity enthaltene Entity wird weder im Dialogfeld, noch der Authentic-Ansicht noch der XSLT-Ausgabe aufgelöst. Das kaufmännische Und-Zeichen einer solchen Entity wird in seiner Escape-Form also als `&amp` angezeigt.
- Externe Entities werden in der Authentic-Ansicht nicht aufgelöst, außer es handelt sich bei der Entity um eine Bilddatei, die als Wert eines Attributs eingegeben wurde, dessen Typ im Schema als **ENTITY** oder **ENTITIES** definiert wurde. Solche Entities werden aufgelöst, wenn das Dokument mit einem in der SPS generierten XSLT verarbeitet wird.

28.9.8 Markup-Code anzeigen

Der Befehl **Markup-Code anzeigen** hat ein Untermenü mit Optionen, wie der Markup-Code in Authentic-XML-Dokument angezeigt werden soll:



Mit dem Befehl **Markup-Symbole ausblenden** werden die Markup-Symbole in der Authentic-Ansicht ausgeblendet.



Mit dem Befehl **Kleine Markup-Symbole einblenden** werden kleine Markup-Symbole in der Authentic-Ansicht angezeigt.



Mit dem Befehl **Große Markup-Symbole einblenden** werden große Markup Symbole in der Authentic-Ansicht angezeigt.

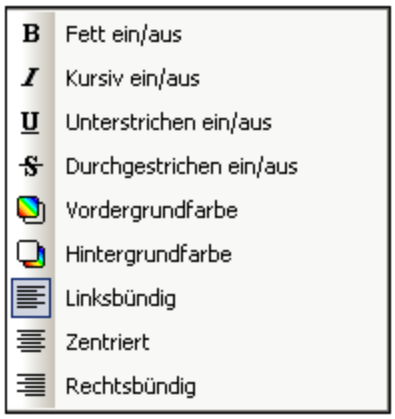


Mit dem Befehl **Gemischte Markup-Symbole einblenden** werden gemischte Markup Symbole in der Authentic-Ansicht angezeigt. Der Ersteller des StyleVision Power Stylesheet kann für die einzelnen Elemente/Attribute im Dokument entweder große, kleine oder gar keine Markup-Symbole definieren. Der Benutzer der Authentic-Ansicht sieht diese benutzerdefinierten Markup-Symbole im Ansichtsmodus für gemischte Markup-Symbole.

© 2016-2022 Altova GmbH

28.9.9 RichEdit

Wenn Sie den Mauszeiger über einen RichEdit-Befehl platzieren, wird ein Untermenü mit den RichEdit-Markup-Befehlen (*Abbildung unten*) angezeigt. Die Menübefehle in diesem Untermenü sind nur in der Authentic-Ansicht und nur, wenn der Cursor sich innerhalb eines als RichEdit-Komponente erstellten Elements befindet, aktiv.



Wenn Sie auf einen RichEdit-Befehl klicken, werden die Textstileigenschaften des RichEdit-Menüs auf den ausgewählten Text angewendet. Der Benutzer der Authentic-Ansicht kann neben der in der Authentic-Symbolleiste definierbaren Schriftart und -größe die Schriftbreite, den Schriftstil, die Schriftverzierung, die Farbe, Hintergrundfarbe und Ausrichtung des ausgewählten Texts definieren.

28.9.10 Zeile anhängen/einfügen/kopieren/löschen



Mit dem Befehl **Zeile anhängen** wird eine Zeile an die Tabelle in der Authentic-Ansicht angehängt.



Mit dem Befehl **Zeile einfügen** wird eine Zeile in die Tabelle in der Authentic-Ansicht eingefügt.



Mit dem Befehl **Zeile kopieren** wird die markierte Zeile in der Tabelle in der Authentic-Ansicht dupliziert.



Mit dem Befehl **Zeile löschen** wird in der Authentic-Ansicht die markierte Zeile aus der Tabelle gelöscht.

28.9.11 Zeile verschieben, Zeile löschen

In der Authentic-Ansicht sind die folgenden Zeilenbefehle aktiv:

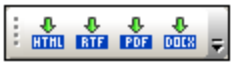
- Mit dem Befehl **Zeile nach oben** wird in der Authentic-Ansicht die markierte Zeile um eine Zeile nach oben verschoben.
- Mit dem Befehl **Zeile nach unten** wird in der Authentic-Ansicht die markierte Zeile um eine Zeile nach unten verschoben.
- Mit **Zeile löschen** wird die aktuelle Zeile gelöscht.

28.9.12 HTML-, RTF-, PDF- Word 2007+-, Text-Dokument generieren

Diese fünf Befehle dienen zum Generieren von Ausgabedokumenten anhand eines in einer PXF-Datei gespeicherten XML-Dokuments in der Authentic-Ansicht:

- **HTML-Dokument generieren**
- **RTF-Dokument generieren**
- **PDF-Dokument generieren**
- **Word 2007+-Dokument generieren**
- **Textdokument generieren**

Diese Befehle stehen auch über die Symbolleiste "Portable XML Form (PXF)" (*Abbildung unten*) zur Verfügung.



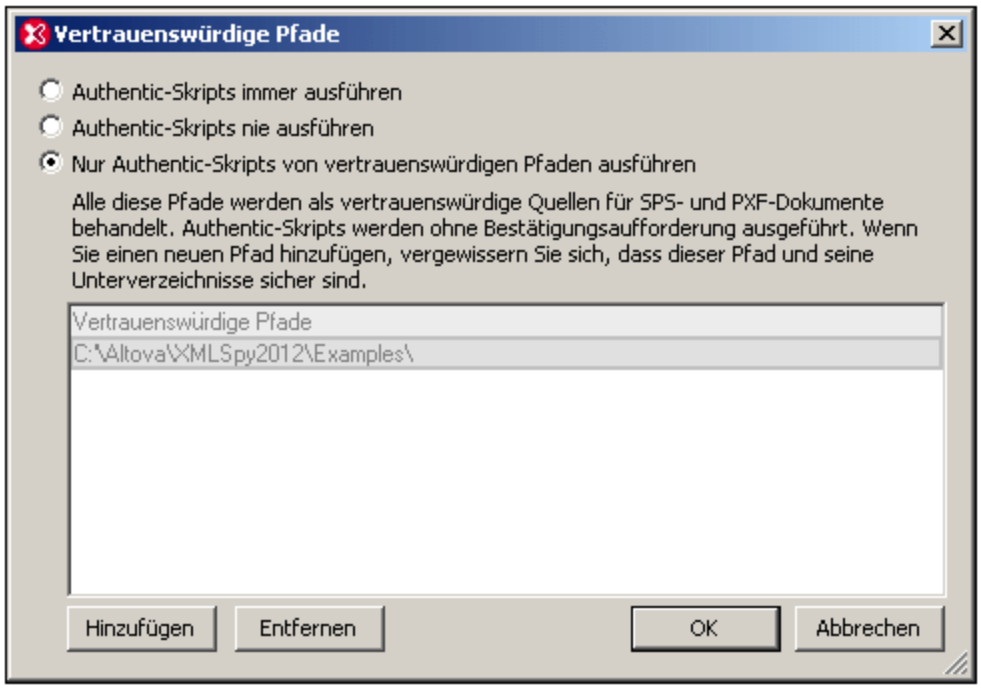
Durch Klicken auf den jeweiligen Befehl wird ein HTML-, RTF-, PDF- bzw. DocX-Ausgabedokument generiert.

Diese Schaltflächen sind aktiv, wenn eine PXF-Datei in der Authentic-Ansicht geöffnet ist. Die einzelnen Befehle und Schaltflächen sind dann aktiviert, wenn die PXF-Datei auch das XSLT Stylesheet für das jeweilige Ausgabeformat enthält. Wenn die PXF-Datei z.B. die XSLT Stylesheets für HTML und RTF enthalten, sind nur die Befehle und Symbolleistenschaltflächen für HTML und RTF aktiviert, während diejenigen für die Text-, PDF- und DocX (Word 2007+)-Ausgabe deaktiviert sind.

28.9.13 Vertrauenswürdige Pfade

Mit dem Befehl "Vertrauenswürdige Pfade" wird das gleichnamige Dialogfeld (*Abbildung unten)* aufgerufen, in dem Sie die Sicherheitseinstellungen für die Verwendung von Skripts in einem SPS vornehmen können. In diesem Dialogfeld können Sie festlegen, ob ein Skript in einem SPS, auf dem eine XML-Datei basiert, ausgeführt werden darf oder nicht, wenn Sie bei Anzeige dieser XML-Datei in die Authentic-Ansicht wechseln.

© 2016-2022 Altova GmbH

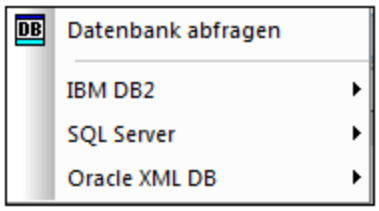


Es stehen die folgenden drei Optionen zur Auswahl:

- Authentic-Skripts werden immer ausgeführt, wenn eine Datei in der Authentic-Ansicht geöffnet wird.
- Authentic-Skripts werden nie ausgeführt, wenn eine Datei in der Authentic-Ansicht geöffnet wird.
- Nur Authentic-Skripts aus vertrauenswürdigen Pfaden werden ausgeführt. Die Liste der vertrauenswürdigen Pfade wird im unteren Bereich angezeigt. Über die Schaltfläche **Hinzufügen** können Sie zu einem Ordner navigieren und diesen zur Liste hinzufügen. Um einen Eintrag aus der Liste zu entfernen, wählen Sie ihn in der Liste der vertrauenswürdigen Pfade aus und klicken Sie auf **Entfernen.**

28.10 Menü "DB"

Das Menü **DB** ist das Menü für Datenbankoperationen. In der Abbildung unten wird das Menü mit seinen Menüeinträgen angezeigt. Die Befehle in den Untermenüs des Menüs "DB" sind in den Unterabschnitten dieses Abschnitts beschrieben.



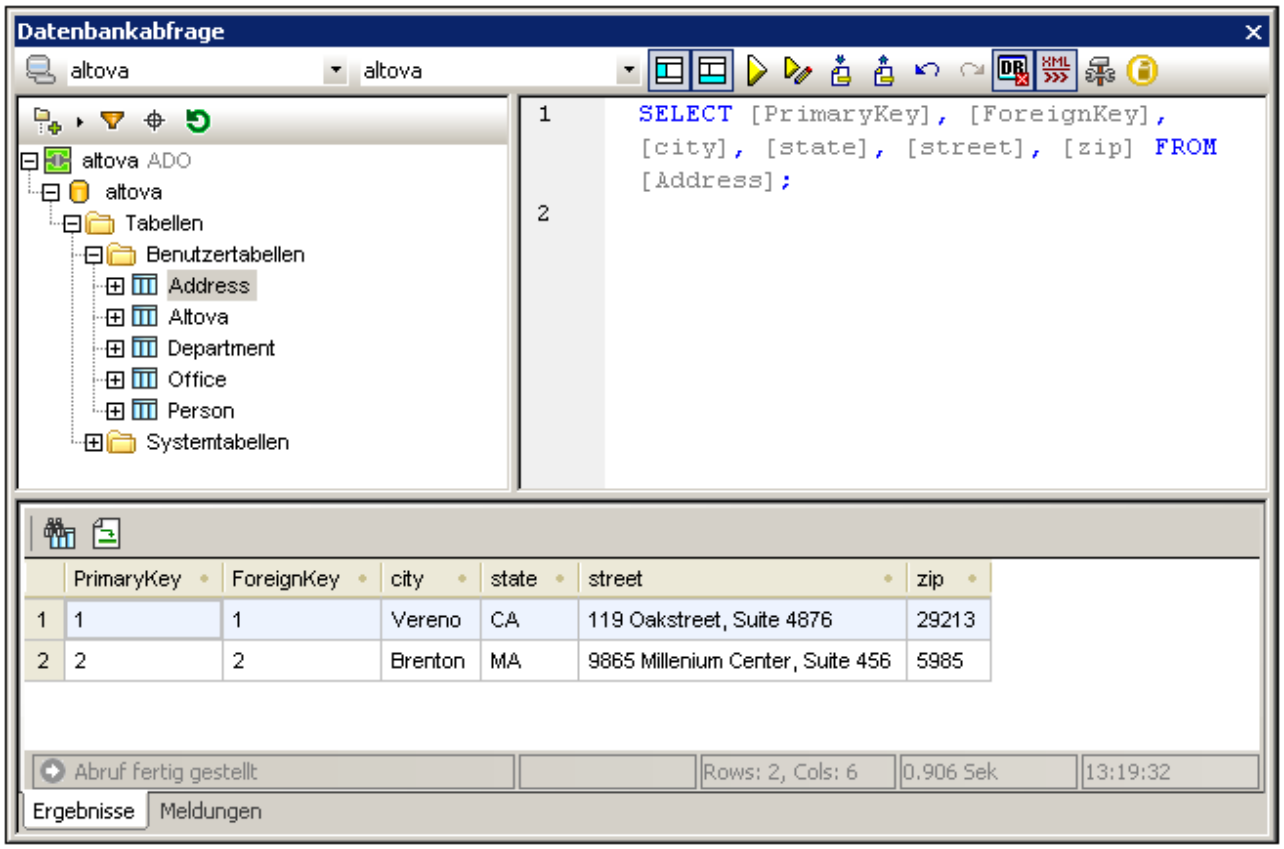
- Datenbank abfragen[1440]: Dient zum Abfragen verschiedener Datenbanken.
- IBM DB2[1456]: Enthält Befehle zur Unterstützung von IBM DB2-spezifischen Funktionalitäten.
- SQL Server[1462]: Enthält Befehle für SQL Server-Funktionalitäten
- Oracle-Datenbanken[1465]: Enthält Befehle für die Arbeit mit Oracle-Datenbanken.

Für alle in diesem Abschnitt beschriebenen Operationen ist eine Datenbankverbindung erforderlich. Informationen dazu finden Sie unter Herstellen einer Verbindung zu einer Datenbank[969]).

28.10.1 Datenbank abfragen

Mit dem Befehl **Datenbank abfragen** wird das Fenster "Datenbankabfrage" (*Abbildung unten*) geöffnet. Sobald dieses Fenster geöffnet wurde, können Sie es durch Auswahl des Befehls **DB | Datenbank abfragen** bzw. durch Klicken auf die Symbolleisten-Schaltfläche "Datenbank abfragen"  ein- und ausblenden.

© 2016-2022 Altova GmbH



Übersicht über das Fenster "Datenbankabfrage"

Das Fenster "Datenbankabfrage" besteht aus drei Teilen:

- einem **Browser-Fenster**[1445] links oben, in dem Verbindungsinformationen und Datenbanktabellen angezeigt werden.
- einem **Abfragefenster**[1449] rechts oben, in das die Abfrage eingegeben wird.
- einem mit Registern versehenen **Ergebnis-/Meldungsfenster**[1454]. Im Fenster "Ergebnisse" werden die Ergebnisabfrage im sogenannten Ergebnisraster angezeigt. Im Fenster "Meldungen" werden die Meldungen (einschl. Fehler- und Warnmeldungen), die bei der Ausführung der DB-Abfrage generiert werden, angezeigt.

Das Fenster "Datenbankabfrage" verfügt im oberen Bereich über eine Symbolleiste. Zwei der Schaltflächen in dieser Symbolleiste sind im Folgenden beschrieben. Die restlichen Schaltflächen werden im Abschnitt Fenster "Abfrage": Beschreibung und Funktionen[1449] beschrieben.

 Blendet das Browser-Fenster ein bzw. aus.



Blendet das Ergebnis-/Meldungsfenster ein bzw. aus.

Übersicht über die Mechanismen zur Abfrage von Datenbanken

Die Vorgangsweise bei der Abfrage von Datenbanken ist die folgende. Sie wird in den Unterabschnitten dieses Abschnitts näher beschrieben.

1. Über das Fenster "Datenbankabfrage" wird eine Verbindung zur Datenbank hergestellt[1442]. Unterstützt werden die Datenbanken: MS Access 2000 und 2003; Microsoft SQL Server; Oracle; MySQL; Sybase; und IBM DB2.
2. Die Datenbank, zu der eine Verbindung hergestellt wurde, bzw. Teile dieser Datenbank werden im Browser-Fenster[1445] angezeigt. Dieses Fenster kann je nach gewünschter Ansicht angepasst werden.
3. Im Abfragefenster[1453] wird eine Abfrage[1453] gemäß den für die gewünschte Datenbank benötigten Syntaxregeln eingegeben und ausgeführt.
4. Die Ergebnisse der Abfrage[1454] können mit verschiedenen Filtern gefiltert, bearbeitet und wieder in der Datenbank gespeichert werden.

28.10.1.1 Datenquellen

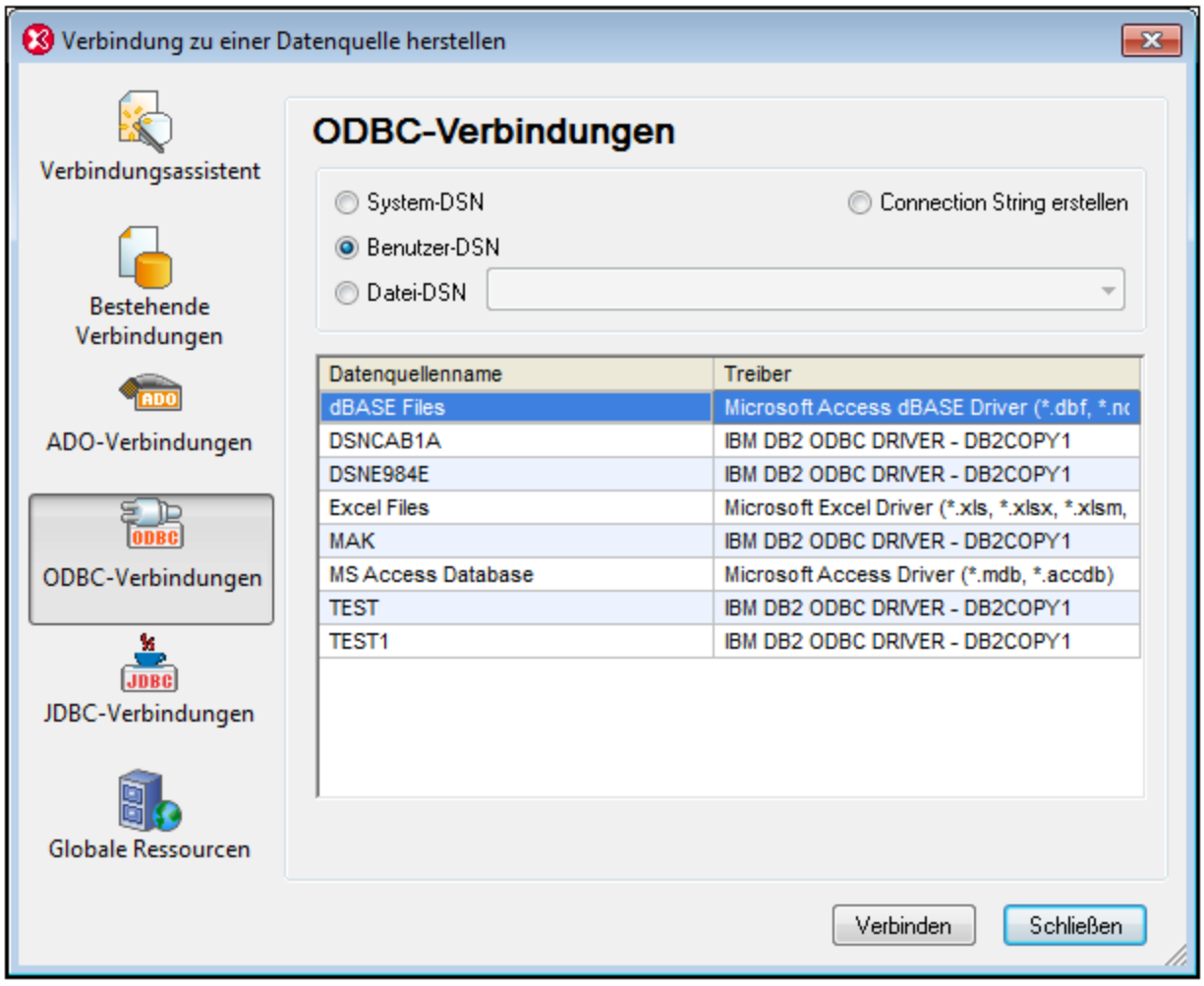
Zur Ausführung einer Datenbankabfrage müssen Sie zuerst eine Verbindung zur gewünschten Datenbank herstellen. In diesem Abschnitt wird beschrieben, wie Sie:

- eine Verbindung zu einer Datenbank herstellen und
- die gewünschte Datenquelle und das Root-Objekt aus mehreren bestehenden Verbindungen auswählen.

Herstellen einer Verbindung zu einer Datenbank

Wenn Sie zum ersten Mal in einer Sitzung (oder wenn keine Datenbankverbindung besteht) auf den Befehl **Datenbank abfragen** klicken, wird das Dialogfeld "Verbindungsassistent" (*Abbildung unten*) aufgerufen, über das Sie eine Verbindung zu einer Datenbank herstellen können. Um später weitere Verbindungen herzustellen, klicken Sie im Fenster "Datenbankabfrage" auf die Schaltfläche "Verbindungsassistent" .

© 2016-2022 Altova GmbH



Eine Beschreibung zum Herstellen einer Verbindung zu einer Datenbank über das Dialogfeld "Verbindungsassistent" wird im Abschnitt Herstellen einer Verbindung zu einer Datenbank 969 beschrieben.

Die nachstehende Tabelle enthält eine Liste aller unterstützten Datenbanken. Altova ist bestrebt, auch Unterstützung für andere Datenbanken zu gewährleisten, doch wurde die erfolgreiche Herstellung der Verbindung und Verarbeitung der Daten nur für die aufgelisteten Datenbanken getestet. Wenn es sich bei Ihrer Altova-Applikation um eine 64-Bit-Version handelt, stellen Sie sicher, dass Sie Zugriff auf die 64-Bit-Datenbanktreiber haben, die für die entsprechenden Datenbank benötigt werden.

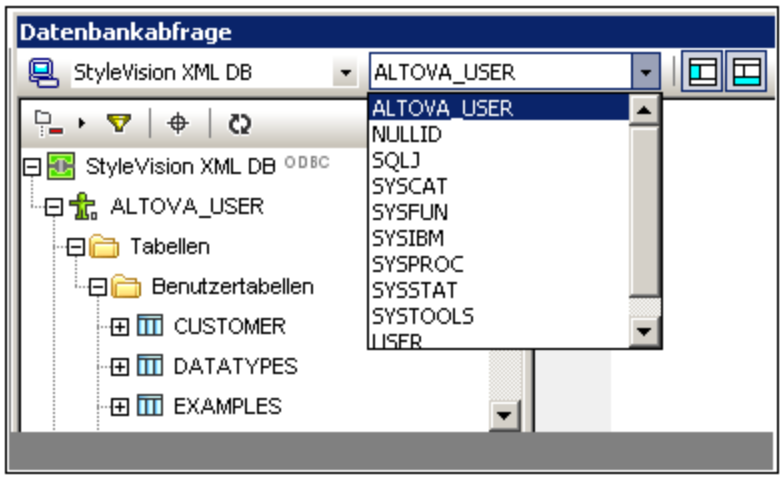
Datenbank	Anmerkungen
Firebird 2.x, 3.x	
IBM DB2 8.x, 9.x, 10.x, 11.x	
IBM Db2 for i 6.x, 7.4	Logische Dateien werden unterstützt und als Ansichten angezeigt.

Datenbank	Anmerkungen
IBM Informix 11.70 und höher	
MariaDB 10 und höher	
Microsoft Access 2003 und höher	Zum Zeitpunkt der Verfassung dieser Dokumentation (Anfang September 2019) gibt es kein Microsoft Access Runtime für Access 2019. Sie können nur dann eine Verbindung von Altova-Produkten zu einer Access 2019-Datenbank herstellen, wenn Microsoft Access 2016 Runtime installiert ist und der Datentyp "Large Number" in der Datenbank nicht verwendet wird.
Microsoft Azure SQL-Datenbank	SQL Server 2016 Codebase
Microsoft SQL Server 2005 und höher Microsoft SQL Server unter Linux	
MySQL 5 und höher	
Oracle 9i und höher	
PostgreSQL 8 und höher	PostgreSQL-Verbindungen werden sowohl als native Verbindungen als auch als treiberbasierte Verbindungen über Schnittstellen (Treiber) wie ODBC oder JDBC unterstützt. Für native Verbindungen werden keine Treiber benötigt.
Progress OpenEdge 11.6	
SQLite 3.x	SQLite-Verbindungen werden als native, direkte Verbindungen zur SQLite-Datenbankdatei unterstützt. Es sind keine separaten Treiber erforderlich. Daten, die aus einer SQLite-Datenbank stammen, können in der Authentic-Ansicht nicht bearbeitet werden. Wenn Sie versuchen, SQLite-Daten über die Authentic-Ansicht zu speichern, erscheint eine entsprechende Meldung dazu.
Sybase ASE15, 16	
Teradata 16	

Auswahl der erforderlichen Datenquelle

Alle bestehenden Verbindungen und ihre Root-Objekte werden in zwei Auswahllisten in der Symbolleiste des Fensters "Datenbankabfrage" (*Abbildung unten*) aufgelistet.

© 2016-2022 Altova GmbH



In der Abbildung oben wurde die Datenbank mit dem Namen `StyleVision DB` ausgewählt. Von den verfügbaren Root-Objekten für diese Datenbank wurde das Root-Objekt `ALTOVA_USER` ausgewählt. Die Datenbank und das Root-Objekt werden anschließend im Browser-Fenster angezeigt.

28.10.1.2 Browser-Fenster: Anzeigen der DB-Objekte

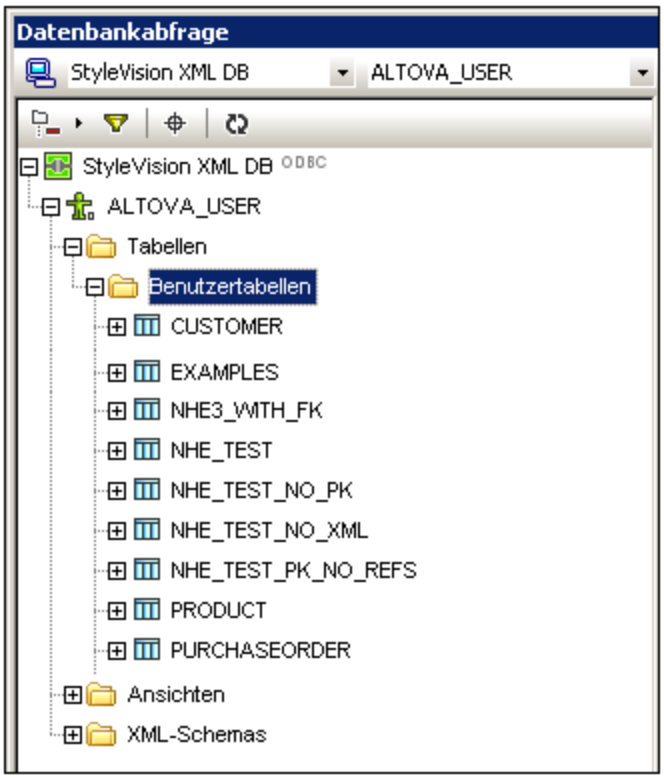
Im Browser-Fenster sehen Sie eine Übersicht über die Objekte in der ausgewählten Datenbank. Diese enthält Datenbank-Constraint-Informationen, z.B. ob es sich bei einer Spalte um einen Primär- oder Sekundärschlüssel handelt. In Datenbanken vom Typ IBM DB2 Version 9 sehen Sie im Browser zusätzlich registrierte XML-Schemas in einem separaten Ordner.

In diesem Abschnitt wird Folgendes beschrieben:

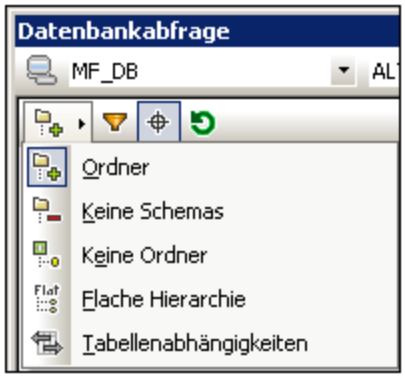
- Die im Browser-Fenster verfügbaren Layouts 1445.
- Wie Datenbankobjekte gefiltert werden 1447.
- Wie Datenbankobjekte gesucht werden 1448.

Browser-Fenster Layouts

Im Standard-Ordner-Layout werden Datenbankobjekte hierarchisch angezeigt. Je nachdem, welches Objekt ausgewählt ist, stehen unterschiedliche Kontextmenüoptionen zur Verfügung, wenn Sie mit der rechten Maustaste auf ein Objekt klicken.



Um ein Layout für den Browser auszuwählen, klicken Sie in der Symbolleiste des Browser-Fensters auf die Schaltfläche "Layout" und wählen Sie das Layout aus der Dropdown-Liste aus (*Abbildung unten*). Beachten Sie, dass sich die Schaltfläche ändert, je nachdem, welches Layout Sie ausgewählt haben.



Die verfügbaren Layouts sind:

- *Ordner:* Gliedert die Datenbankobjekte auf Basis des Objekttyps in hierarchisch angelegten Ordnern. Dies ist die Standardeinstellung.
- *Keine Schemas:* Ähnlich dem Ordner-Layout, mit der Ausnahme, dass es keine Ordner für Datenbank-Schemas gibt; Tabellen werden daher nicht nach dem Datenbankschema gegliedert.
- *Keine Ordner:* Zeigt die Datenbankobjekte in einer Hierarchie an, ohne Ordner zu verwenden.

© 2016-2022 Altova GmbH

- *Flach:* Gliedert die Datenbankobjekte auf der ersten hierarchischen Ebene nach Typ. Anstatt dass Spalten z.B. in der entsprechenden Tabelle angezeigt werden, werden alle Spalten in einem separaten Spaltenordner angezeigt.
- *Tabellenabhängigkeiten:* Kategorisiert die Tabellen nach ihrer Beziehung zu anderen Tabellen. Es gibt Kategorien für Tabellen mit Sekundärschlüsseln, Tabellen, die durch Sekundärschlüssel referenziert werden und Tabellen, die keine Beziehung zu anderen Tabellen haben.

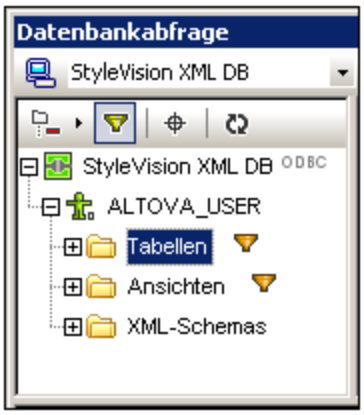
Um Tabellen nach Benutzer- und Systemtabellen zu sortieren, wechseln Sie ins Layout "Ordner", "Keine Schemas" oder "Flach", rechtsklicken Sie auf den Ordner "Tables" und wählen Sie den Befehl **Nach Benutzer- und Systemtabellen sortieren**. Die Tabellen werden in den Ordner "Benutzertabellen" und "Systemtabellen" in alphabetischer Reihenfolge sortiert.

Filtern von Datenbankobjekten

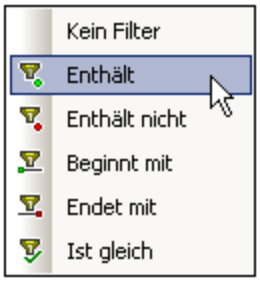
Im Browser-Fenster können Schemas, Tabellen und Ansichten (in allen Layouts mit Ausnahme des Layouts "Keine Ordner" und "Tabellenabhängigkeiten") nach Namen oder dem Teil eines Namens gefiltert werden. Die Objekte werden während der Eingabe der Zeichen gefiltert, dabei wird die Groß- und Kleinschreibung standardmäßig nicht beachtet.

So filtern Sie Objekte im Browser:

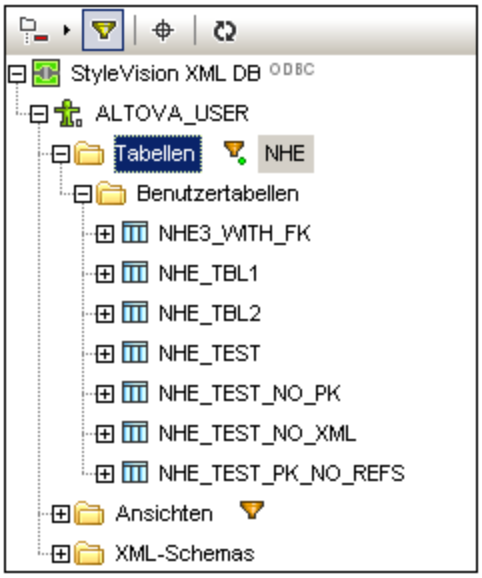
1. Klicken Sie in der Symbolleiste des Browser-Fensters auf die Schaltfläche "Ordnerinhalt filtern". Neben den Ordnern "Tables" und "Views" im ausgewählten Layout erscheinen Filtersymbole (*Abbildung unten*).



2. Klicken Sie auf das Filtersymbol neben dem gewünschten Ordner und wählen Sie aus dem Popup-Menü die Filteroption aus, z.B. `Enthält`.



3. Geben Sie im Eingabefeld, das nun angezeigt wird, den Filterstring ein (in der Abbildung unten ist der Filterstring für den Ordner `Tables` der String `NHE`). Der Filter wird während der Eingabe angewendet.

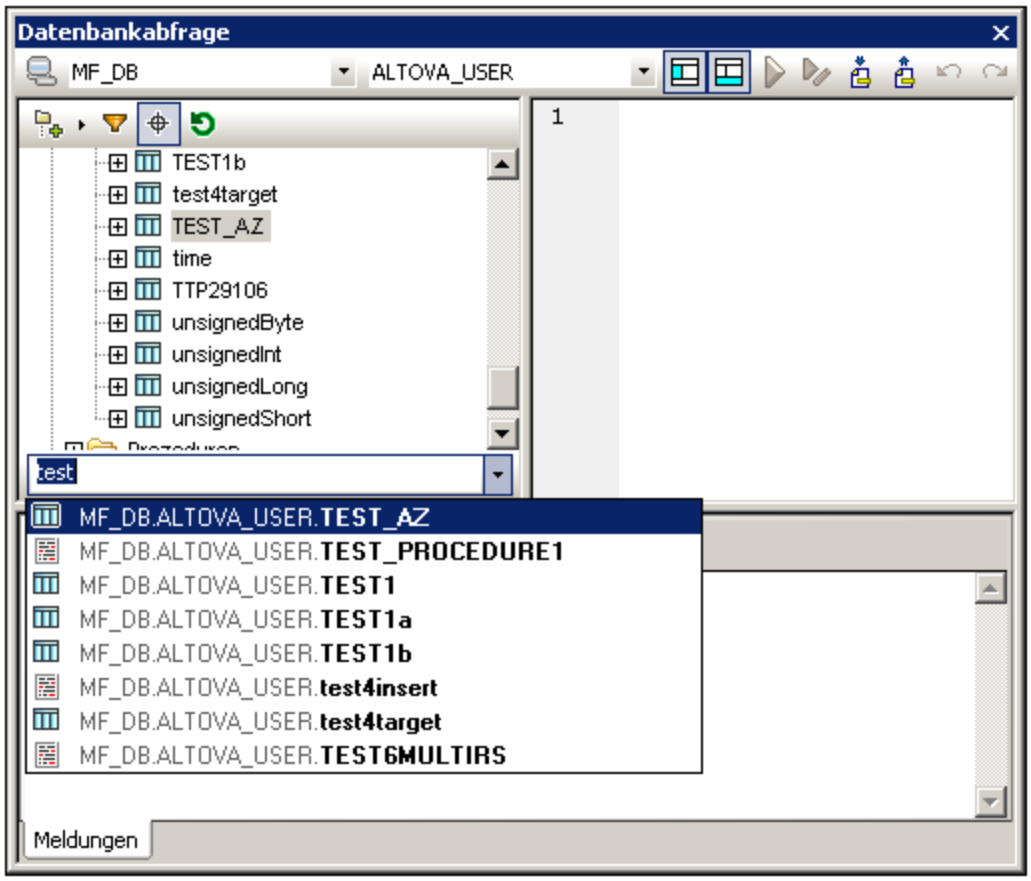


Suchen von Datenbankobjekten

Um ein bestimmtes Datenbankobjekt nach seinem Namen zu suchen, können Sie die Objektsuche des Browser-Fensters verwenden. Dies funktioniert folgendermaßen:

1. Klicken Sie in der Symbolleiste des Browser-Fensters auf die Schaltfläche "Objektsuche". Eine Dropdown-Liste erscheint am unteren Rand des Browsers.
2. Geben Sie den Suchstring, z.B. `test` in das Eingabefeld dieser Liste ein (*Abbildung unten*). Wenn Sie auf den Dropdown-Pfeil klicken, werden alle Objekte, die den Suchstring enthalten, angezeigt.

© 2016-2022 Altova GmbH



3. Klicken Sie auf das Objekt in der Liste, um es im Browser zu sehen.

28.10.1.3 Abfragefenster: Beschreibung und Funktionen

Das Abfragefenster ist ein intelligenter SQL Editor zur Eingabe von Abfragen für die ausgewählte Datenbank. Klicken Sie nach Eingabe der Abfrage im Fenster "Datenbankabfrage" auf die Schaltfläche "Ausführen", um die Abfrage auszuführen. Die Ergebnisse und die Ausführungsmeldungen werden im Ergebnis-/Meldungsfenster angezeigt[1454]. Eine Anleitung zum Arbeiten mit Abfragen finden Sie im nächsten Abschnitt, Abfragefenster: Arbeiten mit Abfragen[1453]. In diesem Abschnitt werden die Hauptfunktionen des Abfragefensters beschrieben.

- SQL Editor-Schaltflächen in der Datenbankabfrage-Symbolleiste
- SQL Editor-Optionen
- Autokomplettierung von SQL-Anweisungen
- Definition von Regionen in einem SQL Script
- Einfügen von Kommentaren in ein SQL-Script
- Verwendung von Lesezeichen

SQL Editor-Schaltflächen in der Datenbankabfrage-Symbolleiste

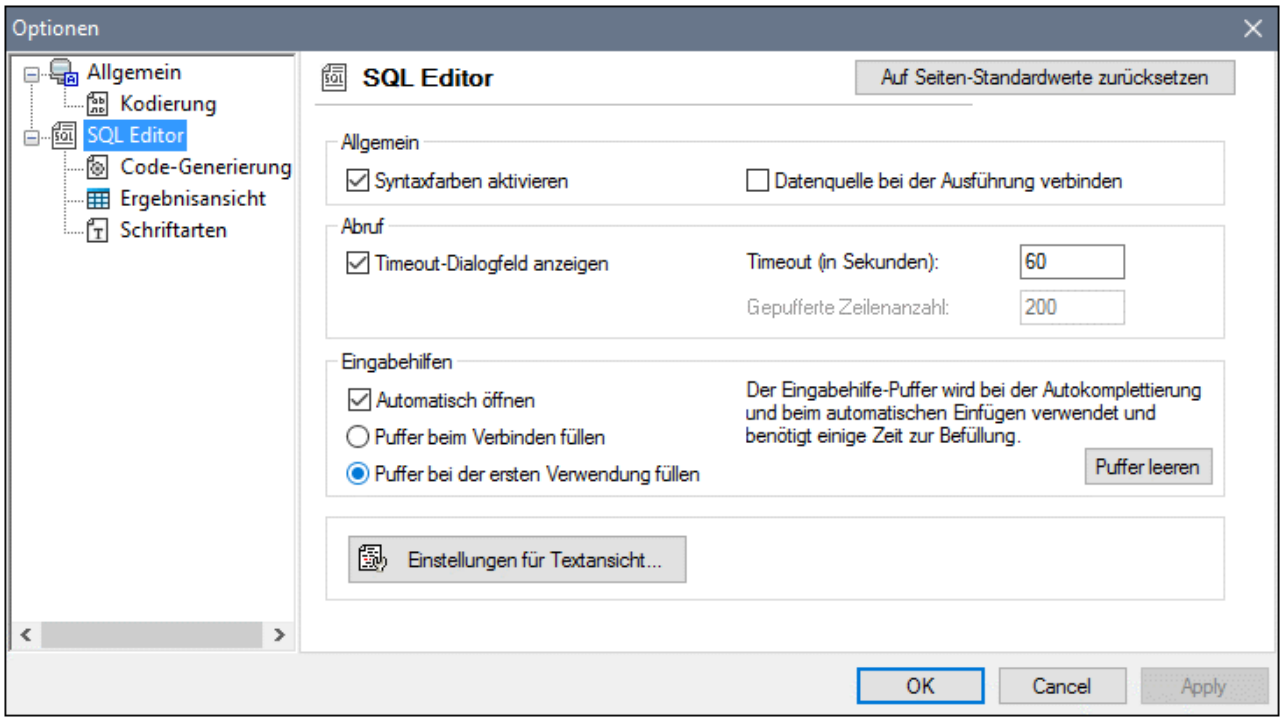
Für die Arbeit mit dem SQL Editor werden die folgenden Schaltflächen in der Symbolleiste des Fensters "Datenbankabfrage" verwendet:

	**Ausführen**	Führt die aktuell ausgewählte SQL-Anweisung aus. Wenn das Script mehrere Anweisungen enthält und keine ausgewählt wird, werden alle ausgeführt.
	**Zur Datenbankbearbeitung ausführen**	Wie der Befehl "Ausführen", mit der Ausnahme, dass die Ergebnisse (im Abschnitt "Ergebnisse") bearbeitet werden können.
	**SQL-Datei importieren**	Öffnet eine SQL-Datei im SQL Editor.
	**SQL-Datei exportieren**	Speichert SQL-Abfragen in einer SQL-Datei.
	**Rückgängig**	Macht unbegrenzt viele Bearbeitungen im SQL Editor rückgängig.
	**Wiederherstellen**	Stellt unbegrenzt viele Bearbeitungen im SQL Editor wieder her.
	**Datenbankabfragefenster beim Öffnen von XML-Dokumenten ausblenden**	Definiert, ob das Datenbankabfragefenster ausgeblendet werden soll, wenn ein XML-Dokument zur Bearbeitung geöffnet wird.
	**Alle im Ergebnisraster vorgenommenen Änderungen beim Speichern der XML-Dokumentinstanz automatisch übernehmen**	Wenn ein bearbeitetes XML-Dokument in XMLSpy gespeichert wird, werden die Änderungen in die Datenbank geschrieben, wenn dieser Ein-/Aus-Befehl aktiv ist. Andernfalls müssen die Änderungen im Ergebnisfenster explizit mit "Commit" übernommen werden.
	**Optionen**	Öffnet das Dialogfeld "Optionen" im SQL Editor.
	**SQL-Script in DatabaseSpy öffnen**	Öffnet das SQL Script in Altova DatabaseSpy.

Optionen

Wenn Sie in der Datenbankabfrage-Symbolleiste auf die Schaltfläche "Optionen" klicken, wird das Dialogfeld "Optionen" angezeigt (*Abbildung unten*). Im linken Bereich können Sie eine Einstellungsseite auswählen. Auf der jeweiligen Seite können Sie Optionen ausgewählt werden. Klicken Sie auf die Schaltfläche **Seite auf Standardeinstellungen zurücksetzen**, um die Standardeinstellungen für die jeweilige Seite wiederherzustellen.

© 2016-2022 Altova GmbH



Die wichtigsten Einstellungen sind die folgenden:

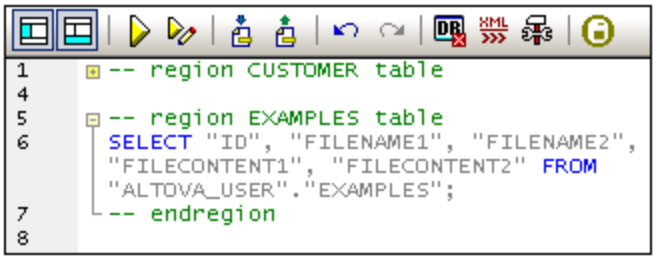
- ***Allgemein | Kodierung:*** Optionen zum Einstellen der Kodierung für neue SQL-Dateien und bestehende SQL-Dateien, bei denen die Kodierung nicht erkannt wird. (Wenn die Kodierung bestehender SQL-Dateien erkannt wird, werden die Dateien geöffnet und gespeichert, ohne dass die Kodierung geändert wird.)
- ***SQL Editor:*** Optionen zum Aktivieren und Deaktivieren der Syntaxfärbung und der Herstellung einer Verbindung zur Datenquelle bei Ausführung der SQL-Anweisung. Sie können für die Abfrageausführung ein Timeout einstellen und bei Überschreitung des Timeout ein Dialogfeld anzeigen lassen, um die Timeout-Einstellungen zu ändern. Eingabehilfen bezieht sich auf die Eingabehilfen, die im Rahmen der Autokomplettierungsfunktion angezeigt werden. Wenn Sie eine SQL-Anweisung eingeben, wird im Editor eine Liste kontextsensitiver Autokomplettierungsvorschläge angezeigt. Diese Vorschläge können so eingestellt werden, dass sie automatisch angezeigt werden. Wenn die automatische Anzeige deaktiviert ist, können Sie die Vorschläge im SQL Editor durch Drücken von **Strg+Leerzeichen** anzeigen lassen. Der Puffer für die Informationen in den Eingabehilfen kann entweder beim Herstellen der Verbindung zur Datenquelle befüllt werden oder, wenn er das erste Mal benötigt wird. Mit der Schaltfläche "Einstellungen für Textansicht 1506" wird das Fenster mit den Optionen für die Textansicht von XMLSpy geöffnet.
- ***SQL Editor | SQL-Generierung:*** Die Applikation generiert SQL-Anweisungen, wenn Sie Objekte aus dem Browser-Fenster in das Abfragefenster ziehen. Die Optionen für die Generierung von SQL-Anweisungen können auf dem Register "SQL-Generierung" konfiguriert werden. Im Bereich *Datenbank* können Sie eine Datenbankart auszuwählen die Optionen für die Generierung von Anweisungen einzeln für die verschiedenen Datenbankarten, mit denen Sie arbeiten, konfigurieren. Wenn Sie das Kontrollkästchen *Auf alle Datenbanken anw.* aktivieren, werden die derzeit ausgewählten Optionen auf alle Datenbanken angewendet. Unter anderem stehen Optionen zum Anhängen von Semikola an Anweisungen und zum Setzen der Identifier innerhalb von Escape-Zeichen zur Verfügung. Wenn das Kontrollkästchen *Semikola an Anweisungsende anhängen* aktiviert ist, wird bei Generierung einer SQL-Anweisung im SQL-Editor ein Semikolon angehängt. Beachten Sie, dass die Bearbeitung von Daten in

Oracle-Datenbanken und IBM iSeries- und DB2-Datenbanken über JDBC nur möglich ist, wenn dieses Kontrollkästchen nicht aktiviert ist.

- ***SQL Editor | Ergebnisansicht:*** Optionen zum Konfigurieren des Ergebnisregisters.
- ***SQL Editor | Schriftarten:*** Optionen zum Einstellen der Schriftart für den Text im Text-Editor und in der Ergebnisansicht.

Definition von Regionen in einem SQL Script

Regionen sind Abschnitte in SQL Scripts, die als Einheit markiert und deklariert sind. Regionen können erweitert und reduziert werden, um Teile von SQL Scripts ein- oder auszublenden. Regionen können auch in andere Regionen verschachtelt werden. Regionen werden vor und nach dem ausgewählten Text durch den Kommentar `--region` bzw. `--endregion` gekennzeichnet. Regionen können auch optional einen Namen erhalten, der nach der `-- region` Markierung eingegeben wird (*siehe Abbildung unten*).



Um eine Region einzufügen, wählen Sie die gewünschte(n) Anweisungen aus, die zu einer Region gemacht werden sollen, rechtsklicken Sie und wählen Sie den Befehl "Region einfügen". Es wird eine erweiterbare/reduzierbare Region erstellt. Fügen Sie ggf. einen Namen hinzu. Beachten Sie in der Abbildung oben auch die Zeilennummerierung. Um eine Region zu löschen, löschen Sie die beiden `--region` und `--endregion` Markierungen.

Einfügen von Kommentaren in ein SQL-Script

Text in einem SQL-Script kann auskommentiert werden. Diese Script-Abschnitte werden bei der Ausführung des Scripts übersprungen.

- Um einen Block auszukommentieren, markieren Sie den Block, rechtsklicken Sie und wählen Sie den Befehl **Blockkommentar einfügen/entfernen**. Um den Blockkommentar zu entfernen, rechtsklicken Sie und wählen Sie den Befehl **Blockkommentar einfügen/entfernen**.
- Um eine Zeile oder den Teil einer Zeile auszukommentieren, platzieren Sie den Cursor an die Stelle, an der der Zeilenkommentar beginnen soll, rechtsklicken Sie und wählen Sie den Befehl **Zeilenkommentar einfügen/entfernen**. Um den Zeilekommentar zu entfernen, markieren Sie den Kommentar, rechtsklicken Sie und wählen Sie den Befehl **Zeilenkommentar einfügen/entfernen**

Verwendung von Lesezeichen

Lesezeichen können an bestimmten Zeilen eingefügt werden. Sie können anschließend durch die Lesezeichen im Dokument navigieren. Um ein Lesezeichen einzufügen, platzieren Sie den Cursor in die gewünschte Zeile, rechtsklicken Sie und wählen Sie den Befehl **Lesezeichen einfügen/löschen**. Um zum nächsten oder vorhergehenden Leszeichen zu navigieren, wählen Sie den Befehl **Nächstes Lesezeichen** bzw. **Vorheriges Lesezeichen**. Um ein Lesezeichen zu entfernen, platzieren Sie den Cursor in die Zeile, in der das Lesezeichen entfernt werden soll, rechtsklicken Sie und wählen Sie den Befehl **Lesezeichen einfügen/löschen.** Um alle Lesezeichen zu entfernen, rechtsklicken Sie und wählen Sie den Befehl **Alle Lesezeichen löschen.**

© 2016-2022 Altova GmbH

28.10.1.4 Abfragefenster: Arbeiten mit Abfragen

Nachdem Sie eine Verbindung zu einer Datenbank hergestellt haben, kann ein SQL-Script im SQL Editor eingegeben und ausgeführt werden. In diesem Abschnitt wird beschrieben:

- wie ein SQL-Script in SQL Editor eingegeben wird.
- wie das Script im Fenster "Datenbankabfrage" ausgeführt wird.

In diesem Abschnitt werden die folgenden Schaltflächen verwendet:

	**Abfrage Ausführen**	Führt die aktuell ausgewählte SQL-Anweisung aus. Wenn das Script mehrere Anweisungen enthält und keine ausgewählt wird, werden alle ausgeführt.
	**Zur Datenbearbeitung ausführen**	Wie der Befehl "Ausführen", mit der Ausnahme, dass die Ergebnisse (auf dem Register "Ergebnisse") bearbeitet werden können.
	**SQL-Datei importieren**	Öffnet eine SQL-Datei im SQL Editor.

Erstellen von SQL-Anweisungen und Scripts im SQL Editor

Zum Erstellen von SQL-Anweisungen und Scripts können die folgenden GUI-Methoden verwendet werden:

- *Drag-and-Drop:* Ziehen Sie ein Objekt aus dem Browser-Fenster in den SQL Editor. Eine SQL-Anweisung wird generiert, um die Datenbank nach diesem Objekt abzufragen.
- *Kontextmenü:* Rechtsklicken Sie auf ein Objekt im Browser-Fenster und wählen Sie den Befehl **In SQL Editor anzeigen | Select**.
- *Manuelle Eingabe:* Geben Sie die SQL-Anweisung direkt im SQL Editor ein. Beim Bearbeiten hilft Ihnen die Autokomplettierungsfunktion.
- *SQL-Script importieren:* Klicken Sie in der Symbolleiste des Fensters "Datenbankabfrage" auf die Schaltfläche **SQL-Datei importieren.**

Ausführen von SQL-Anweisungen

Wenn das SQL-Script im SQL Editor mehr als eine SQL-Anweisung enthält, wählen Sie die gewünschte Anweisung aus und klicken Sie in der Symbolleiste des Fensters "Datenbankabfrage" entweder auf die Schaltfläche **Ausführen** oder auf **Zur Datenbearbeitung ausführen**. Wenn im SQL-Script keine Anweisung ausgewählt ist, werden alle Anweisungen im Script ausgeführt. Die Datenbankdaten werden abgerufen und in einem Raster auf dem Ergebnisregister angezeigt[1454]. Wenn Sie den Befehl **Zur Datenbearbeitung ausführen** gewählt haben, können die abgerufenen Daten im Ergebnisregister bearbeitet werden[1454]. Meldungen über die Ausführung der Anweisung werden auf dem Meldungsregister[1454] angezeigt.

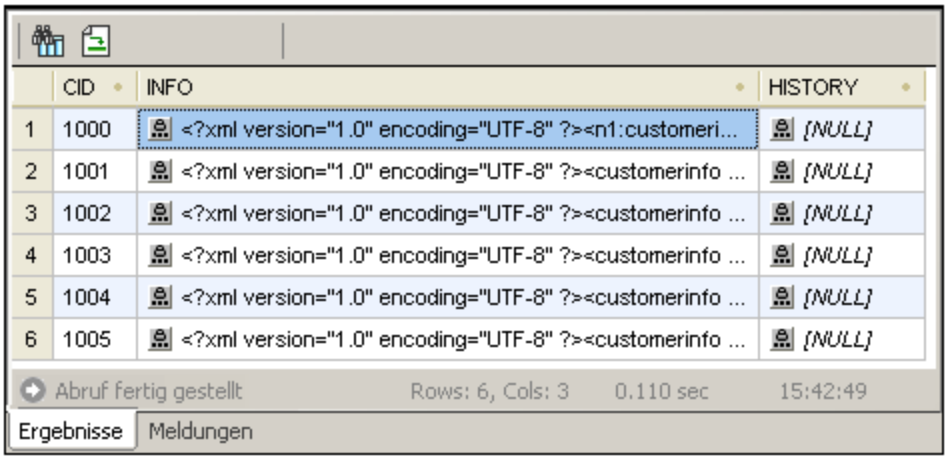
28.10.1.5 Ergebnisse und Meldungen

Das Ergebnis-/Meldungsfenster hat zwei Register:

- Auf dem Register "Ergebnisse" 1454 werden die von der Abfrage abgerufenen Daten angezeigt.
- Auf dem Register Register "Meldungen" 1456 sehen Sie Meldungen über die Ausführung der Abfrage.

Register "Ergebnisse"

Die von der Abfrage abgerufenen Daten werden in Tabellenform auf dem Ergebnisregister angezeigt (*Abbildung unten*). Wenn die Abfrageergebnisse mit Hilfe des Befehls **Abfrage ausführen** generiert wurden, werden die XML-Dokumente auf dem Ergebnisregister mit dem XML-Symbol (*Abbildung unten*) gekennzeichnet. Wenn der Befehl **Zur Datenbearbeitung ausführen** verwendet wurde, werden die XML-Dokumente mit dem Symbol für editierbare XML-Daten angezeigt.

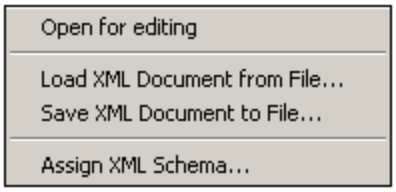


Die folgenden Operationen können auf dem Ergebnisregister über das Kontextmenü, das angezeigt wird, wenn Sie auf dem Ergebnisregister an der entsprechenden Stelle auf die rechte Maustaste klicken, ausgeführt werden:

- *Nach einer Spalte sortieren:* Rechtsklicken Sie an eine beliebige Stelle in der Spalte, nach der die Ergebnisse sortiert werden sollen und wählen Sie den Befehl **Sortierung | Aufsteigend/Absteigend/Standardeinstellung wiederherstellen**.
- *Kopieren in die Zwischenablage:* Dieser Vorgang erfolgt in zwei Schritten: (i) Auswählen des Datenbereichs; und (ii) Kopieren der Auswahl. Die Daten können auf verschiedene Arten ausgewählt werden: (i) durch Klicken auf eine Spaltenüberschrift zur Auswahl der Spalte bzw. Zeile; (ii) durch Auswählen einzelner Zellen (Zur Auswahl mehrerer Zellen verwenden Sie **Umschalt** und/oder **Strg**); (iii) durch Rechtsklick auf eine Zelle und Auswahl von **Auswahl | Zeile/Spalte/Alle**. Klicken Sie anschließend auf die rechte Maustaste und wählen Sie den Befehl **Ausgewählte Zellen kopieren.** Daraufhin wird die Auswahl in die Zwischenablage kopiert, von wo sie in andere Applikationen eingefügt werden kann.
- *Anhängen einer neuen Zeile:* Wenn die Abfrage zur Bearbeitung ausgeführt wurde, klicken Sie mit der rechten Maustaste an eine beliebige Stelle im Ergebnisfenster, um den Befehl **Zeile anhängen** aufzurufen..
- *Löschen einer Zeile:* Wenn die Abfrage zur Bearbeitung ausgeführt wurde, rechtsklicken Sie an eine beliebige Stelle in einer Zeile, um den Befehl **Zeile löschen** aufzurufen.

© 2016-2022 Altova GmbH

- *Bearbeiten von Datensätzen:* Wenn die Abfrage zur Datenbearbeitung ausgeführt wurde, können einzelne Felder bearbeitet werden. Um die Änderungen zu übernehmen, klicken Sie in der Symbolleiste des Ergebnisregisters auf die Schaltfläche "Übernehmen".
- *Bearbeiten von XML-Datensätzen:* Diese Funktion wird für IBM DB2-, SQLServer-, PostgreSQL- (8.3 und höher) sowie Oracle- (9 und higher) Datenbanken unterstützt und nur bei DB-Tabellen, die einen Primärschlüssel haben. Wenn die Abfrage zur Datenbearbeitung ausgeführt wurde und wenn es sich bei einem editierbaren Feld um ein XML-Feld handelt, wird bei Klick auf das Symbol für editierbares XML im Ergebnisraster das Menü"XML bearbeiten" geöffnet (*Abbildung unten*). Sie können ein XML-Feld auch für die Datenbearbeitung öffnen, indem Sie mit der rechten Maustaste im Bereich "Ordner" auf das XML-Feld klicken und den Befehl **Daten bearbeiten** auswählen.



Mit dem Befehl **Zur Bearbeitung öffnen** wird das XML-Dokument in einem XMLSpy Fenster geöffnet und das Symbol "Editierbares XML" ändert sich in das Symbol , in dem die drei Punkte rot erscheinen. Wenn dieses Dokument gespeichert wird und in der Datenbankabfrage-Symbolleiste die Schaltfläche "XML-Änderungen automatisch übernehmen" beim Öffnen des Dokuments aktiviert war, werden die Änderungen am XML-Dokument automatisch in der Datenbank übernommen. Andernfalls müssen gespeicherte Änderungen mit der Schaltfläche "Commit" des Ergebnisfensters übernommen werden. (Beachten Sie: Um zwischen dem XML-Dokumentfenster und dem Fenster "Datenbankabfrage" zu wechseln, müssen Sie den Befehl **DB | Datenbank abfragen** aufrufen). Mit dem Befehl **XML-Dokument aus Datei laden** wird ein externes XML-Dokument in das ausgewählte Feld in der Datenbank geladen. Mit dem Befehl **XML-Dokument in Datei speichern** wird das XML-Dokument im ausgewählten Datenbankfeld unter einem Dateipfad Ihrer Wahl gespeichert. Mit dem Befehl "XML-Schema zuweisen" wird das Dialogfeld "XML-Schema auswählen" 1460 aufgerufen, in dem Sie ein XML-Schema auswählen können, das Sie dem XML-Dokument zuweisen. Die Zuweisung wird in der Datenbank gespeichert. Die Zuweisung von XML-Schemas wird im Abschnitt IBM DB2 | Zuweisen eines XML-Schemas 1460 näher behandelt.

- *NULL definieren, Standard definieren, Änderungen für diese Zelle rückgängig machen:* Wenn die Abfrage zur Bearbeitung ausgeführt wurde, erhalten Sie bei Rechtsklick in eine Zelle Zugriff auf Befehle zum Setzen eines Null-Werts oder, falls definiert, eines Spalten-Standardwerts für diese Zelle. An einer Zelle vorgenommene Änderungen können mit dem Befehl **Änderungen für diese Zelle rückgängig machen** rückgängig gemacht werden; der aktuelle bearbeitete Wert wird durch den Wert aus der Datenbank ersetzt.

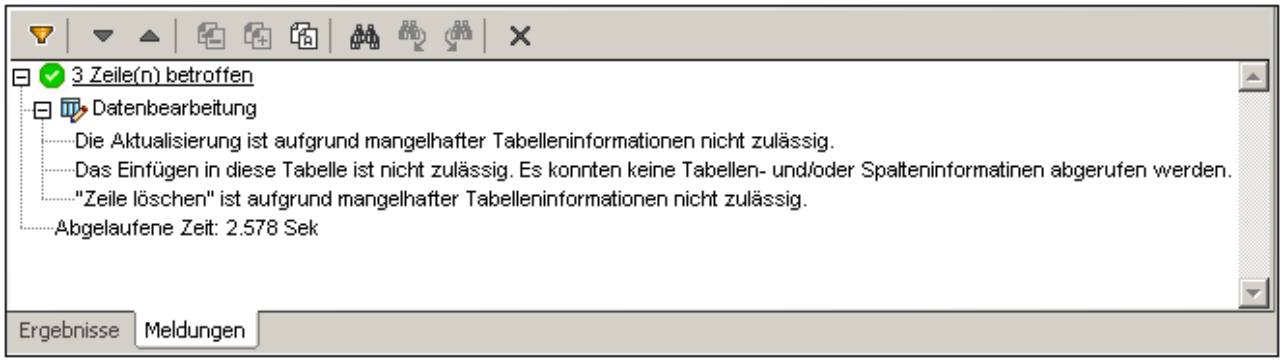
Das Ergebnisregister hat die folgenden Symbolleistenschaltflächen:

	Gehe zu Anweisung	Markiert die SQL Anweisung im SQL Editor, die das entsprechende Ergebnis erzeugt hat.
	Suchen	Sucht Text im Ergebnisfenster. Aus XML-Dokumentinhalt wird durchsucht.
	Neue Zeile hinzufügen	Fügt eine neue Zeile zum Ergebnisraster hinzu.

	Zeile löschen	Löscht die aktuelle Zeile im Ergebnisraster.
	Änderungen am Ergebnisraster rückgängig machen	Macht alle Änderungen am Ergebnisraster rückgängig.
Commit	**Übernehmen**	Übernimmt die im Ergebnisraster vorgenommenen Änderungen in der Datenbank.

Register "Meldungen"

Das Register "Meldungen" enthält Informationen über die zuvor ausgeführte SQL-Anweisung und gibt Fehlermeldungen und Warnmeldungen aus.



Die Symbolleiste des Meldungsregisters enthält Schaltflächen, mit denen Sie die Ansicht anpassen, darin navigieren sowie Meldungen in die Zwischenablage kopieren können. Mit Hilfe der Schaltfläche **Filter** können Sie bestimmte Arten von Meldungen ein- und ausblenden. Mit Hilfe der Schaltflächen **Nächste** und **Vorherige** wird die jeweils vorige bzw. nächste Meldung in der Liste markiert. Meldungen können auch mit oder ohne Ihre Sub-Einträge in die Zwischenablage kopiert werden, sodass sie in Dokumente eingefügt werden können. Mit Hilfe der Funktion **Suchen** können Sie einen Suchbegriff angeben und dann die Liste nach diesem Begriff durchsuchen. Mit der Schaltfläche **Löschen** können Sie den Inhalt des Meldungsfensters löschen.

Anmerkung: Diese Symbolleisten-Befehle stehen auch im Kontextmenü zur Verfügung.

28.10.2 IBM DB2

Das Menü **IBM DB2** enthält ein Untermenü mit den folgenden Befehlen (i) zum Registrieren und Löschen der Registrierung von Schemas bei einer IBM DB2 Datenbank (**Verwalten von XML-Schemas**[1457]), und (ii) zum Zuweisen von Schemas für die XML-Dateivalidierung (**Zuweisen eines XML-Schemas**[1460]).

Für beide Funktionen muss eine Verbindung zur jeweiligen IBM DB2-Datenbank bestehen. Wie man eine Verbindung zur Datenbank herstellt, ist im Beispiel IBM DB2 (ODBC)[1009]. beschrieben. In diesem Abschnitt wird beschrieben, wie Schemas in einer IBM DB2 Datenbank verwaltet und wie XML-Schemas einer Datenbank-XML-Datei zugewiesen werden.

Anmerkung: Das Ergebnisraster des Fensters "Datenbankabfrage"[1440] enthält wichtige Funktionen für die Arbeit mit XML-Dateien in IBM DB2-Datenbanken. Dazu gehören Funktionen zum Bearbeiten, Laden

© 2016-2022 Altova GmbH

von XML-Dateien in Datenbankzellen als XML-Dateien, externem Speichern von DB-XML-Dateien und Zuweisen von XML-Schemas zu DB-XML-Dateien.

28.10.2.1 XML-Schemas verwalten

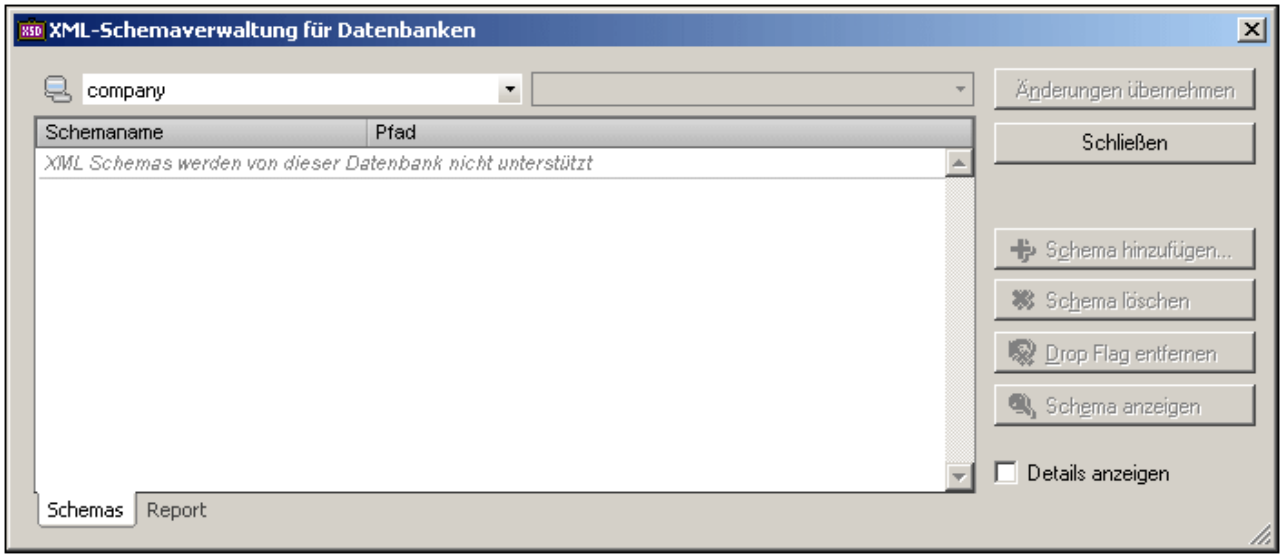
Mit dem Befehl **XML-Schemas verwalten** können Schemas aus einzelnen Datenbank-Schemas in einer IBM DB2-Datenbank hinzugefügt und gelöscht werden. Dazu müssen folgendes tun:

- eine Verbindung zu einer IBM DB2-Datenbank herstellen
- das/die Datenbank-Schema(s) auswählen, für das/die XML-Schemas hinzugefügt oder gelöscht werden sollen und
- anschließend die Schemaverwaltungsaktionen durchführen.

Diese Schritte sind im Folgenden beschrieben.

Herstellen einer Verbindung zur IBM DB2-Datenbank

Wenn Sie den Befehl **XML-Schemas verwalten** auswählen, wird das Dialogfeld "XML-Schemaverwaltung für Datenbanken" aufgerufen (*Abbildung unten*).

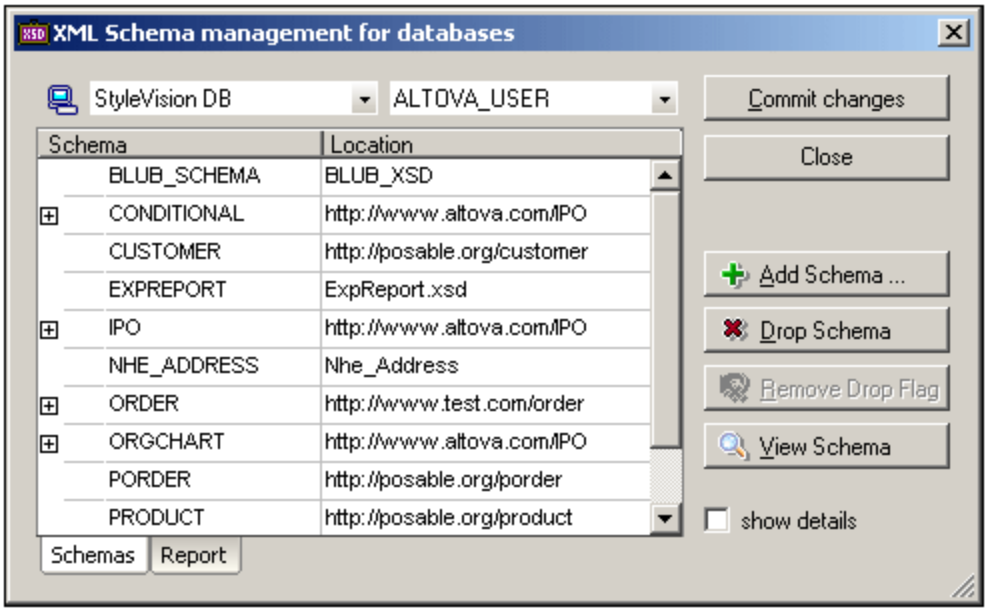


Um die Schemas einer IBM DB2-Datenbank zu verwalten, müssen Sie zuerst eine Verbindung zur Datenbank herstellen. Klicken Sie dazu im Dialogfeld auf die Schaltfläche "Verbindungsassistent" . Daraufhin wird das gleichnamige Dialogfeld aufgerufen, über das Sie die Verbindung zur Datenbank herstellen können. Die Verwendung des Verbindungsassistenten ist im Abschnitt Herstellen einer Verbindung zu einer Datenbank 969 beschrieben.

Anzeigen einer Liste von XML-Schemas

Nachdem die Verbindung zur IBM DB2-Datenbank hergestellt wurde, wird die Datenbank in der Auswahlliste auf der linken Seite (*siehe Abbildung unten*) aufgelistet, Wenn derzeit mehrere Verbindungen bestehen, können

Sie die gewünschte Datenbank in der Auswahlliste auswählen. In der Abbildung unten ist die Datenbank "StyleVision DB" ausgewählt.



In der Auswahlliste auf der rechten Seite sind alle Datenbankschemas der ausgewählten IBM DB2-Datenbank aufgelistet. Wenn ein Datenbankschema in dieser Auswahlliste ausgewählt ist, werden alle für das ausgewählte Datenbankschema registrierten XML-Schemas im Hauptfenster angezeigt. In der Abbildung oben werden alle mit dem Datenbankschema `Altova_User` registrierten Schemas zusammen mit Ihren Pfaden aufgelistet. Wenn Sie das Kontrollkästchen "Details anzeigen" aktivieren, werden im Hauptfenster weitere Datenspalten angezeigt.

Verwalten der XML-Schemas

Die Liste der Schemas im Hauptfenster enthält die für das ausgewählte Datenbankschema registrierten Schemas. Sie können nach Anzeige der Liste der XML-Schemas Schemas zur Liste hinzufügen oder aus der Liste löschen.

Um ein Schema hinzuzufügen, klicken Sie auf die Schaltfläche **Hinzufügen,** suchen Sie nach der gewünschten Schemadatei und wählen Sie sie aus. Die ausgewählte Schemadatei wird zur Liste im Hauptfenster hinzugefügt. Wenn Sie auf die Schaltfläche **Änderungen übernehmen** klicken, werden die neu hinzugefügten Schemas im Datenbankschema registriert.

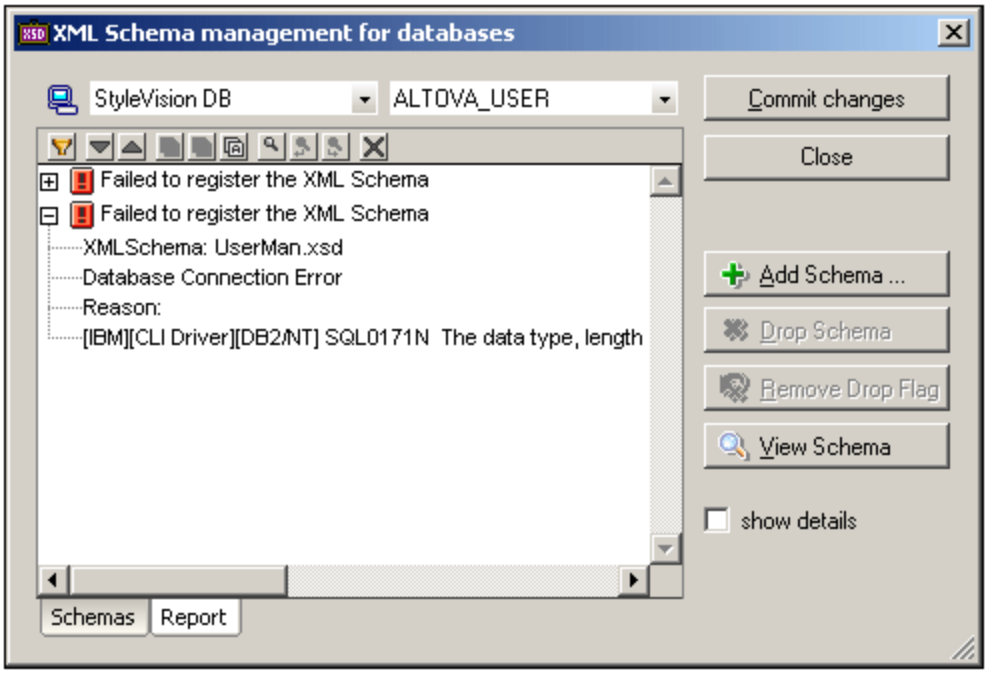
Um ein Schema zu löschen, wählen Sie das Schema im Hauptfenster aus und klicken Sie auf die Schaltfläche **Schema löschen.** Dem Schema wird ein Drop Flag zugewiesen, um anzuzeigen, dass es bei der Übernahme der nächsten Änderungen gelöscht wird. Sie können das Drop Flag löschen, indem Sie das so gekennzeichnete Schema auswählen und auf die Schaltfläche **Drop Flag entfernen** klicken. Wenn Sie auf die Schaltfläche **Änderungen übernehmen** klicken, wird die Registrierung aller für die Löschung markierten Schemas im Datenbankschema aufgehoben.

Wenn Sie auf die Schaltfläche "Schema anzeigen" klicken, wird das Schema in XMLSpy geöffnet. Um das Dialogfeld "XML-Schemaverwaltung" zu schließen, klicken Sie auf die Schaltfläche **Schließen.**

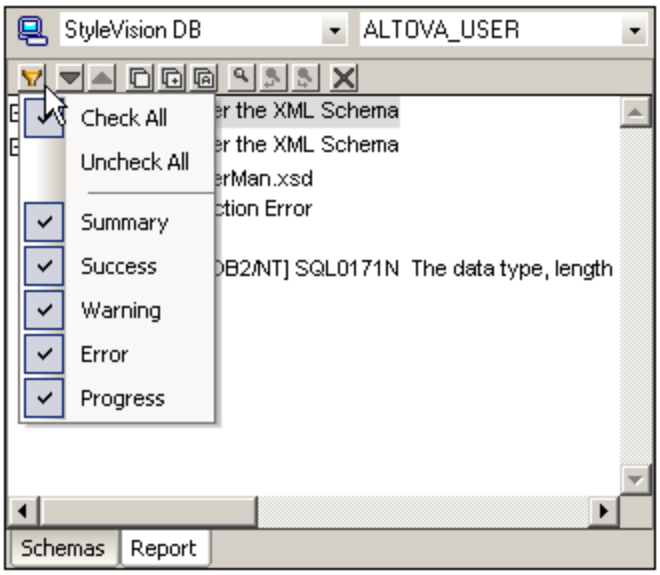
© 2016-2022 Altova GmbH

Berichte

Wenn Sie auf die Schaltfläche **Änderungen übernehmen** klicken, werden die von Ihnen vorgenommenen Änderungen in der Datenbank übernommen. Ein Bericht der Aktion "Übernehmen" wird im Berichtsfenster (*Abbildung unten*) angezeigt, sodass Sie sehen, ob die Aktion erfolgreich durchgeführt wurde, und um mögliche Fehler beheben zu können. Anschließend wird ein weiterer Bericht unterhalb des vorherigen angezeigt.



Das Berichtsfenster verfügt über Schaltflächen, mit denen Sie die Anzeige der Berichtsliste anpassen, in der Liste navigieren, Berichtsmeldungen kopieren, nach Text suchen und den Inhalt des Fensters löschen können (*siehe Abbildung unten*).



Mit Hilfe der Schaltfläche **Filter** können Sie bestimmte Arten von Meldungen ein oder ausblenden. Mit Hilfe der Schaltflächen **Nächste** und **Vorherige** wird die jeweils vorige bzw. nächste Meldung in der Liste markiert. Meldungen können auch mit oder ohne Ihre Sub-Einträge in die Zwischenablage kopiert werden, sodass sie in Dokumente eingefügt werden können. Mit Hilfe der Funktion **Suchen** können Sie einen Suchbegriff angeben und dann die Liste nach diesem Begriff durchsuchen. Mit der Schaltfläche **Löschen** können Sie den Inhalt des Berichtsfensters löschen.

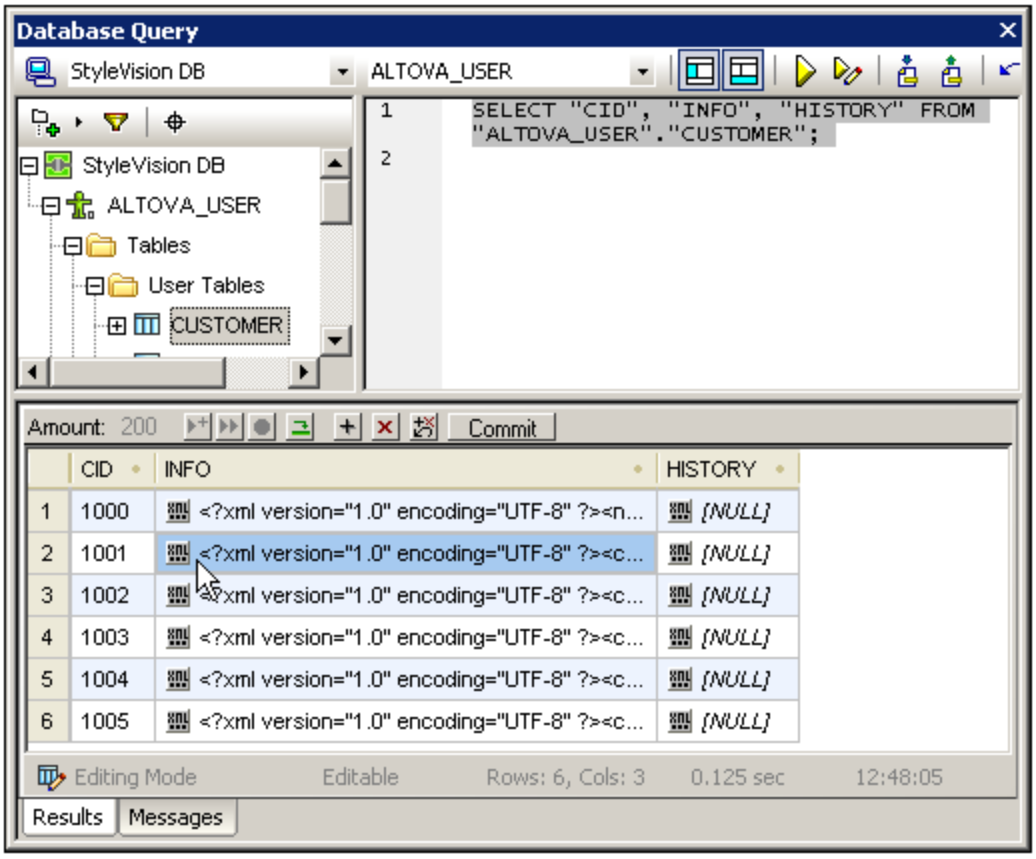
28.10.2.2 XML-Schema zuweisen

Mit dem Befehl "XML-Schema zuweisen" wird einer XML-Datei, die zur Bearbeitung über das Ergebnisraster des Fensters "Datenbankabfrage" geöffnet wurde, ein XML-Schema zugewiesen. Nach Zuweisung des Schemas kann die XML-Datei gegen das zugewiesene Schema validiert werden. Die Zuweisung wird in die Datenbank geschrieben, wenn die XML-Datei in XMLSpy gespeichert wurde.

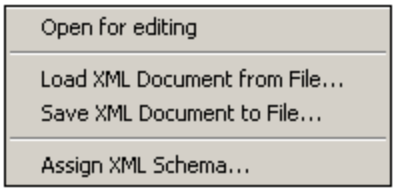
Öffnen einer Datenbank-XML-Datei zur Bearbeitung

Wenn im Fenster "Datenbankabfrage" eine Abfrage an einer XML-Datenbank durchgeführt werden soll und die Abfrage zur Bearbeitung von Daten ausgeführt wird, haben Sie im Ergebnisraster im unteren Bereich des Fensters "Datenbankabfrage" Zugriff auf die XML-Dateien in der Datenbank, sodass diese bearbeitet werden können (*siehe Abbildung unten*).

© 2016-2022 Altova GmbH



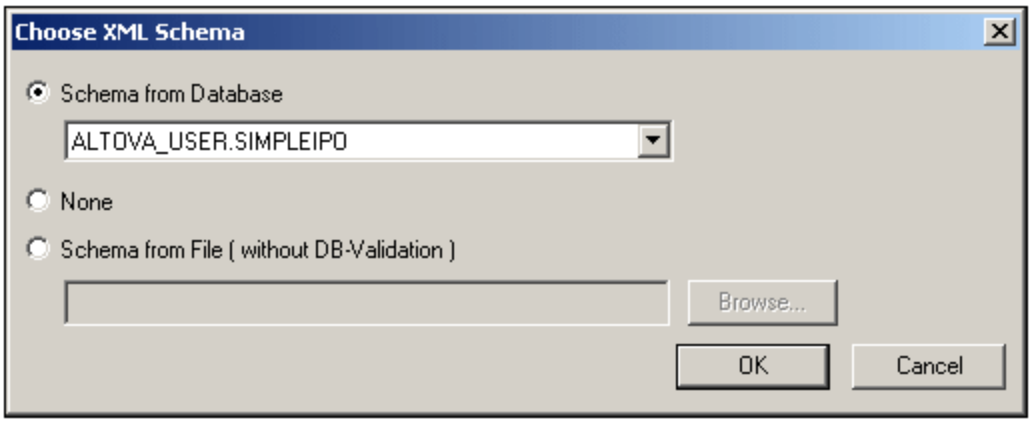
Wenn Sie auf die Schaltfläche "XML" klicken, erscheint das folgenden Menü.



Wenn Sie den Befehl **Zur Bearbeitung öffnen** auswählen, wird das XML-Dokument in XMLSpy geöffnet, wo es bearbeitet werden kann.

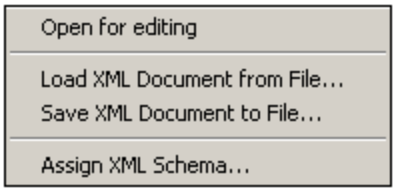
Zuweisen eines Schemas zur Datenbank-XML-Datei

Der Befehl **IBM DB2 | XML-Schema zuweisen** steht zur Verfügung, wenn die Datenbank-XML-Datei zur Bearbeitung in XMLSpy geöffnet wird. Wenn Sie den Befehl **XML-Schema zuweisen** aufrufen, während das XML-Dokument in XMLSpy aktiv ist, wird das Dialogfeld "XML-Schema auswählen" (*Abbildung unten*) geöffnet.



Sie können ein Schema aus den in der Datenbank gespeicherten Schemas auswählen (diese Schemas werden in der Dropdown-Liste der Auswahlliste "Datenbank" aufgelistet) oder aus externen Dateien, die durchsucht werden können. Wenn Sie auf **OK** klicken, wird das Schema der XML-Datei zugewiesen. Beachten Sie, dass die Zuweisung nicht in die XML-Datei geschrieben wird. Die Schemazuweisung wird in der Datenbank gespeichert, wenn das Dokument in XMLSpy gespeichert wird - und falls in der Datenbankabfrage-Symbolleiste beim Öffnen des Dokuments die Schaltfläche "XML-Änderungen automatisch übernehmen" aktiviert war. Beachten Sie, dass die Schemazuweisung in die Datenbank und nicht in die XML-Datei geschrieben wird.

Anmerkung: Das Menü "XML bearbeiten" im Ergebnisraster des Fensters "Datenbankabfrage" hat ebenfalls einen Befehl **XML-Schema zuweisen** (*siehe Abbildung unten*), der ebenfalls einer Datenbank-XML-Datei ein Schema zuweist.



Der Unterschied zwischen den beiden "XML-Schema zuweisen"-Befehlen besteht darin, dass Sie mit Hilfe des Befehls im Menü **DB | IBM DB2** ein XML-Schema zuweisen können, während Sie die XML-Datei bearbeiten, sodass Sie die Schemazuweisungen während der Editierung des XML-Dokuments ändern und das XML-Dokument sofort validieren können.

28.10.3 SQL Server

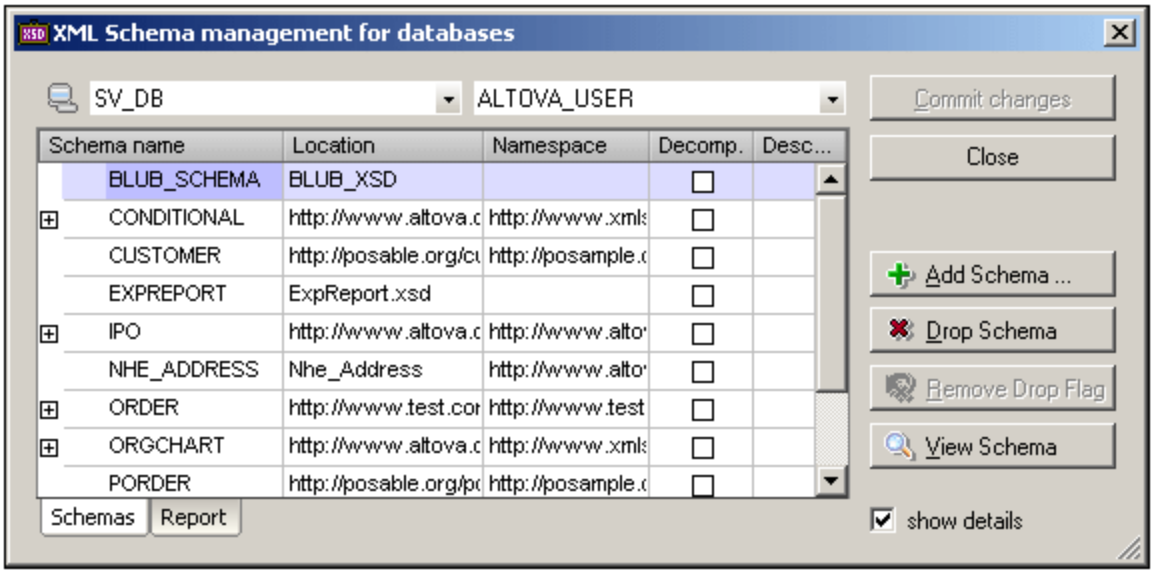
Der Befehl **SQL-Server** enthält ein Untermenü, das den Befehl "XML-Schemas verwalten" enthält.

28.10.3.1 XML-Schemas verwalten

Über das Dialogfeld "XML-Schemaverwaltung für Datenbanken" können Sie XML-Schemas aus dem Schema-Repository einer XML-Datenbank hinzufügen und entfernen. Nachdem Sie eine Verbindung zur Datenbank hergestellt haben, ruft XMLSpy das Dialogfeld "XML-Schemaverwaltung für Datenbanken" auf, in dem Sie die

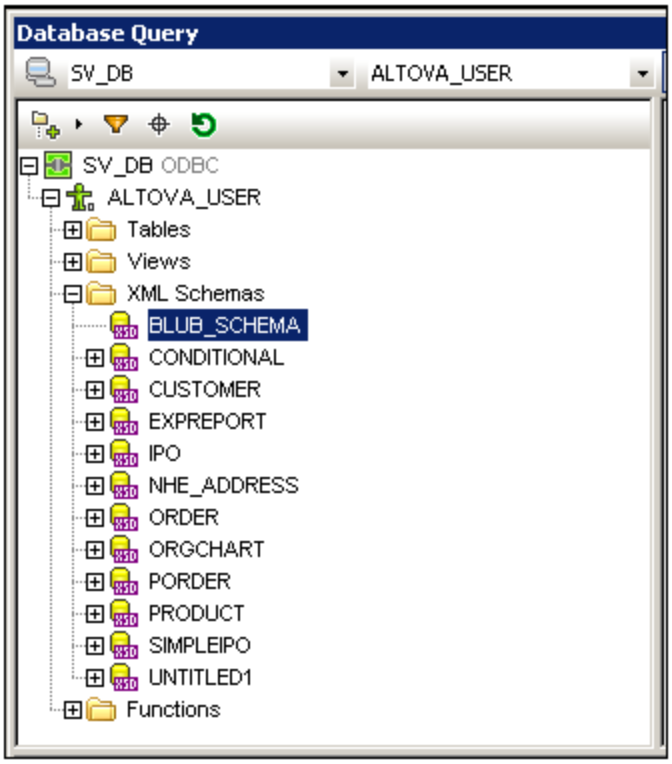
 © 2016-2022 Altova GmbH

XML-Schemas verwalten können.



Das Dialogfeld enthält eine Schnellverbindungsschaltfläche zum Aufrufen des Verbindungsassistenten[969], über den eine Verbindung zu einer Datenquelle hergestellt werden kann. Wenn mehrere Verbindungen vorhanden sind, kann die gewünschte Verbindung aus der Auswahlliste links oben ausgewählt werden. Anschließend können Sie aus der Auswahlliste auf der rechte Seite das benötigte Root-Objekt auswählen. Im Dialogfeld werden alle XML-Schemas, die sich für dieses Root-Objekt im Repository befinden angezeigt. Es werden der Name, Pfad und Namespace der einzelnen Schemas aufgelistet.

Beachten Sie: Die gespeicherten Schemas können auch im Fenster "Datenbankabfrage" (*Abbildung unten*) angezeigt werden, können dort aber nicht verwaltet werden. Verwenden Sie zum Verwalten von Schemas das Dialogfeld "XML-Schemaverwaltung für Datenbanken".

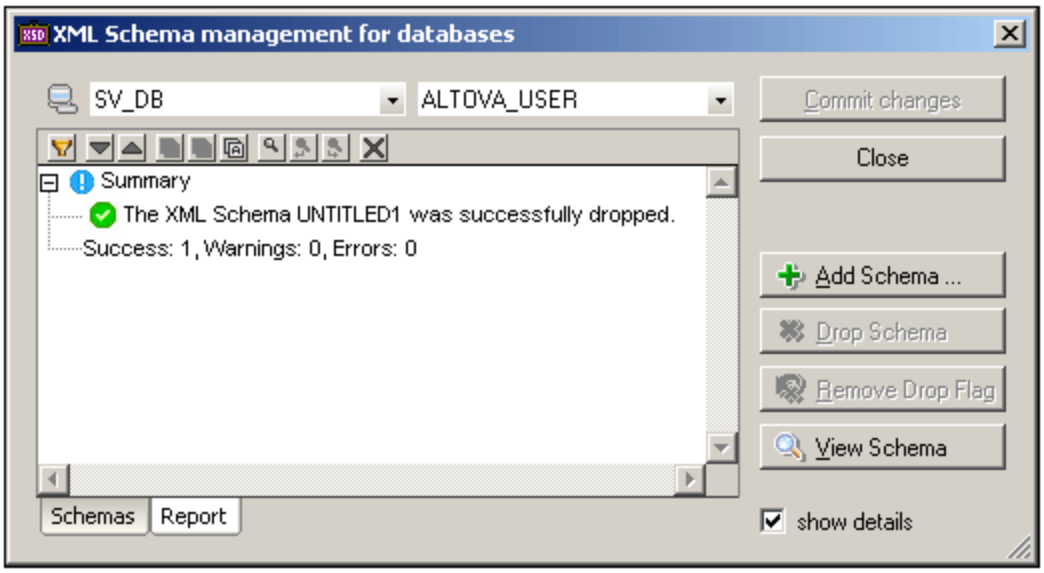


Gehen Sie im Dialogfeld "XML-Schemaverwaltung für Datenbanken" folgendermaßen vor:

- Fügen Sie mit Hilfe der Schaltfläche **Schema hinzufügen** ein Schema hinzu. Das ausgewählte Schema wird an die Liste angehängt und für das Hinzufügen markiert.
- Über die Schaltfläche "Schema löschen" können Sie Schemas in der Liste für die Löschung markieren. Die "Löschen"-Markierung kann über die Schaltfläche "Drop Flag entfernen" entfernt werden.
- Um ein ausgewähltes Schema in der Schema-Ansicht zu öffnen, klicken Sie auf die Schaltfläche **Schema anzeigen.**
- Über die Schaltfläche **Änderungen übernehmen** werden die Änderungen (hinzugefügte und gelöschte Schemas) übernommen.

Nachdem Sie die Änderungen übernommen haben, können Sie auf dem Register "Bericht" (*Abbildung unten*) einen Bericht der Übernehmen-Aktionen anzeigen.

© 2016-2022 Altova GmbH



28.10.4 Oracle XML DB

Mit XMLSpy können Sie eine Verbindung zu Oracle XML-Datenbanken herstellen, um diese abzufragen.

Folgende Datenbankfunktionen werden unterstützt:

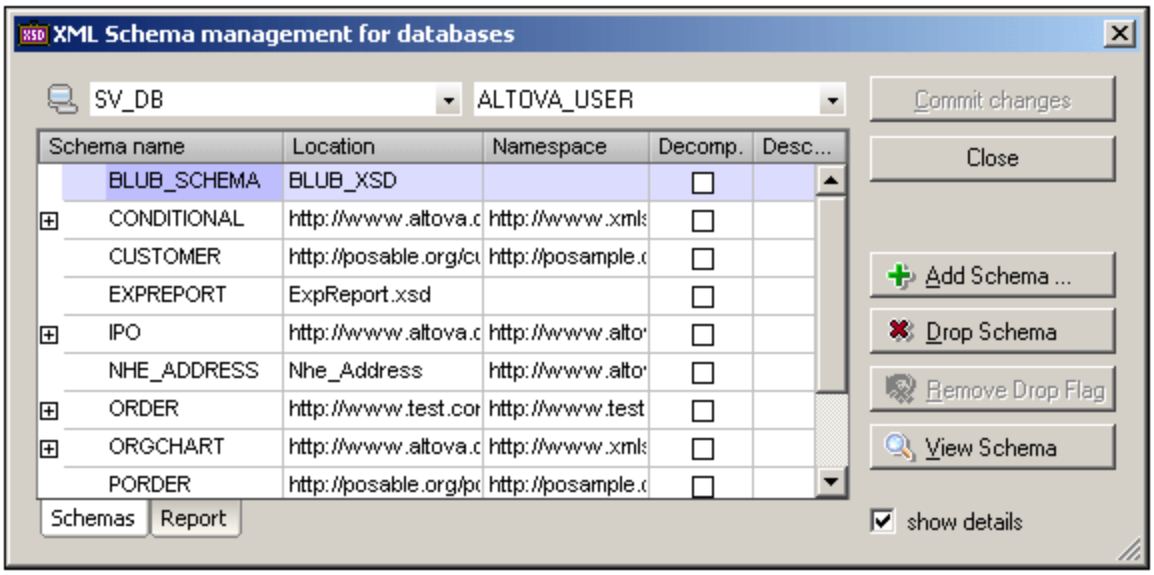
- Hinzufügen (und Registrieren) von XML-Schemas in einer Oracle XML-Datenbank. Zum Registrieren von XML-Schemas mit XMLSpy muss zuvor der Oracle XML DB Client installiert werden.
- Öffnen und Löschen von Schemas
- Abfrage der Datenbank durch Verwendung von XPath Statements (DBUri)
- Durchsuchen von XML-Dokumenten (mittels WebDAV)
- Erstellen von XML-Dokumenten basierend auf Schemas, die in der Datenbank gespeichert sind.

Installation:

- Downloaden und installieren Sie XMLSpy
- Installieren Sie Oracle Server (falls nötig)
- Erzeugen Sie eine Oracle Datenbank

28.10.4.1 XML-Schemas verwalten

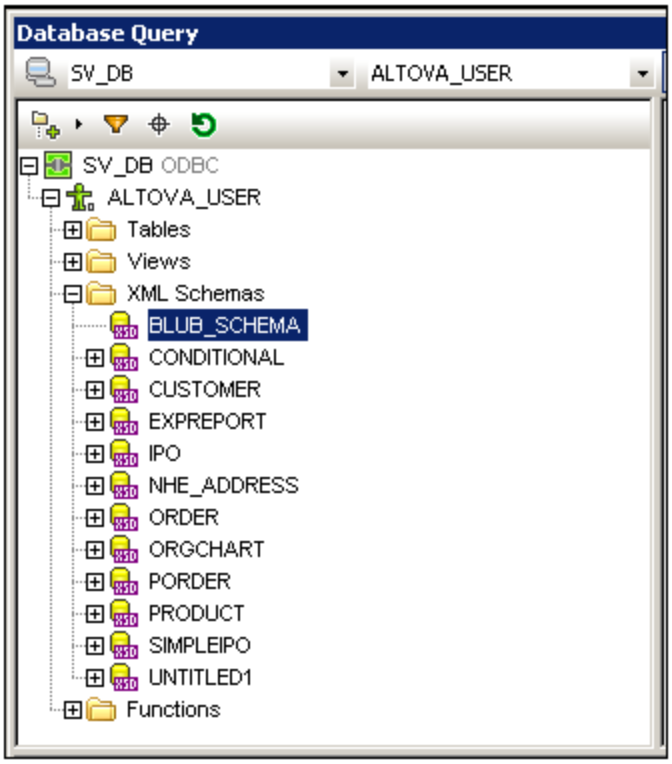
Über das Dialogfeld "XML-Schemaverwaltung für Datenbanken" können Sie XML-Schemas aus dem Schema-Repository einer XML-Datenbank hinzufügen und entfernen. Nachdem Sie eine Verbindung zur Datenbank hergestellt haben, ruft XMLSpy das Dialogfeld "XML-Schemaverwaltung für Datenbanken" auf, in dem Sie die XML-Schemas verwalten können.



Das Dialogfeld enthält eine Schnellverbindungsschaltfläche zum Aufrufen des Verbindungsassistenten, über den eine Verbindung zu einer Datenquelle hergestellt werden kann. Wenn mehrere Verbindungen vorhanden sind, kann die gewünschte Verbindung aus der Auswahlliste links oben ausgewählt werden. Anschließend können Sie aus der Auswahlliste auf der rechte Seite das benötigte Root-Objekt auswählen. Im Dialogfeld werden alle XML-Schemas, die sich für dieses Root-Objekt im Repository befinden angezeigt. Es werden der Name, Pfad und Namespace der einzelnen Schemas aufgelistet.

Beachten Sie: Die gespeicherten Schemas können auch im Fenster "Datenbankabfrage" (*Abbildung unten*) angezeigt werden, können dort aber nicht verwaltet werden. Verwenden Sie zum Verwalten von Schemas das Dialogfeld "XML-Schemaverwaltung für Datenbanken".

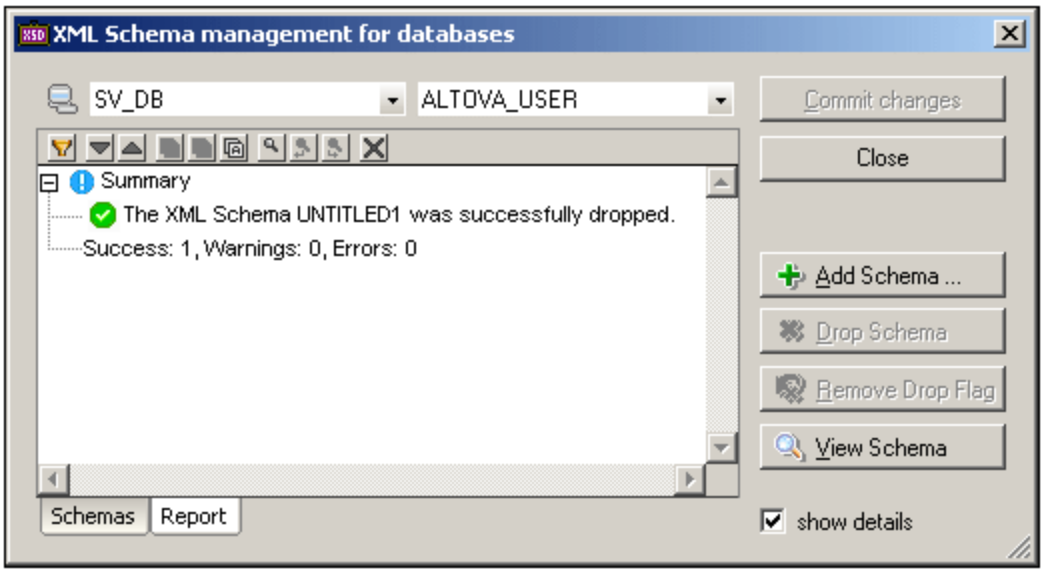
© 2016-2022 Altova GmbH



Gehen Sie im Dialogfeld "XML-Schemaverwaltung für Datenbanken" folgendermaßen vor:

- Fügen Sie mit Hilfe der Schaltfläche **Schema hinzufügen** ein Schema hinzu. Das ausgewählte Schema wird an die Liste angehängt und für das Hinzufügen markiert.
- Über die Schaltfläche "Schema löschen" können Sie Schemas in der Liste für die Löschung markieren. Die "Löschen"-Markierung kann über die Schaltfläche "Drop Flag entfernen" entfernt werden.
- Um ein ausgewähltes Schema in der Schema-Ansicht zu öffnen, klicken Sie auf die Schaltfläche **Schema anzeigen.**
- Über die Schaltfläche **Änderungen übernehmen** werden die Änderungen (hinzugefügte und gelöschte Schemas) übernommen.

Nachdem Sie die Änderungen übernommen haben, können Sie auf dem Register "Bericht" (*Abbildung unten*) einen Bericht der Übernehmen-Aktionen anzeigen.

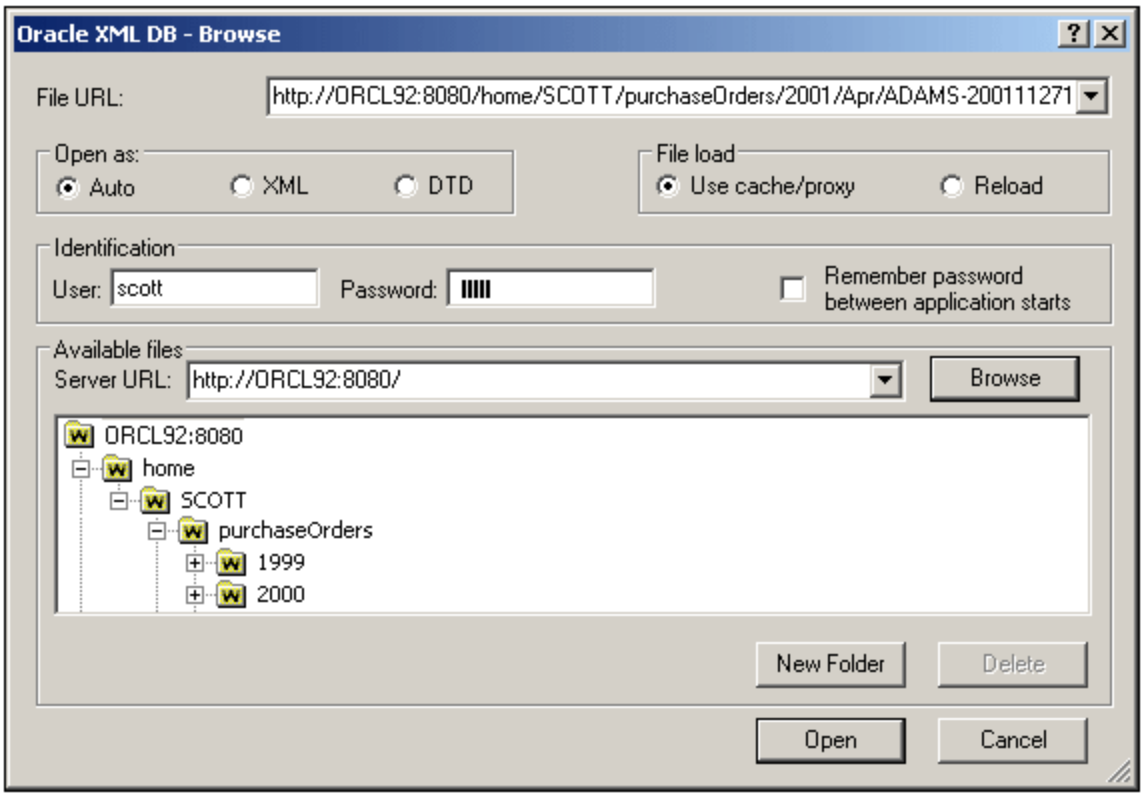


28.10.4.2 Oracle XML-Dokumente durchsuchen



Mit diesem Befehl können Sie die XML-Dokumente durchsuchen, die auf Ihrem Server zur Verfügung stehen. Serverinformationen, die von früheren Datenbank-Abfragen zur Verfügung stehen, werden automatisch eingegeben. Sollte dies nicht der Fall sein, geben Sie die Informationen händisch ein.

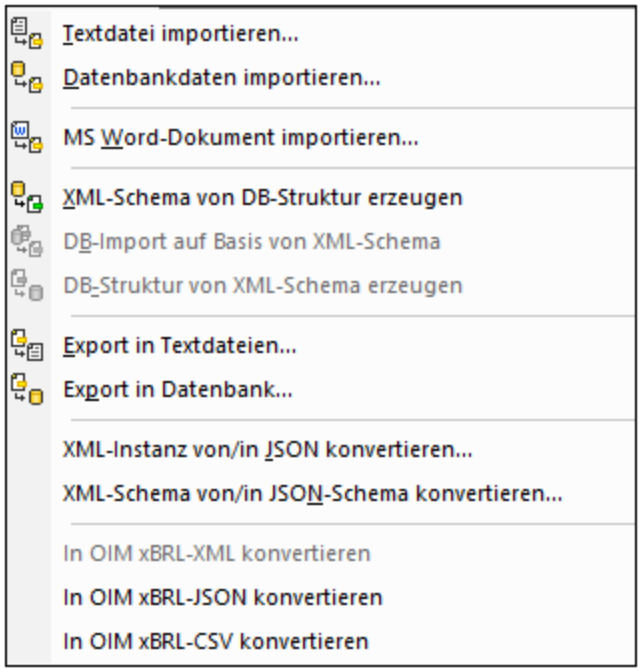
© 2016-2022 Altova GmbH



Im Baumdiagramm finden Sie die spezifischen XML Dateien. Doppelklicken Sie auf das Baumdiagramm um es weiter auszuklappen. Sie können auch auf eine Datei klicken und dann auf **Öffnen** klicken, um dasselbe Ergebnis zu erzielen. Mit **Neuer Ordner** können Sie neue Ordner einfügen, mit **Löschen** entfernen Sie zuvor ausgewählte XML Dateien.

28.11 Menü "Konvertieren"

Im Menü **Konvertieren** (*Abbildung unten*) stehen Ihnen leistungsfähige Funktionen für den Austausch von Daten zur Verfügung:



- Importieren und Exportieren von Text-, Word-Prozessor-, Datenbank- und XML-Dateien.
- Importieren von Datenbankdaten[1483] auf Basis eines vorhandenen XML-Schemas.
- Erzeugen von XML-Schemas[1478], die auf einer bereits existierenden Datenbank basieren.
- Erzeugen einer **Datenbankstruktur**[1484], die auf einem existierenden XML-Schema basiert.
- Konvertieren zwischen XML- und JSON-Instanzen[1493] sowie zwischen XML- und JSON-Schemas[1497].
- Konvertierung von XBRL-Daten in die OIM-Darstellung von xBRL-JSON und xBRL-CSV und Konvertierung einer beliebigen OIM-Darstellung (xBRL-XML, xBRL-JSON und xBRL-CSV) in eine andere.

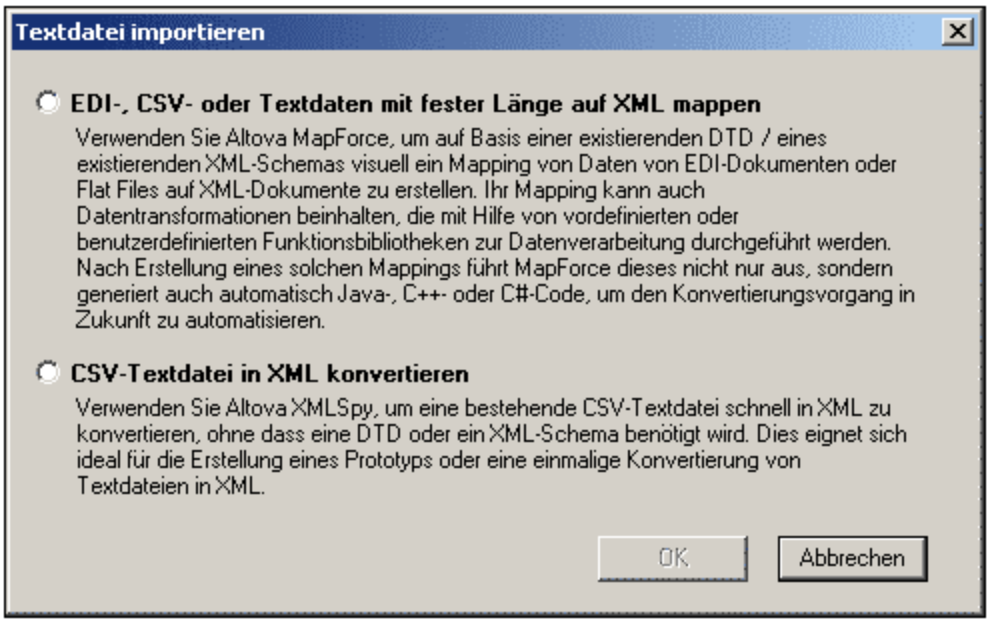
28.11.1 Textdatei importieren



Mit diesem Befehl lässt sich jede Datei mit **strukturierten Text** in XMLSpy **importieren** und sofort ins XML-Format konvertieren. Diese Funktion ist nützlich, um Altdatenbestände aus älteren Systemen zu importieren. Die Schritte zum Importieren von Daten in einer Textdatei als XML-Dokument sind unten beschrieben.

© 2016-2022 Altova GmbH

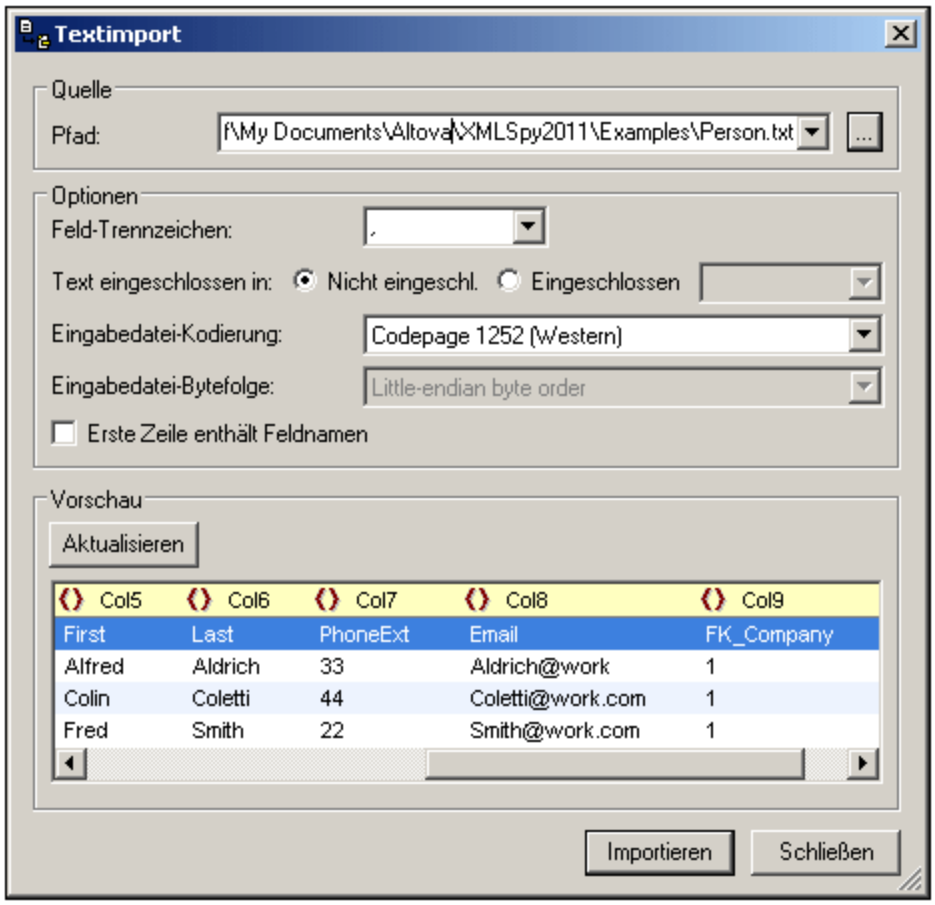
1. Wählen Sie den Menübefehl Konvertieren | Textdatei importieren.
2. Wählen Sie im daraufhin angezeigten Dialogfeld (*Abbildung unten*) eine der folgenden beiden Optionen aus. (Damit die Mapping-Option verwendet werden kann, muss Altova MapForce installiert sein).



3. Klicken Sie auf OK. Daraufhin wird das Dialogfeld "Textimport" angezeigt.
4. Wählen Sie die gewünschten Textimportoptionen aus (siehe Beschreibung im nächsten Abschnitt) und klicken Sie auf "Importieren". Die importierten Daten werden in ein XML-Dokument konvertiert, das in der Grid-Ansicht angezeigt wird.

Textimportoptionen

Die Textimportoptionen werden im Dialogfeld "Textimport" (*Abbildung unten*) definiert und sind weiter unten beschrieben. Wie Sie das Dialogfeld aufrufen, ist im vorherigen Abschnitt beschrieben.



Pfad
Geben Sie den Pfad zu der zu importierenden Datei in das Textfeld "Pfad" ein oder wählen Sie die Datei mit Hilfe der Durchsuchen-Schaltfläche rechts vom Textfeld aus. Nachdem Sie die Datei ausgewählt haben, wird eine Grid-Vorschau der XML-Datei im Fenster "Vorschau" angezeigt. Alle Änderungen an den in diesem Dialogfeld ausgewählten Optionen, werden sofort in der Vorschau angezeigt.

Feld-Trennzeichen
Um eine Textdatei erfolgreich importieren zu können, müssen Sie das Feld-Trennzeichen angeben, das zum Trennen von Spalten oder Feldern in der Datei verwendet wird. XMLSpy erkennt gebräuchliche Zeilen-Trennzeichen automatisch (CR, LF oder CR+LF).

Text eingeschlossen in
In Textdateien, die aus alten Datenbeständen exportiert wurden, sind Textwerte manchmal zur besseren Unterscheidung von numerischen Werten in Anführungszeichen eingeschlossen. In diesem Fall können Sie angeben, welche Art von Anführungszeichen in Ihrer Datei verwendet werden, damit diese Anführungszeichen beim Import der Daten automatisch entfernt werden.

Kodierung
Die Daten werden ins Unicode[2271]-Format (auf dem alle XML-Dokumente basieren) konvertiert, daher müssen Sie angeben, in welchem Zeichensatz die Datei derzeit kodiert ist. Bei amerikanischen oder westeuropäischen Windows-Systemen ist dies meist Codepage 1252, auch als ANSI Encoding bezeichnet.

© 2016-2022 Altova GmbH

Bytefolge
Bei Import von 16-Bit oder 32-Bit Unicode (UCS-2, UTF-16 oder UCS-4) Dateien, können Sie auch zwischen der Bytefolge Little Endian und Big Endian wählen.

Erste Zeile enthält Feldnamen
Es kommt außerdem bei Textdateien häufig vor, dass diese in der ersten Zeile der Datei die Feldnamen enthalten. Aktivieren Sie dieses Kontrollkästchen, wenn das der Fall ist.

Vorschau
Im Bereich "Vorschau" können Sie die Spaltenüberschriften umbenennen, indem Sie auf den Namen klicken und ihn bearbeiten. Die Spaltenüberschriften bilden im XML-Dokument die Element- oder Attributnamen. Sie können auch auswählen, ob eine Spalte im XML-Dokument ein Element oder ein Attribut werden soll oder ob sie nicht in das XML-Dokument importiert werden soll. Klicken Sie in der jeweiligen Spalte auf das Symbol für den Spaltentyp, um diese Optionen zu aktivieren bzw. zu deaktivieren. Im obigen Screenshot wird die Spalte *Longitude in Seconds* (*LonS*) nicht importiert.

28.11.2 Datenbankdaten importieren

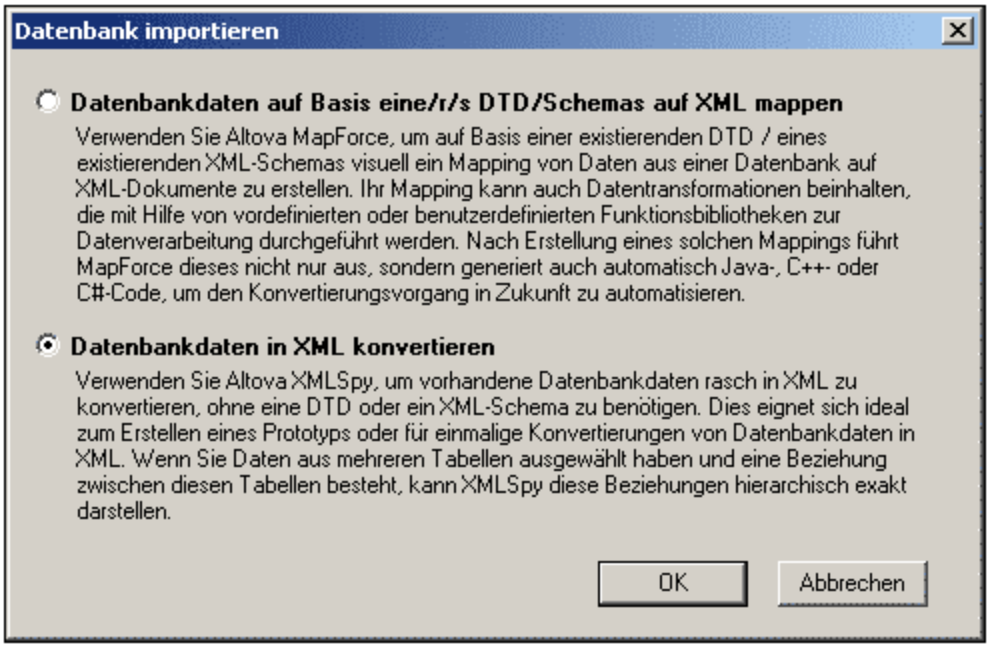


Mit Hilfe des Befehls **Datenbankdaten importieren** können Sie Daten aus verschiedenen Datenbanktypen in eine XML-Datei importieren. Der Import erfolgt in zwei Schritten:

1. Aufbau einer Verbindung zur Datenbank. Eine Anleitung dazu finden Sie unter Herstellen einer Verbindung zu einer Datenbank 969.
2. Auswahl der zu importierenden Daten 1473

Um Datenbankdaten zu importieren, gehen Sie folgendermaßen vor:

1. Wenn Sie auf den Befehl "Datenbankdaten importieren" klicken, wird das Dialogfeld **Datenbank importieren** (*Abbildung unten*) geöffnet.



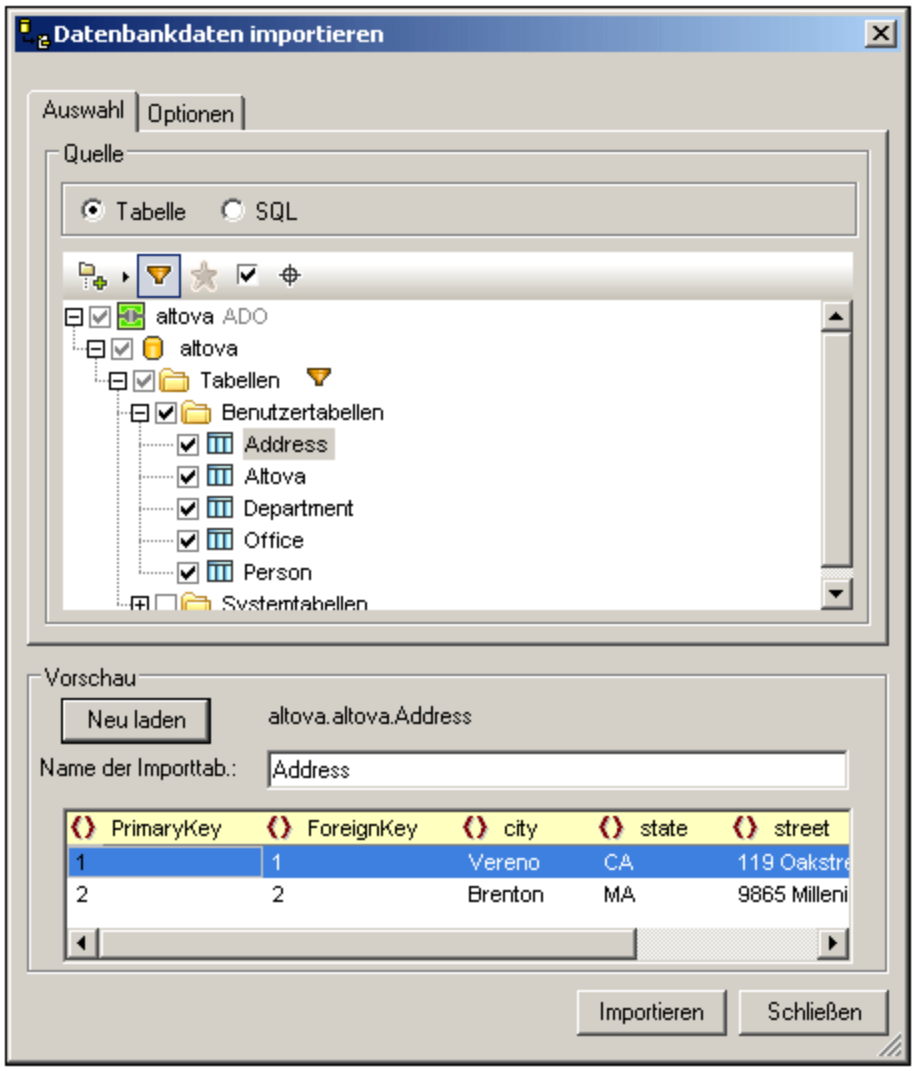
2. Wählen Sie die Option "Datenbankdaten in XML konvertieren" aus und klicken Sie auf **OK**. Daraufhin wird das Dialogfeld "Connect to a Data Source" geöffnet. (Wenn Sie die Option "Datenbankdaten auf Basis eine/r/s DTD/Schemas auf XML mappen" auswählen, muss Altova MapForce installiert sein, damit das Mapping durchgeführt werden kann.)
3. Stellen Sie im Dialogfeld "Connect to Data Source" eine Verbindung zur Datenbank her. Eine Anleitung dazu finden Sie unter Herstellen einer Verbindung zu einer Datenbank 969.
4. Nachdem Sie die Verbindung hergestellt haben, werden im Dialogfeld "Datenbankdaten importieren" Register und Fenster angezeigt, über die Sie die gewünschten Datenbankdaten auswählen können. Diese Optionen sind weiter unten beschrieben. Nachdem Sie damit fertig sind, klicken Sie auf die Schaltfläche **Import,** um die Datenbankdaten zu importieren.

Datenauswahl und Importoptionen

Das Dialogfeld "Datenbankdaten importieren" zum Auswählen der Daten und Einstellen der Importoptionen besteht aus zwei Teilen (*die in den Abbildungen unten separat gezeigt werden*):

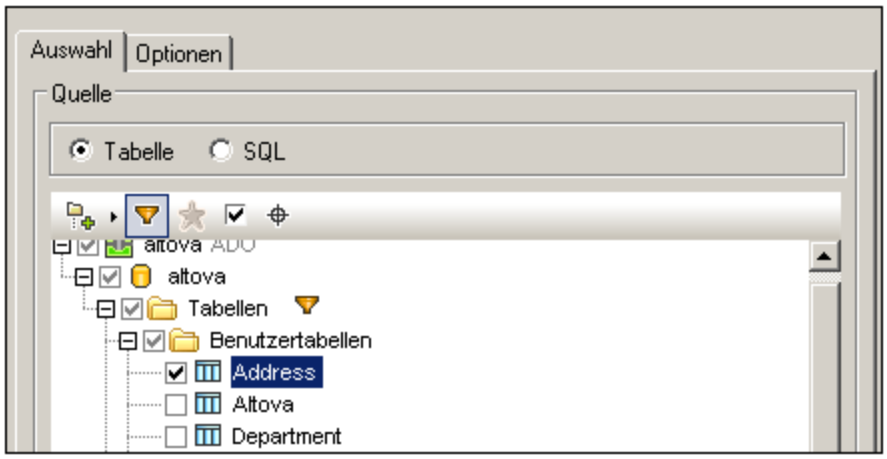
- einem oberen Bereich mit zwei Registern: (i) Auswahl und (ii) Optionen.
- einem unteren Bereich, dem Vorschaufenster, in dem die Daten entsprechend der getroffenen Auswahl und der gewählten Optionen angezeigt werden.

© 2016-2022 Altova GmbH



Datenauswahlmethode

Auf dem Register *Auswahl* (*Abbildung oben*) werden im Bereich *Quelle* (*Abbildung unten*) entweder die Tabellen der Datenbank angezeigt oder Sie sehen eine editierbare SQL-Anweisung zur Auswahl der gewünschten Tabellen, wobei die einzelnen Ansichten durch Aktivieren des entsprechenden Optionsfelds ausgewählt werden.



Tabellenauswahloptionen

In der Tabellenansicht können Sie die Tabellen in der Datenbank auswählen, die Sie importieren möchten. Aktivieren Sie dazu das Kästchen neben der jeweiligen Tabelle (*Abbildung oben*). Der Inhalt der Tabelle kann dann im Vorschaufenster durch Klicken auf die Schaltfläche **Vorschau** angezeigt werden.

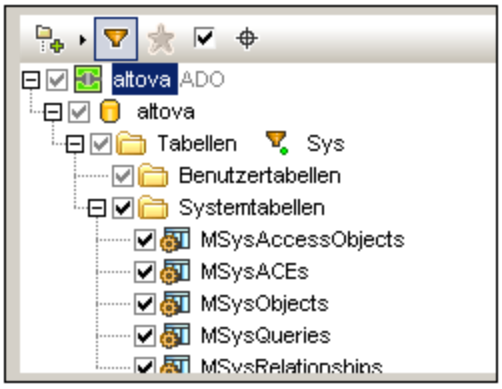
In der Tabellenauswahlansicht stehen Auswahlschaltflächen in der Symbolleiste zur Verfügung (*Abbildung unten*).



Diese Schaltflächen sind von links nach rechts:

- *Ordner-Layout*, in dem Sie Datenbankobjekte in (i) Ordnern nach Objekttyp; (ii) Ordnern nach Objekttyp oder ohne Schemaordner; (iii) in einer Hierarchie aber ohne Ordner; und (iv) in Tabellenkategorien basierend auf Beziehungen zu anderen Tabellen gliedern können.
- Wenn Sie auf die Schaltfläche *Ordnerinhalt filtern* klicken, wird ein Filter auf den ausgewählten Ordner angewendet, um die Objekte im Ordner zu filtern. In der Abbildung unten wurde z.B. ein Filter angewendet, um Objekte anzuzeigen, die in ihrem Namen den Text `Sys` enthalten. Wenn Sie auf das Filtersymbol klicken, erscheint ein Menü mit einer Liste von Filteroptionen.

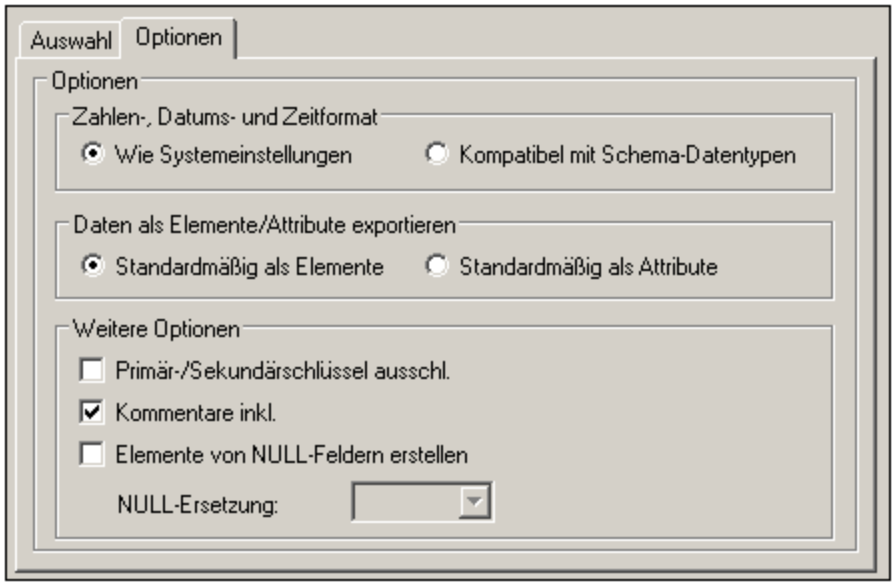
© 2016-2022 Altova GmbH



- *Favoriten anzeigen:* Filtert die Objekte nach Favoriten.
- *Nur ausgewählte Objekte anzeigen*: Filtert die Objekte nach aktivierten Objekten.
- *Objektsuche:* Ruft ein Textfeld auf, das sich wie ein Suchfeld verhält. Sie können einen Textstring eingeben und in der Dropdown-Liste werden alle Objekte angezeigt, deren Namen den Textstring enthalten. Bei Auswahl eines dieser Objekte aus der Dropdown-Liste wird das Objekt in der Struktur markiert.

Register "Optionen"

Auf dem Register "Optionen" (*Abbildung unten*) können Sie definieren, wie Zahlen, Datums- und Uhrzeitwerte importiert werden sollen, ob Daten als Elemente oder Attribute importiert werden sollen und ob Kommentare und `NULL` Felder mitimportiert werden sollen.

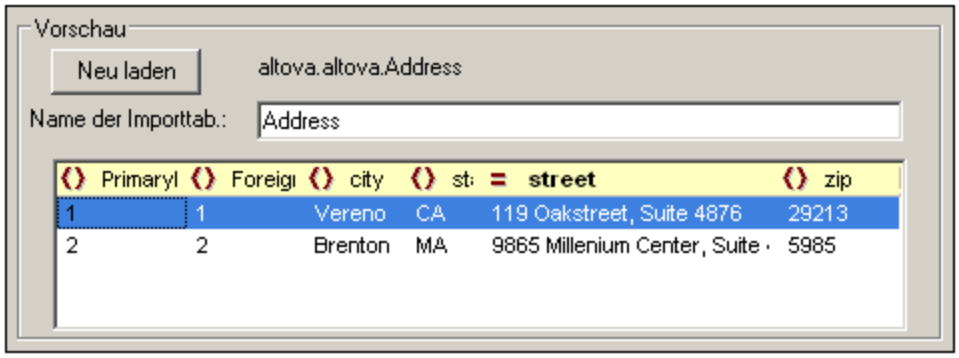


Wenn `NULL`-Felder für den Import aktiviert sind, können Sie dafür einen XML-Ersetzungswert eingeben.

Vorschaubereich

Im Vorschaubereich (*Abbildung unten*) wird die Struktur der auf dem Register "Auswahl" ausgewählten Tabelle angezeigt. Wenn auf dem Register "Auswahl" eine neue Tabelle ausgewählt wird, klicken Sie im Bereich

"Vorschau" auf die Schaltfläche "Vorschau", um die Tabelle anzuzeigen. Klicken Sie auf die Schaltfläche "**Neu laden**", um die Vorschau zu aktualisieren.



Sie können festlegen, ob ein Feld als Element oder Attribut oder gar nicht importiert werden soll. Klicken Sie dazu auf das Symbol links vom Spaltennamen. Klicken Sie darauf, um der Reihe nach die Optionen "Element", "Attribut" und "Ignorieren" zu aktivieren. In der Abbildung oben wurde das Feld `city` für den Import als Element und das Feld `street` als Attribut definiert.

Datentyp-Konvertierungen

Informationen zu Konvertierung von Datenbank-Datentypen in XML-Schema-Datentypen finden Sie in den Anhängen[2136].

28.11.3 MS Word Dokument Importieren



Mit diesem Befehl können Sie ein Word-Dokument direkt importieren und ins XML-Format konvertieren. Bei Auswahl dieses Befehls wird das Dialogfeld "Öffnen" angezeigt, in dem Sie das gewünschte Word-Dokument auswählen können. XMLSpy erzeugt automatisch ein XML-Dokument, das ein CSS Stylesheet enthält. Anhand eines jeden Absatzes in Word wird ein XML-Element generiert, dessen Name anhand des entsprechenden Absatzstils im Microsoft Word-Dokument erstellt wird.

28.11.4 XML-Schema von DB-Struktur erzeugen



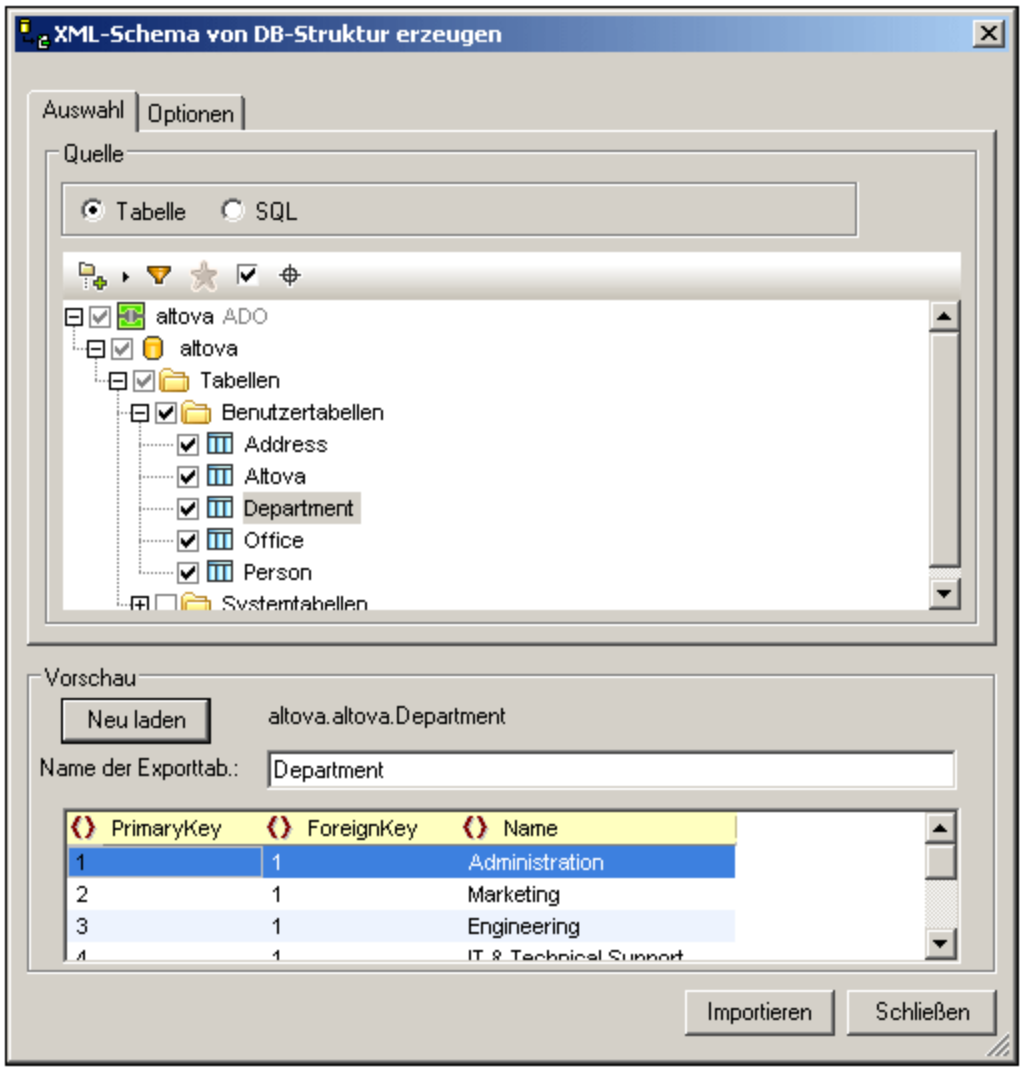
Mit Hilfe des Befehls **XML-Schema von DB-Struktur erzeugen** können Sie anhand der Struktur einer Datenbank ein XML-Schema erzeugen. Es werden die verschiedensten Datenbanktypen unterstützt. Die Erstellung des XML-Schemas erfolgt in zwei Schritten:

© 2016-2022 Altova GmbH

1. Aufbau einer Verbindung zur Datenbank. Eine Anleitung dazu finden Sie unter Herstellen einer Verbindung zu einer Datenbank 969.
2. Auswahl der Optionen für die Datenbankdaten und Angabe des XML-Schemas. Diese sind weiter unten beschrieben.

Auswahl der zu importierenden Datenstruktur

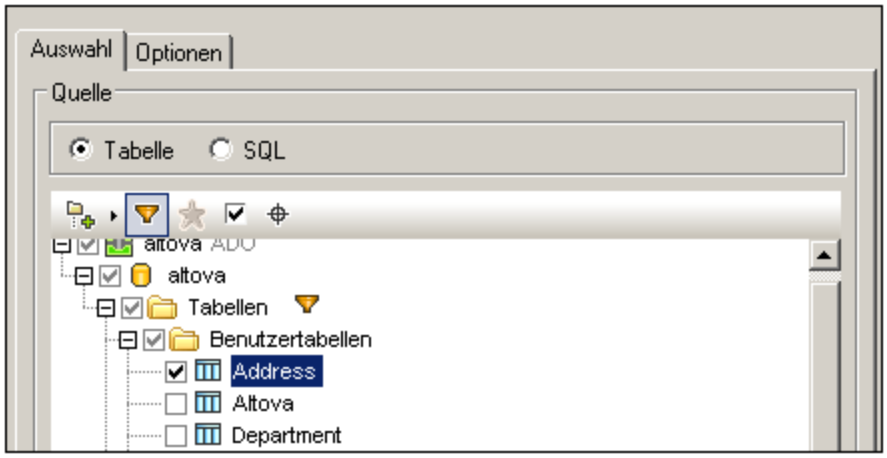
Nachdem Sie eine Verbindung zur Datenbank hergestellt haben, wird das Dialogfeld "XML-Schema von DB-Struktur erzeugen" angezeigt (*Abbildung unten*). Hier können Sie die zu importierende Datenbankstruktur auswählen. Klicken Sie anschließend auf Importieren, um ein XML-Schema zu erzeugen, in dem die Struktur eines XML-Dokuments definiert ist.



Datenauswahlmethode

Auf dem Register *Auswahl* (*Abbildung oben*) werden im Bereich *Quelle* (*Abbildung unten*) entweder die Tabellen der Datenbank angezeigt oder Sie sehen eine editierbare SQL-Anweisung zur Auswahl der gewünschten Tabellen, wobei die einzelnen Ansichten durch Aktivieren des entsprechenden Optionsfelds

ausgewählt werden.



Tabellenauswahloptionen

In der Tabellenansicht können Sie die Tabellen in der Datenbank auswählen, die Sie importieren möchten. Aktivieren Sie dazu das Kästchen neben der jeweiligen Tabelle (*Abbildung oben*). Der Inhalt der Tabelle kann dann im Vorschaufenster durch Klicken auf die Schaltfläche **Vorschau** angezeigt werden.

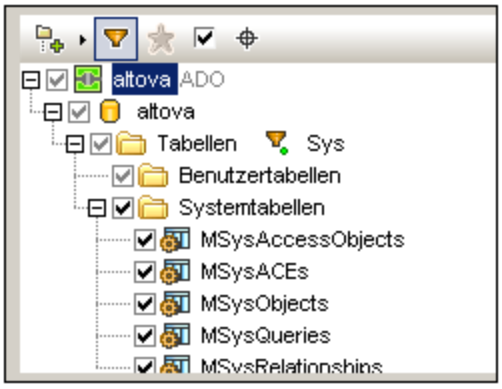
In der Tabellenauswahlansicht stehen Auswahlschaltflächen in der Symbolleiste zur Verfügung (*Abbildung unten*).



Diese Schaltflächen sind von links nach rechts:

- *Ordner-Layout*, in dem Sie Datenbankobjekte in (i) Ordnern nach Objekttyp; (ii) Ordnern nach Objekttyp oder ohne Schemaordner; (iii) in einer Hierarchie aber ohne Ordner; und (iv) in Tabellenkategorien basierend auf Beziehungen zu anderen Tabellen gliedern können.
- Wenn Sie auf die Schaltfläche *Ordnerinhalt filtern* klicken, wird ein Filter auf den ausgewählten Ordner angewendet, um die Objekte im Ordner zu filtern. In der Abbildung unten wurde z.B. ein Filter angewendet, um Objekte anzuzeigen, die in ihrem Namen den Text `Sys` enthalten. Wenn Sie auf das Filtersymbol klicken, erscheint ein Menü mit einer Liste von Filteroptionen.

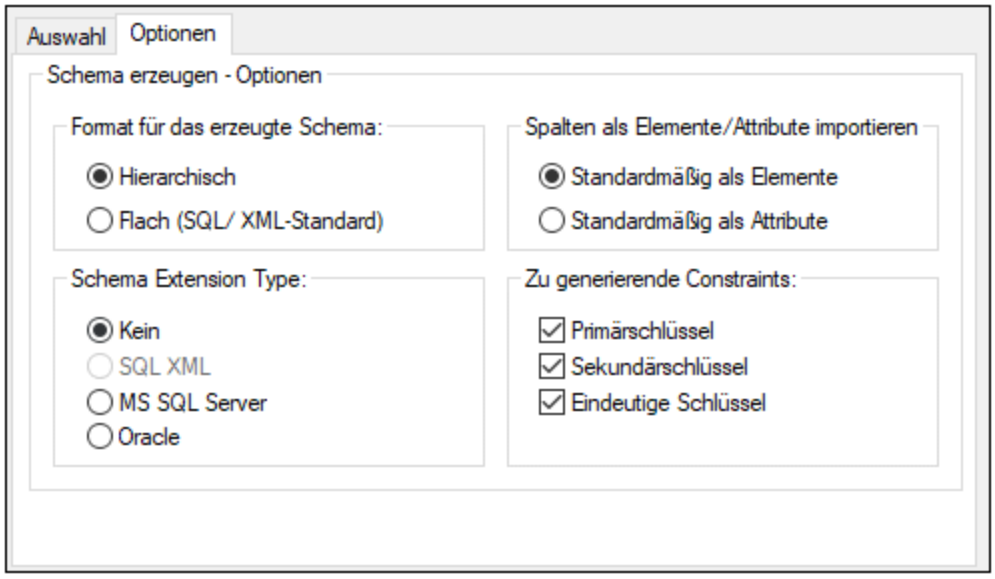
© 2016-2022 Altova GmbH



- *Favoriten anzeigen:* Filtert die Objekte nach Favoriten.
- *Nur ausgewählte Objekte anzeigen*: Filtert die Objekte nach aktivierten Objekten.
- *Objektsuche:* Ruft ein Textfeld auf, das sich wie ein Suchfeld verhält. Sie können einen Textstring eingeben und in der Dropdown-Liste werden alle Objekte angezeigt, deren Namen den Textstring enthalten. Bei Auswahl eines dieser Objekte aus der Dropdown-Liste wird das Objekt in der Struktur markiert.

Register "Optionen"

Auf dem Register *Optionen* (*Abbildung unten*) können Sie das Schemaformat, seinen Extension-Typ und die im Schema zu generierenden Datenbank-Constraints definieren und festlegen, ob Spalten als Elemente oder Attribute importiert werden sollen.



Schema-Format

Sie können zwischen einem flachen Format (SQL/XML-Standard) und einem hierarchischen Schemaformat auswählen.

- Das *flache Schema-Modell* basiert auf der ISO-ANSI SQL/XML Spezifikation INCITS/ISO/IEC 9075-14-2008. Darin wird definiert, wie Datenbanken auf XML gemappt werden. Beziehungen werden im Schema mittels Identity Constraints definiert und es gibt keine Referenzen auf Elemente. Daher hat

das Schema eine flache Struktur, die einer Baumstrukturansicht der Datenbank ähnelt. Die Spezifikation kann im ANSI Store erworben werden. Nähere Informationen dazu finden Sie unter www.iso.org.

- Im *hierarchischen Schema-Modell* werden die Tabellenabhängigkeiten visuell in Form einer Baumstrukturansicht dargestellt, in der abhängige Tabellen im Content Model als eingerückte Child-Elemente angezeigt werden. Die Tabellenabhängigkeiten werden auch auf dem Register "Identity Constraints" angezeigt.

Die Tabellen werden im Schema als globale Elemente aufgelistet und die Spalten sind die Elemente oder Attribute dieser globalen Elemente. (Der Benutzer entscheidet, ob die Spalten als Elemente oder Attribute gemappt werden sollen). Beziehungen werden auf hierarchische Art erstellt, sodass ein Sekundärschlüssel in einer Tabelle eine Referenz zu dem globalen Element ist, das für die Tabelle steht.

Schema-Extension-Typ
Schema-Extension-Informationen sind zusätzliche Informationen, die aus einer Datenbank ausgelesen werden und die anschließend in Form von Annotationsdaten oder Attributen in das Schema eingebettet werden. Beim Generieren von Schemas gibt es vier Extension-Typ-Optionen: (i) Keine Extension-Informationen; (ii) SQL/XML Extensions; (iii) MS SQL Server Extensions; und (iv) Oracle Extensions. Die folgenden Optionen stehen zur Verfügung:

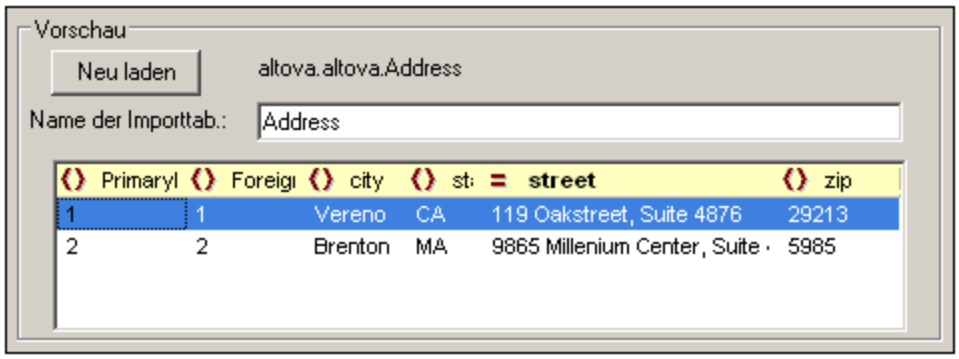
- *Keine:* Aus der Datenbank werden keine zusätzlichen Informationen bereitgestellt.
- *SQL XML:* SQL/XML Extensions werden nur eingefügt, wenn Schemas im flachen Format generiert werden. Die Extension-Informationen werden in Form von Annotationen gespeichert und in der SQL/XML Spezifikation (INCITS/ISO/IEC 9075-14-2008) beschrieben.
- *MS SQL Server:* Bei Auswahl von Microsoft SQL Server werden SQL Server Extensions generiert. Nähere Informationen dazu finden Sie unter SQL Server Books Online und den MSDN-Informationen zu mit Annotationen versehenen XSD-Schemas. Im Schema wird die folgende Gruppe von annotationsbezogenen Elementen generiert: `sql:relation`, `sql:field`, `sql:datatype`, `sql:mapped`.
- *Oracle:* Beim Arbeiten mit Oracle-Datenbanken werden standardmäßig Oracle-Extensions ausgewählt. Zusätzliche Datenbankinformationen werden als Attribute gespeichert. Ausführliche Informationen dazu finden Sie in der Online-Dokumentation von Oracle. Die folgende Attribut-Untergruppe wird derzeit generiert: `SQLName`, `SQLType`, `SQLSchema`.

Anmerkung: SQL Server und Oracle Extensions können zwar für die jeweilige Datenbank generiert werden, unterliegen aber keinen Einschränkungen. Dies erweist sich v.a. beim Arbeiten mit einer dritten Datenbank als nützlich, wenn Sie ein Schema generieren möchten, das später mit SQL Server oder Oracle verwendet werden soll.

Vorschaubereich

Im Vorschaubereich (*Abbildung unten*) wird die Struktur der auf dem Register "Auswahl" ausgewählten Tabelle angezeigt. Wenn auf dem Register "Auswahl" eine neue Tabelle ausgewählt wird, klicken Sie im Bereich "Vorschau" auf die Schaltfläche "Vorschau", um die Tabelle anzuzeigen. Klicken Sie auf die Schaltfläche "**Neu laden**", um die Vorschau zu aktualisieren.

 © 2016-2022 Altova GmbH



Sie können festlegen, ob ein Feld als Element oder Attribut oder gar nicht importiert werden soll. Klicken Sie dazu auf das Symbol links vom Spaltennamen. Klicken Sie darauf, um der Reihe nach die Optionen "Element", "Attribut" und "Ignorieren" zu aktivieren. In der Abbildung oben wurde das Feld `city` für den Import als Element und das Feld `street` als Attribut definiert.

Datentyp-Konvertierungen

Informationen zu Konvertierung von Datenbank-Datentypen in XML-Schema-Datentypen finden Sie in den Anhängen[2256].

28.11.5 DB-Import auf Basis von XML-Schema



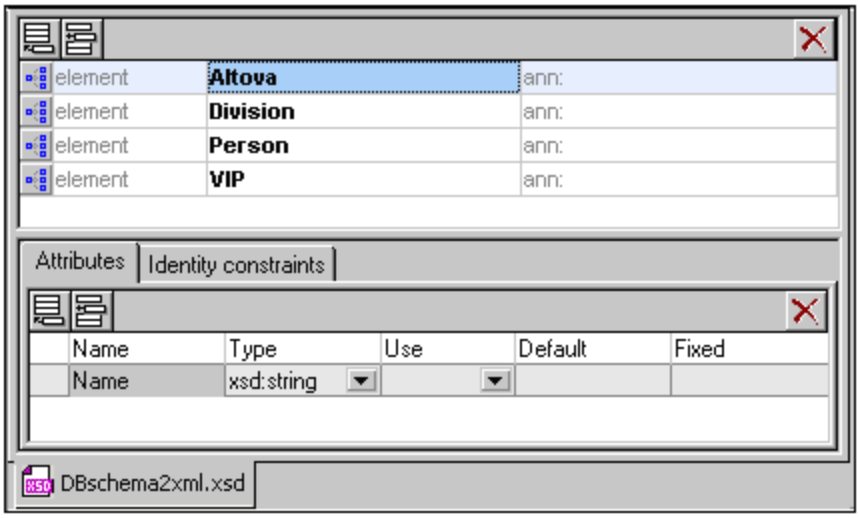
Mit Hilfe des Befehls **DB-Import auf Basis von XML-Schema** wird ein XML-Dokument erstellt, welches gemäß einem bestimmten XML-Schema gültig ist und aus einer Datenbank importierte Daten enthält. Diese Funktion unterstützt die folgenden Datenbanken:

- Microsoft Access 2000 und 2003
- Microsoft SQL Server
- Oracle
- MySQL
- Sybase
- IBM DB2

Welche Daten importiert werden, hängt davon ab, welche Tabelle in der Datenbank ausgewählt wurde. Stellen Sie eine Verbindung zur Datenbank her, wobei das gewünschte XML-Schema (d.h. das Schema auf Basis dessen der Import durchgeführt werden soll), in der Schema-Ansicht aktiv sein muss. Wählen Sie anschließend die gewünschte(n) Tabelle(n) aus und klicken Sie auf "Import". Die Daten werden in ein XML-Dokument importiert. Das Dokument hat die Struktur der Schemas, das beim Import der Daten aktiv war.

Im unten gezeigten Beispiel werden Daten aus der MS Access Datenbank `DB2schema.mdb` importiert, während in der Schema-Ansicht ein XML-Schema aktiv ist. Im Folgenden sind die Schritte beschrieben, die beim Import durchgeführt werden müssen:

1. Öffnen Sie die Schema-Datei in der Schema-Ansicht (*Abbildung unten*).



2. Wählen Sie den Menübefehl **DB-Import auf Basis von XML-Schema**. Daraufhin wird der Verbindungsassistent[969] geöffnet.
3. Wählen Sie die Option Microsoft Access (ADO) und klicken Sie auf **Weiter**.
4. Klicken Sie auf **Durchsuchen** und wählen Sie die Datenbankdatei und klicken Sie anschließend auf **Weiter**.

5. Klicken Sie im Dialogfeld "DB-Import auf Basis von XML-Schema" auf das Register "Tabellen", wählen Sie eine oder mehrere Tabellen aus, die Sie importieren möchten (z.B. `Altova`) und klicken Sie anschließend auf **Import**. Die Tabelle wird in ein XML-Dokument importiert, das in der Grid-Ansicht angezeigt wird.

Datentyp-Konvertierungen

Informationen zu Konvertierung von Datenbank-Datentypen in XML-Schema-Datentypen finden Sie in den Anhängen[2136].

28.11.6 DB-Struktur von XML-Schema erzeugen



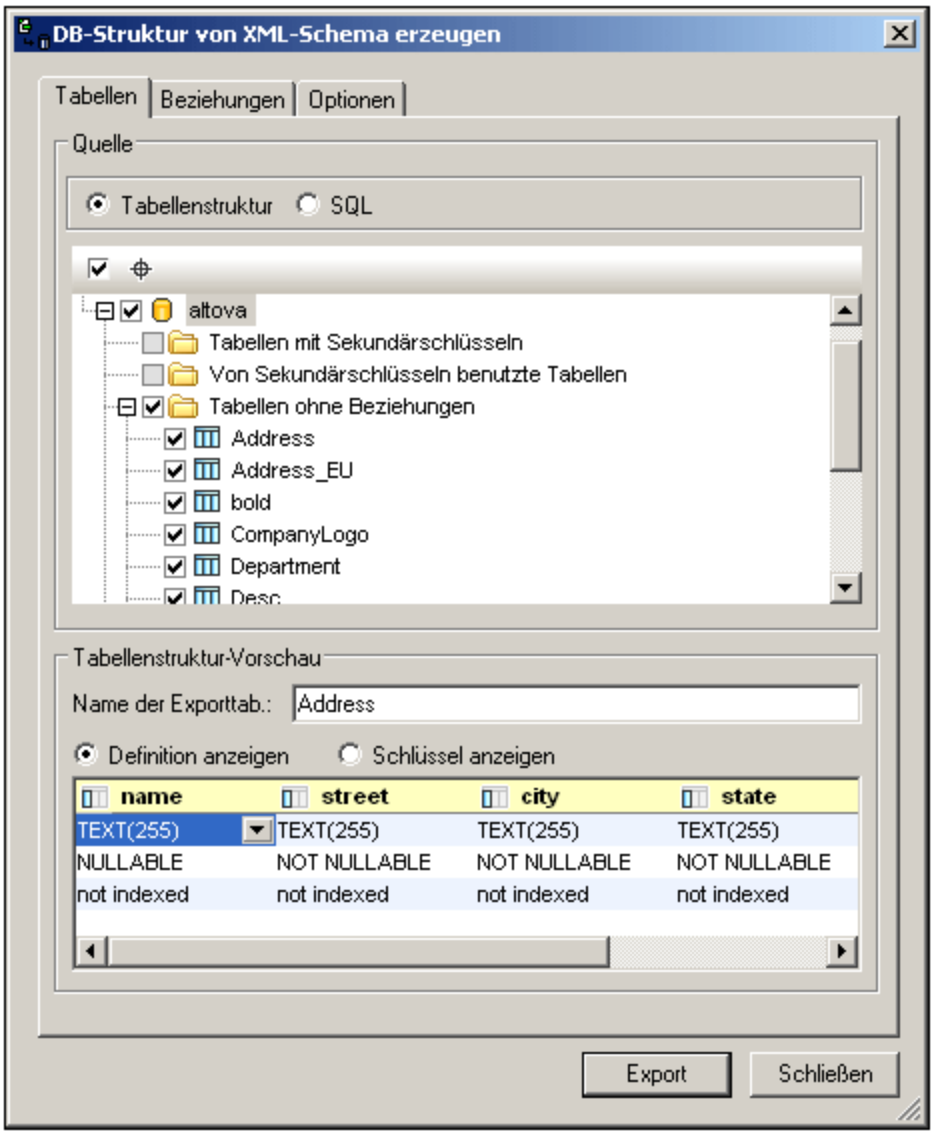
In XMLSpy können Sie auf Basis einer vorhandenen Schema-Datei eine leere Datenbank (oder ein Datenbankgerüst) erstellen.

1. Öffnen Sie die Schema-Datei in der Schema-Ansicht.
2. Wählen Sie den Menübefehl **Konvertieren | DB-Struktur von XML-Schema erzeugen**. Daraufhin wird das Dialogfeld Verbindung zur Datenquelle herstellen[969] aufgerufen, in dem Sie eine Verbindung zu einer Datenbank (DB) herstellen können.
3. Gehen Sie vor, wie unter Herstellen einer Verbindung zu einer Datenquelle[969] beschrieben, um eine Verbindung zur gewünschten Datenbank herzustellen. Um z.B. eine Verbindung zu einer Microsoft

 © 2016-2022 Altova GmbH

Access-Datenbank herzustellen, aktivieren Sie das Optionsfeld "Microsoft Access" und fahren Sie fort, um eine Datenbank auszuwählen. Sie können eine vorhandene Datenbank verwenden oder eine neue Datenbank erstellen

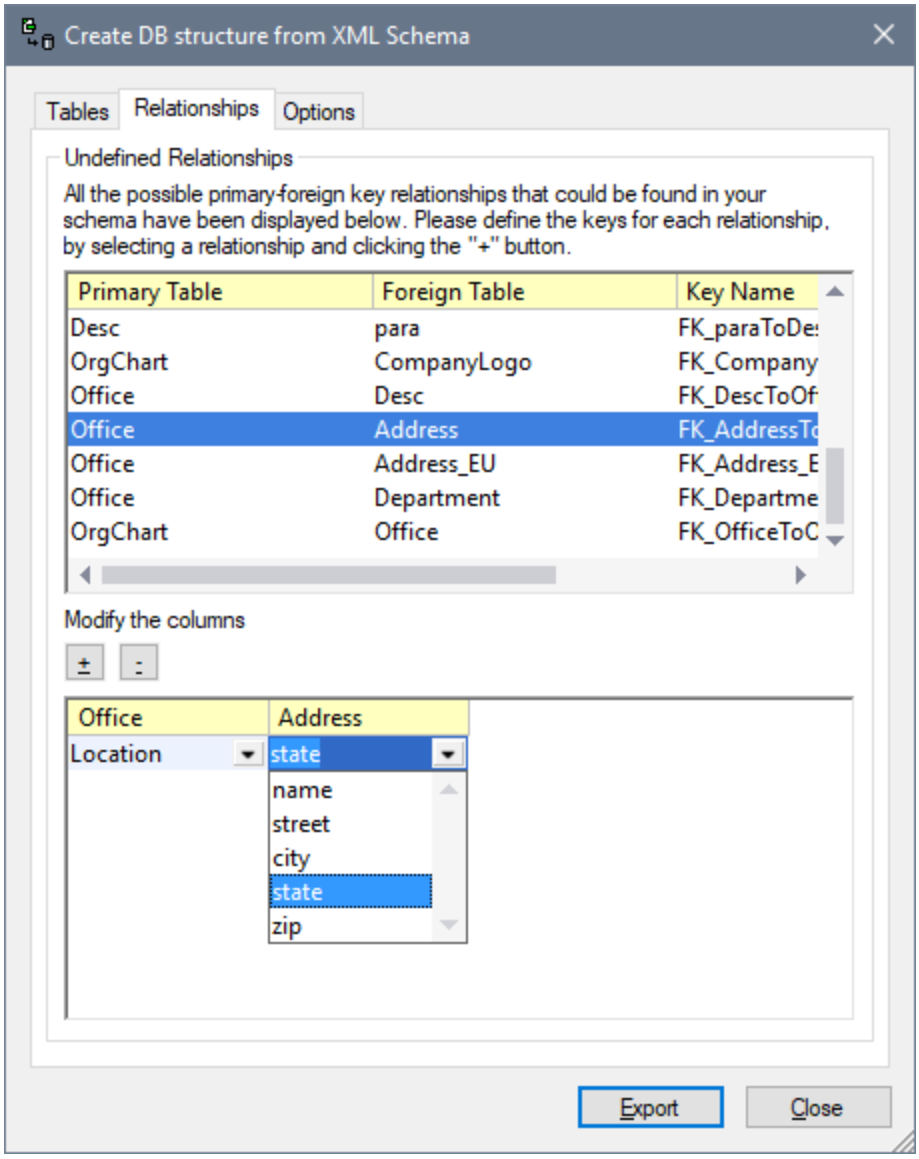
4. Im Dialogfeld "DB-Struktur von XML-Schema erzeugen" werden anhand des Schemas Tabellen erstellt und in hierarchischer Form an der Stelle in der Datenbank angezeigt, wo sie in der DB vorkommen sollen. In der Abbildung unten wird z.B. die Tabelle "Address" erstellt und für den Export ausgewählt. Tabellen, die nicht exportiert werden sollen, müssen deselektiert werden (durch Deaktivieren des Kontrollkästchens oder Auswahl des entsprechenden Eintrags für die Tabelle aus dem Kontextmenü).



Erstellen der DB-Tabellen mit Beziehungen

Wenn das XML-Schema, anhand dessen die DB-Struktur generiert wird, Beziehungen aufweist, die in Form von Identity Constraints definiert sind, so werden diese Beziehungen automatisch in der generierten DB-Struktur erstellt und in der Tabellenstruktur angezeigt. Tabellen mit Beziehungen werden unter folgenden Abschnitten aufgelistet: "Tabellen mit Sekundärschlüsseln" und "Von Sekundärschlüsseln benutzte Tabellen". Tabellen ohne Beziehungen werden im Abschnitt "Tabellen ohne Beziehungen" aufgelistet.

Auf dem Register "Beziehungen" können Sie Tabellenbeziehungen erstellen und ändern. Auf dem Register werden alle möglichen Primär-/Sekundärschlüsselbeziehungen aufgelistet (*Abbildung unten*).



Um eine Beziehung zu erstellen, gehen Sie folgendermaßen vor:

1. Wählen Sie eine der möglichen Primär-/Sekundärschlüsselbeziehungen aus.
2. Klicken Sie im unteren Bereich des Dialogfelds auf die Plus-Schaltfläche, um eine Beziehung zu erstellen.
3. Wählen Sie die in jeder der beiden Tabellen benötigten Spalten aus den entsprechenden Dropdown-Listen aus.

Sie können eine Beziehung auch entfernen, indem Sie sie auswählen und anschließend auf die Minus-Schaltfläche klicken.

© 2016-2022 Altova GmbH

Anmerkungen zur Datenbankstruktur und zum Herstellen der Verbindung

Die durch die Identity Constraints definierte Schemastruktur entspricht der erzeugten Datenbank. In der Tabelle unten sehen Sie die Art der erstellten Datenbank, die Einschränkungen und die Verbindungsmethoden bei Verwendung des Befehls **DB-Struktur von XML-Schema erzeugen**.

	Direkt	**über ODBC**	**über ADO**
MS Access (2000 und 2003)	OK *	OK	OK
MS SQL Server	OK *	OK	OK
Oracle	OK *	OK	OK
MySQL	-	OK *	OK +
Sybase	-	OK *	OK
IBM DB2	-	OK *	OK

* *Empfohlene Verbindungsmethode für die jeweilige Datenbank.*
\+ *MySQL: Bei Herstellen der ADO-Verbindung auf Basis von ODBC wird empfohlen, entweder den Benutzer oder den System DSN zu verwenden.*
\- *Nicht unterstützt*

XMLSpy mappt sowohl hierarchisch als auch flach formatierte Schemas[1478]. XMLSpy erkennt beide Formate automatisch.
Das flache Format wird auf zwei verschiedene Arten auf SQL gemappt.

- SQL Server DB, Oracle DB oder Sybase DB:
  Bei einem Schema, das für eine der oben angeführten Datenbanken im flachen Format generiert wurde, wird der Schema-Katalogname extrahiert und im generierten SQL Script als DB-Name verwendet. Dies bedeutet, dass das erzeugte SQL Script an einer Ziel-DB ausgeführt wird, deren Name mit dem Schema-Katalognamen identisch sein muss.
- Access (2000 oder 2003), MySQL oder DB2 DB:
  Bei einem Schema, das für eine der oben angeführten Datenbanken im flachen Format generiert wurde, wird der Schema-Katalogname **ignoriert**, wenn das SQL Script generiert wird. Dies bedeutet, dass das erzeugte SQL Script an einer Ziel-DB ausgeführt wird, bei der der Benutzer eingeloggt war:

Datentyp-Konvertierungen

Informationen zu Konvertierung von Datenbank-Datentypen in XML-Schema-Datentypen finden Sie in den Anhängen[2256].

28.11.7 Export in Textdateien



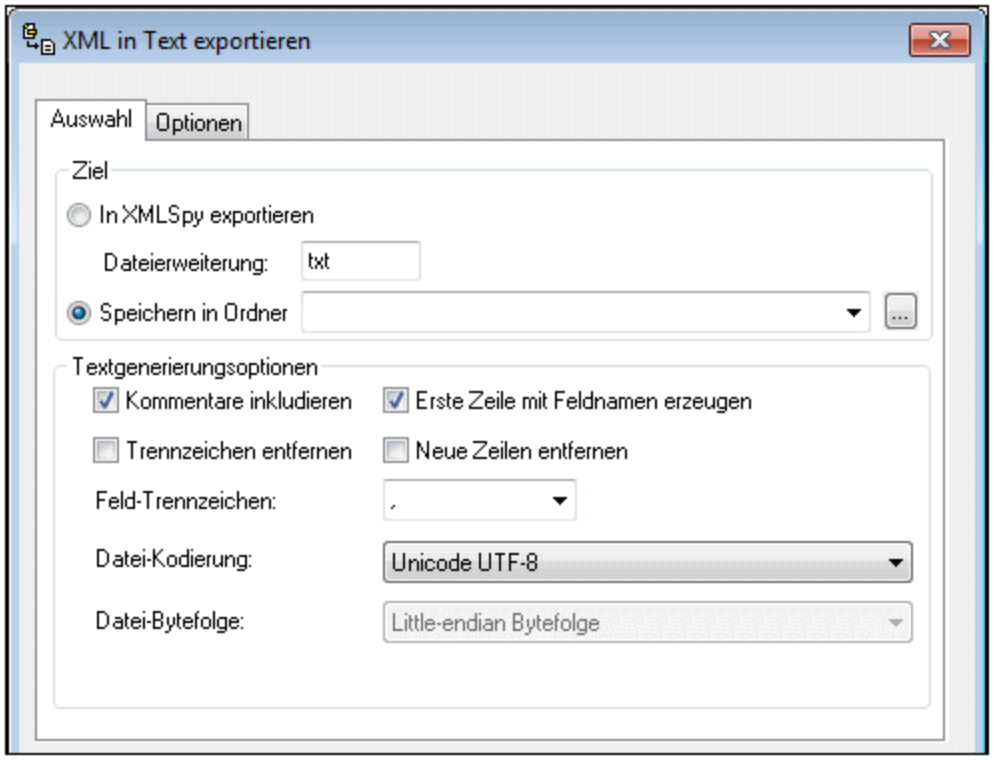
Mit dem Befehl **Konvertieren | Export in Textdateien** werden XML-Daten für den Austausch mit Datenbanken oder Altdatenbeständen in Textformate exportiert. Wenn Sie auf diesen Befehl klicken, wird das Dialogfeld "XML in Text exportieren" aufgerufen. Es besteht aus zwei Teilen (*die in der Abbildung unten separat angezeigt werden*):

- einem oberen Bereich mit zwei Registern: (i) "Auswahl" und (ii) "Optionen".
- einem unteren Bereich, dem Vorschaufenster.

Nachdem Sie die gewünschten Optionen in diesem Dialogfeld ausgewählt haben (*Beschreibung siehe unten*), klicken Sie zum Exportieren der Textdatei(en) auf die Schaltfläche **Export**.

Auswahl

Auf dem Register "Auswahl" (*Abbildung unten*) können Sie den Zielordner für die zu exportierende Datei sowie die Textgenerierungsoptionen auswählen.



Ziel: Die exportierte Datei kann direkt in einem Ordner gespeichert werden. Die Dateierweiterung kann angegeben werden Die Dateinamen sind jene der Elemente (in der XMl-Datei), die exportiert werden. Alternativ dazu können auch Dateien ohne Titel in XMLSpy exportiert werden. Diese Dateien werden auf der Benutzeroberfläche angezeigt und können später gespeichert werden.

Kommentare inkludieren: Aktivieren Sie diese Option, um einen von XMLSpy generierten Kommentar in die exportierte XML-Datei zu inkludieren. Der Kommentar enthält die zur Auswahl der Daten verwendete SQL-Abfrage sowie eine Liste mit einem Eintrag für jede Spaltenüberschrift in der Datenbanktabelle.

Erste Zeile mit Feldnamen erzeugen: Ist diese Option aktiviert, enthalten die exportierten Tabellen die Namen der Tabellen aus der Datenbank. Andernfalls werden die Spaltennamen nicht in die exportierte Textdatei inkludiert.

 © 2016-2022 Altova GmbH

Trennzeichen entfernen: Entfernt die in Textwerten in den exportierten Daten enthaltenen Trennzeichen. Definieren Sie auf diesem Register in der Auswahlliste "Trennzeichen", welches Trennzeichen entfernt werden soll. Wenn diese Option z.B. aktiviert ist und das ausgewählte Trennzeichen das Apostroph ist, wird bei Export des XML-Werts `Ba'ker` der String im exportierten Text `Baker` lauten.

Neue Zeilen entfernen: Neue Zeilen werden aus den exportierten Daten entfernt.

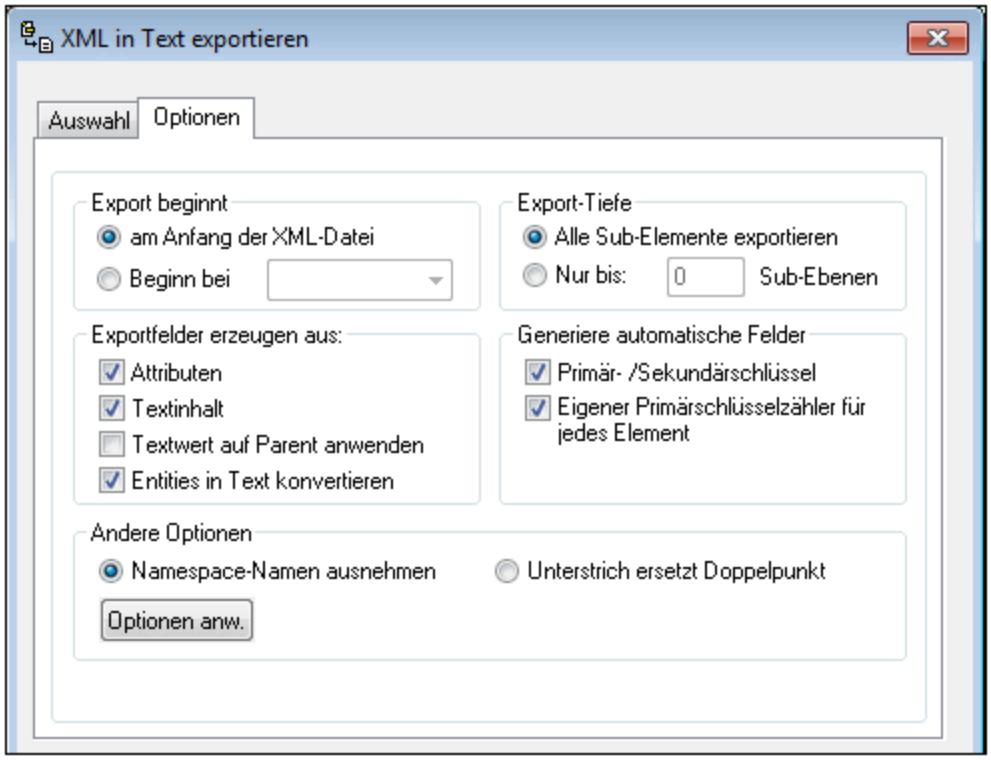
Trennzeichen: Wählen Sie in der Dropdown-Liste das Zeichen aus, das beim Export entfernt werden soll. Geben Sie alternativ dazu den gewünschten Zeichenstring ein.

Datei-Kodierung: Wählen Sie aus der Dropdown-Liste für die beim Export generierten Dateien das gewünschte Encoding aus.

Datei-Bytefolge: Beim Export von 16-Bit oder 32-Bit Unicode (UCS-2, UTF-16 oder UCS-4) Dateien können Sie auch zwischen den Bytefolgen Little-Endian und Big-Endian wechseln.

Exportoptionen

Weitere Export-Optionen (Beschreibung weiter unten) können auf dem Register "Optionen" (*Abbildung unten)* festgelegt werden.



Export beginnt: Sie können festlegen, ob das gesamte XML-Dokument exportiert werden soll oder Sie können den Export auf die Datenhierarchie, beginnend beim gerade ausgewählten Element beschränken. Die Anzahl der untergeordneten Ebenen unterhalb des zu exportierenden Startpunkts wird in der Option "Export-Tiefe" festgelegt.

Export-Tiefe: Gibt die Anzahl der untergeordneten Ebenen an, bis zu denen die Daten exportiert werden

sollen.

Exportfelder erzeugen aus: Abhängig von Ihren XML-Daten möchten Sie eventuell nur Elemente, Attribute oder den Textinhalt Ihrer Elemente exportieren. Beachten Sie, dass Sie den Export für einzelne Elemente im Vorschaufenster aufheben können.

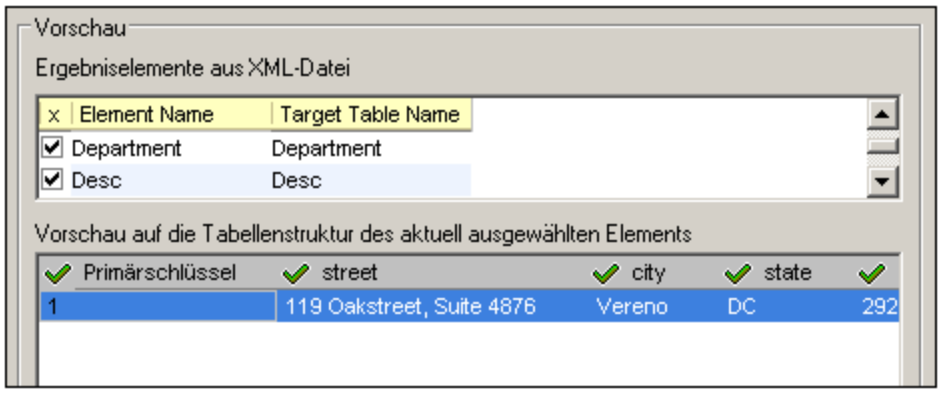
Generiere automatische Felder: XMLSpy erzeugt für jeden ausgewählten Elementtyp eine Ausgabedatei oder -tabelle. Sie können festlegen, ob zum Verknüpfen Ihrer Daten im relevanten Modell automatisch Primär-/Sekundärschlüsselpaare erstellt werden sollen oder ob für jedes Element ein Primärschlüssel erstellt werden soll.

Namespace-Namen ausnehmen: Zusammen mit dem Optionsfeld "Unterstrich ersetzt Doppelpunkt" bildet dies eine Entweder-Oder-Auswahl. Legt fest, ob Namespace-Präfixe von Elementen und Attributen ausgeschlossen werden sollen oder ob der Doppelpunkt im Namespace-Präfix durch einen Unterstrich ersetzt werden soll.

Optionen anwenden: Nachdem Sie die Optionen definiert haben, klicken Sie auf diese Schaltfläche, um die Optionen anzuwenden. Die Vorschau im Fenster "Vorschau" wird unter Verwendung der neuen Optionen aktualisiert.

Vorschaufenster

Das Vorschaufenster (*Abbildung unten)* wird unterhalb der Register "Auswahl" und "Optionen" angezeigt.



Im Bereich "*Ergebniselemente aus der XML-Datei*" werden die zu exportierenden Node-Namen sowie der jeweilige Name in der generierten Datei angezeigt. Sie können auswählen, welche Nodes exportiert bzw. welche nicht exportiert werden sollen. Wenn ein Element ausgewählt wird, wird im zweiten Bereich unterhalb davon eine Vorschau seiner Struktur angezeigt. Durch Anklicken der Kästchen links neben den Spaltennamen definieren Sie, welche Spalten exportiert werden sollen.

28.11.8 Export in Datenbank



 © 2016-2022 Altova GmbH

Mit dem Befehl **Konvertieren | Export in Datenbank** werden XML-Daten in eine Datenbank exportiert. Bei Auswahl dieses Befehls wird der Verbindungsassistent gestartet und gestattet Ihnen, eine Verbindung zu der Datenbank aufzubauen, die aktualisiert werden soll. Nach Herstellung der Verbindung wird das Dialogfeld "Daten in die Datenbank exportieren" angezeigt. Es besteht aus zwei Bereichen (*die in den Abbildungen unten separat gezeigt werden*):

- einem oberen Bereich mit zwei Registern: (i) Selection und (ii) Export Options.
- einem unteren Bereich, dem Vorschaufenster.

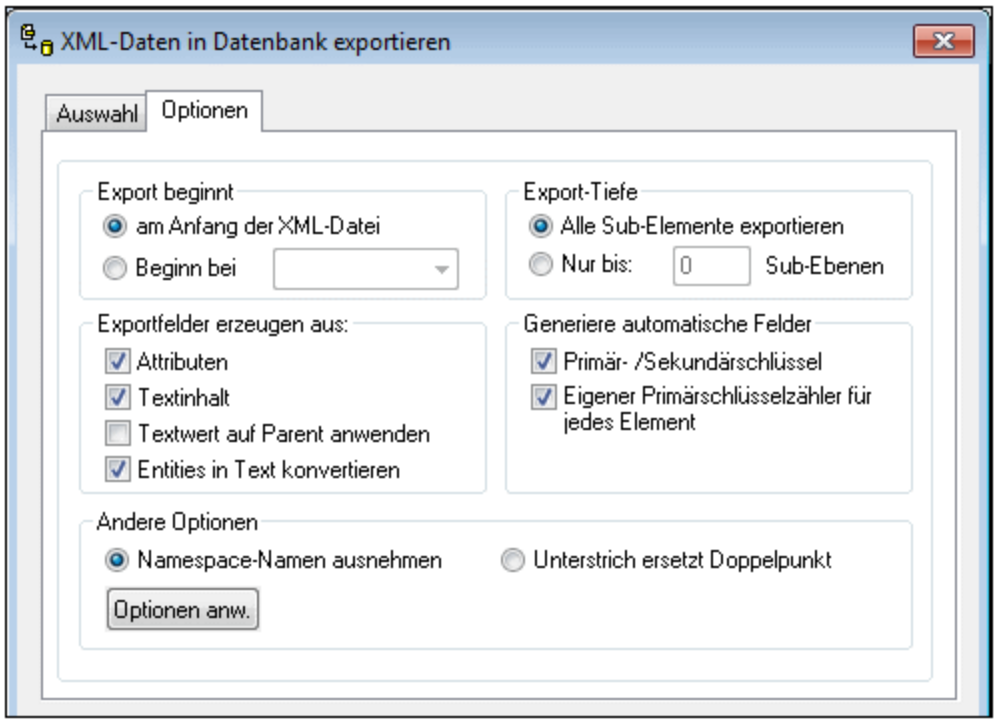
Nachdem Sie in diesem Dialogfeld (*Beschreibung siehe unten*) die gewünschten Optionen gewählt haben, klicken Sie auf die Schaltfläche **Exportieren**, um die Daten in die Datenbank zu exportieren.

Auswahl

Auf dem Register "Auswahl" können Sie die Zieldatenbank und die Tabellengenerierungsoptionen auswählen. Im Feld "Ziel" wird die Verbindung zur Datenbank ausgewählt. Sie müssen auswählen, ob die Daten als neue Tabellen erstellt werden sollen, ob bestehende Tabelle aktualisiert werden sollen oder ob zuerst versucht werden soll, eine bestehende Tabelle zu aktualisieren und anschließend eine neue Tabelle erstellt werden soll, falls das Aktualisieren nicht möglich ist. Außerdem können Sie auf Basis der Anzahl der Fehler eine Stopp-Aktion definieren und optional ein SQL Script Logging.

Exportoptionen

Auf dem Register "Optionen" (*Abbildung unten)* können Sie die unten beschriebenen Exportoptionen festlegen.



Export beginnt: Sie können festlegen, ob das gesamte XML-Dokument exportiert werden soll oder Sie können den Export auf die Datenhierarchie, beginnend beim gerade ausgewählten Element beschränken. Die Anzahl der untergeordneten Ebenen unterhalb des zu exportierenden Startpunkts wird in der Option "Export-Tiefe" festgelegt.

Export-Tiefe: Gibt die Anzahl der untergeordneten Ebenen an, bis zu denen die Daten exportiert werden sollen.

Exportfelder erzeugen aus: Abhängig von Ihren XML-Daten möchten Sie eventuell nur Elemente, Attribute oder den Textinhalt Ihrer Elemente exportieren. Beachten Sie, dass Sie den Export für einzelne Elemente im Vorschaufenster aufheben können.

Generiere automatische Felder: XMLSpy erzeugt für jeden ausgewählten Elementtyp eine Ausgabedatei oder -tabelle. Sie können festlegen, ob zum Verknüpfen Ihrer Daten im relevanten Modell automatisch Primär-/Sekundärschlüsselpaare erstellt werden sollen oder ob für jedes Element ein Primärschlüssel erstellt werden soll.

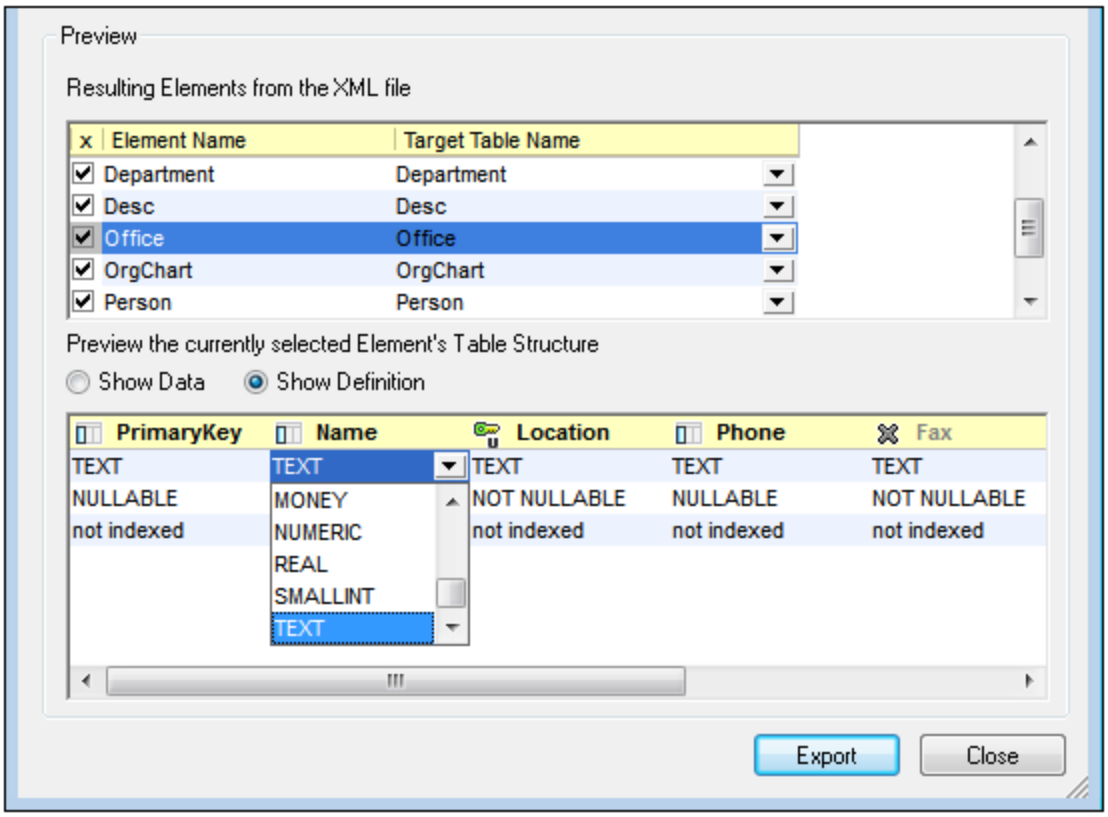
Namespace-Namen ausnehmen: Zusammen mit dem Optionsfeld "Unterstrich ersetzt Doppelpunkt" bildet dies eine Entweder-Oder-Auswahl. Legt fest, ob Namespace-Präfixe von Elementen und Attributen ausgeschlossen werden sollen oder ob der Doppelpunkt im Namespace-Präfix durch einen Unterstrich ersetzt werden soll.

Optionen anwenden: Nachdem Sie die Optionen definiert haben, klicken Sie auf diese Schaltfläche, um die Optionen anzuwenden. Die Vorschau im Fenster "Vorschau" wird unter Verwendung der neuen Optionen aktualisiert.

Vorschaufenster

Das Vorschaufenster (*Abbildung unten)* wird unterhalb der Register "Auswahl" und "Optionen" angezeigt.

© 2016-2022 Altova GmbH



Im Bereich "*Ergebniselemente aus der XML-Datei*" werden die zu exportierenden Node-Namen sowie der jeweilige Name in der generierten Datei angezeigt. Sie können auswählen, welche Nodes exportiert bzw. welche nicht exportiert werden sollen. Wenn ein Element ausgewählt wird, wird im zweiten Bereich unterhalb davon eine Vorschau seiner Struktur angezeigt. Sie können zwischen einer Vorschau (i) der Daten in der generierten Struktur (*Daten anzeigen*) und (ii) der Definition der einzelnen Spalten in der generierten Struktur (*Definition anzeigen*) wechseln. In der Spalte oben sehen Sie die Spaltendefinitionen.

Wenn Sie in diesem zweiten Bereich auf das Symbol links vom Spaltennamen klicken, aktivieren Sie der Reihe nach vier Einstellungen: (i) In die Tabellenstruktur inkludieren; (ii) Eindeutiger Constraint; (iii) Primärschlüssel-Constraint; (iv) Von Tabellenstruktur ausschließen. In der Abbildung oben hat die Spalte `Location` einen eindeutigen Constraint, während die Spalte `Fax` aus der Tabellenstruktur ausgeschlossen wurde. Alle anderen Spalten sind in die Tabellenstruktur inkludiert.

Wenn in der Struktur der Elementtabelle Felddefinitionen angezeigt werden (*Definition anzeigen*), können Sie die Definitionen bearbeiten, indem Sie die Definition auswählen und eine Option aus der Auswahlliste der Definition auswählen (*siehe Abbildung oben*).

28.11.9 XML-Instanz von/in JSON konvertieren

Wenn es sich beim aktiven Dokument um ein XML-Dokument handelt, generiert dieser Befehl ein JSON-Dokument anhand dieses Dokuments. Wenn es sich beim aktiven Dokument um ein JSON-Dokument handelt, wird mit dem Befehl ein XML-Dokument anhand dieses Dokuments generiert. Das generierte Dokument wird in einem neuen Fenster geöffnet und kann anschließend in einem beliebigen Ordner gespeichert werden. Im

Folgenden sind die Konvertierungsoptionen beschrieben. Nähere Informationen zu JSON und der Unterstützung für die Bearbeitung von JSON-Dateien in XMLSpy finden Sie im Abschnitt JSON und JSON-Schema 676.

Beispielkonvertierungen

Unten sehen Sie ein Beispiel für ein XML-Quelldokument. Unterhalb davon sehen Sie das JSON-Dokument, das mit dem Befehl **XML-Instanz von/in JSON konvertieren** generiert wurde.

XML-Dokument

```
<?xml version="1.0" encoding="UTF-8"?>
<Person first="Jim" last="James">
  <Address>
    <street>4 New Street</street>
    <city>New York</city>
    <state>NY</state>
    <code>10123</code>
  </Address>
  <Tel type="home">
      123 123-1234
  </Tel>
  <Tel type="office">
      123 987-9876
  </Tel>
</Person>
```

JSON-Dokument

```
{
       "XML": {
              "version": 1.0,
              "encoding": "UTF-8"
       },
       "Person": {
              "first": "Jim",
              "last": "James",
              "Address": {
                     "street": "4 New Street",
                     "city": "New York",
                     "state": "NY",
                     "code": 10123
              },
              "Tel": [ { "type": "home",
                     "Text": "\r      123 123-1234\r  "}, { "type": "office",
                     "Text": "\r      123 987-9876\r  "} ]
       }
}
```

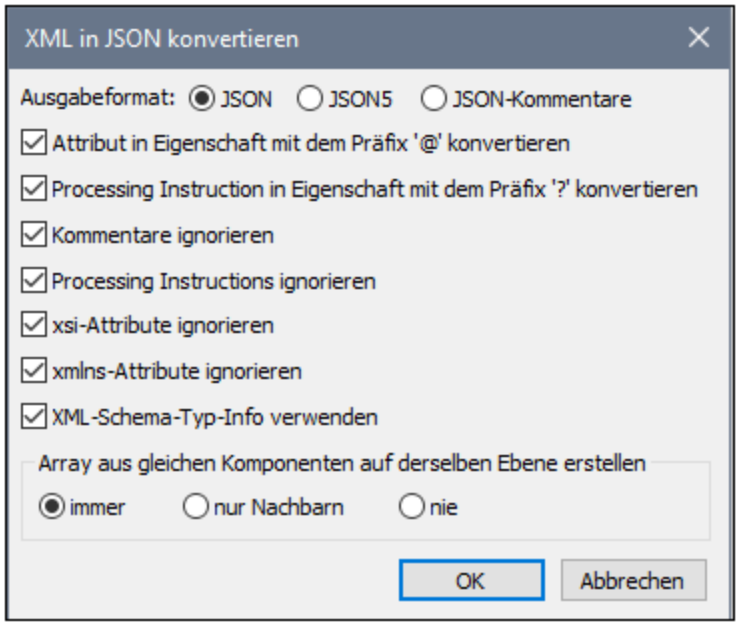
Um ein JSON-Dokument in XML zu konvertieren, machen Sie das JSON-Dokument zum aktiven und klicken Sie auf den Befehl **XML-Instanz von/in JSON konvertieren.**

Optionen für die Konvertierung von XML in JSON

Wenn Sie den Befehl **XML-Instanz von/in JSON konvertieren** auswählen, um ein XML-Instanzdokument in ein JSON-Instanzdokument zu konvertieren, wird das Dialogfeld "XML in JSON konvertieren" (*Abbildung unten*) angezeigt. Sie können auswählen, ob Sie JSON, JSON5 oder JSON-Kommentare konvertieren möchten.

 © 2016-2022 Altova GmbH

Aktivieren Sie die gewünschten Konvertierungsoptionen und klicken Sie auf **OK**. Daraufhin wird anhand der XML-Instanz ein JSON-Instanzdokument generiert und das generierte JSON-Dokument wird in einem neuen Fenster geöffnet.



Mit den beiden ersten Optionen wird definiert, ob Präfixe zu JSON-Eigenschaftsnamen hinzugefügt werden sollen, damit es zu keinen Konflikten zwischen Elementen auf derselben Ebene kommt. Die beiden Codefragmente unten veranschaulichen dies. Das XML-Attribut `somenode` wurde in die JSON-Eigenschaft `@somenode` konvertiert. Auf diese Art wird ein Konflikt mit der (anhand des XML-Elements `somenode` erstellten) JSON-Eigenschaft `somenode` verhindert.

XML-Instanz

```
<root somenode="value">
    <somenode>content</somenode>
</root>
```

JSON-Instanz

```
{
    "root": {
        "@somenode": "value",
        "somenode": "content"
    }
}
```

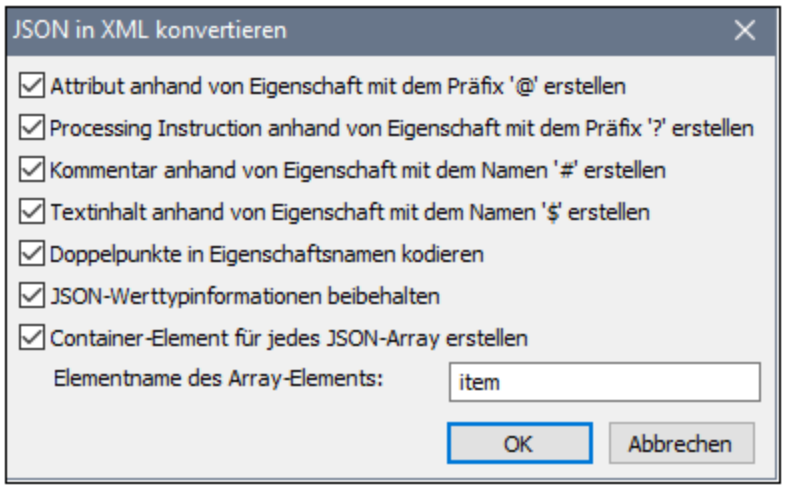
Mit den nächsten Optionen können Sie definieren, ob bestimmte Arten von XML-Nodes konvertiert werden sollen oder nicht. Wenn XML-Kommentare inkludiert werden, erhalten diese den Namen `"#"`. Text-Nodes (die normalerweise in Elementen mit gemischtem Inhalt vorkommen) erhalten den Namen `"$"`. Wenn ein XML-Node ein Namespace-Präfix hat, so wird der entsprechende JSON-Name mit diesem Namespace-Präfix erstellt. Wenn es auf ein und derselben Ebene Elemente mit demselben Namen gibt, werden diese als gleiche

Komponenten betrachtet. Auch Nodes wie z.B. Kommentare, Processing Instructions und Text() auf derselben Ebene sind gleiche Komponenten. Wenn auf derselben Ebene gleiche Komponenten vorhanden sind, können Sie auswählen, ob ein Array erstellt werden soll oder nicht. Die Optionen sind, ob ein Array aus allen gleichen Komponenten, nur aus benachbarten gleichen Komponenten oder ob gar kein Array erstellt werden soll.

Bei Verwendung der Option *XML-Schema-Typ-Info verwenden* können Konvertierungen auf Basis des XML-Schema-Typs des Quell-Node durchgeführt werden. Wenn ein Node z.B. im Schema als `xs:string`-Typ definiert ist, erhält die Eigenschaft des JSON-Zielobjekts den JSON-Typ `string` und wird in Anführungszeichen gesetzt. Dies ist z.B. nützlich, wenn eine Zahl im XML-Quell-Node als String gespeichert ist und bei der Konvertierung in JSON der genaue Typ beibehalten werden soll.

Optionen für die Konvertierung von JSON in XML

Wenn Sie den Befehl **XML-Instanz von/in JSON konvertieren** auswählen, um ein JSON-Instanzdokument in ein XML-Instanzdokument zu konvertieren, wird das Dialogfeld "JSON in XML konvertieren" (*Abbildung unten*) angezeigt. Aktivieren Sie die gewünschten Konvertierungsoptionen und klicken Sie auf **OK**. Daraufhin wird anhand der JSON-Instanz ein XML-Instanzdokument generiert und das generierte XML-Dokument wird in einem neuen Fenster geöffnet.



Beachten Sie die folgenden Punkte:

- JSON-Objekteigenschaften werden in XML-Elemente konvertiert. Mit Hilfe der ersten Optionen im Dialogfeld können Sie auswählen, ob einige Arten von Eigenschaften erstellt werden sollen oder nicht.
- *Doppelpunkte in Eigenschaftsnamen kodieren:* Bei Auswahl dieser Option werden Doppelpunkte in JSON-Namen kodiert und nicht als Doppelpunkte erstellt. Falls die Option deaktiviert ist, werden die Doppelpunkte beibehalten.
- *JSON-Werttypinformationen beibehalten:* Bei Auswahl dieser Option werden die JSON-Typinformationen einer Eigenschaft als Attribut-Wert-Paar des entsprechenden Elements erstellt.
- *Container-Element für jedes JSON-Array erstellen:* Das Container-Element in der XML-Dokumentation erhält den Namen des JSON-Array-Objekts. Die Elemente des JSON-Array werden als XML-Elemente in diesem Container erstellt. Jedes erhält im Textfeld *Elementname des Array-Elements* definierten Namen.

© 2016-2022 Altova GmbH

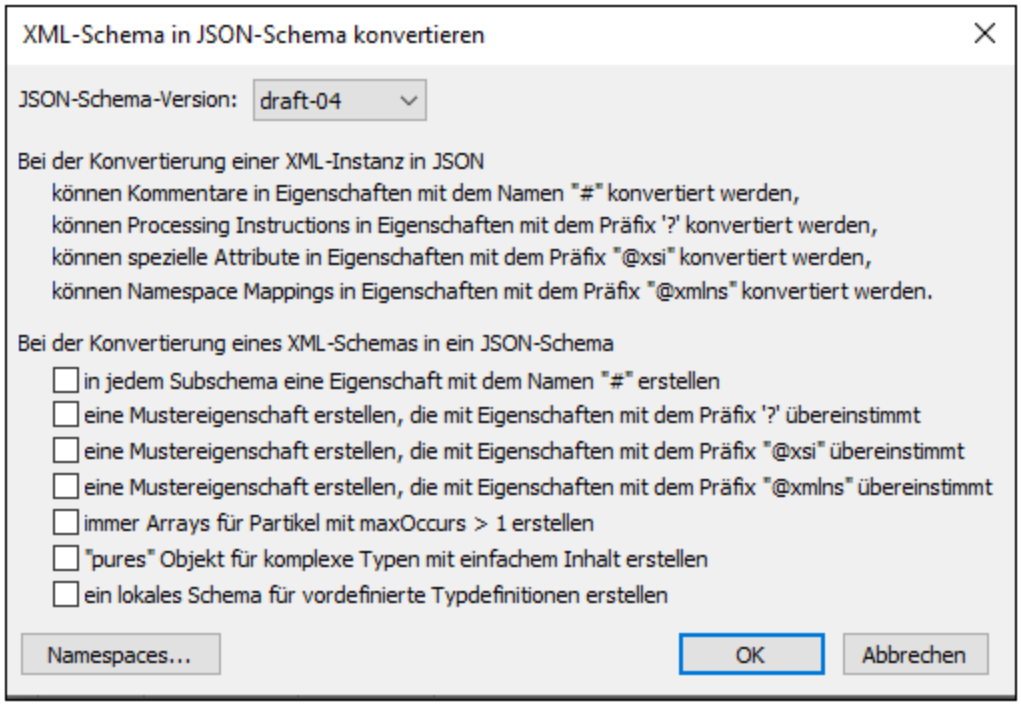
28.11.10 XML-Schema von/in JSON-Schema konvertieren

Wenn es sich beim aktiven Dokument um ein XML-Schema handelt, wird mit diesem Befehl ein JSON-Schema anhand des XML-Schemas generiert. Wenn es sich beim aktiven Dokument um ein JSON-Schema handelt, wird mit dem Befehl ein XML-Schema anhand des JSON-Schemas generiert. Das generierte Dokument wird in einem neuen Fenster geöffnet und kann anschließend in einem beliebigen Ordner gespeichert werden. Im Folgenden sind die Konvertierungsoptionen beschrieben. Nähere Informationen zu JSON und der Unterstützung für die Bearbeitung von JSON-Dateien in XMLSpy finden Sie im Abschnitt JSON und JSON-Schema 676.

Optionen für die Konvertierung von XML-Schemas in JSON-Schemas

Wenn Sie den Befehl **XML-Schema von/in JSON-Schema konvertieren** auswählen, um ein XML-Schema-Dokument in ein JSON-Schema-Dokument zu konvertieren, wird das Dialogfeld "XML-Schema in JSON-Schema konvertieren" (*Abbildung unten*) angezeigt. Wählen Sie die gewünschte JSON-Schema-Version 727 aus und klicken Sie auf **OK**. Daraufhin wird anhand des XML-Schemas ein JSON-Schema generiert und das generierte Dokument wird in einem neuen Fenster geöffnet.

Allgemein wird dabei die folgende Konvertierungsstrategie verwendet: (i) XML Schema SimpleTypes werden auf JSON Schema SimpleTypes (wie String und Zahl) gemappt; (ii) XML Schema ComplexTypes werden auf JSON-Objekte gemappt.



Im oberen Teil des Dialogfelds finden Sie Informationen darüber, wie bestimmte XML-Schemakomponenten konvertiert werden. Der untere Teil des Dialogfelds enthält die folgenden Optionen:

- *in jedem Subschema eine Eigenschaft mit dem Namen "#" erstellen:* Falls diese Option aktiviert ist, wird in jeder JSON-Schema-Definition eine Eigenschaft mit diesem Namen erstellt.
- *Mustereigenschaften erstellen, die mit Eigenschaften mit dem Präfix '?', "@xsi", "@xmlns" übereinstimmen:* Legt für jedes dieser Präfixe eine Mustereigenschaft für Eigenschaften mit Namen,

die diese Präfixe haben, fest. Nähere Informationen zu Mustereigenschaften finden Sie im Abschnitt JSON-Objekte und -Eigenschaften[737].

- *immer Arrays für Partikel mit maxOccurs > 1 erstellen:* Partikel sind in XML-Schema Elemente komplexer Inhaltsmodelle. Wenn die Anzahl der Instanzen mehr als eine beträgt, werden die Partikel in JSON-Schema als Array definiert. Andernfalls werden sie als Eigenschaften eines JSON-Objekts definiert.
- *"pure" Objekt für komplexe Typen mit einfachem Inhalt erstellen:* Der ComplexType von XML-Schema mit einfachem Inhalt ist ein Typ, der Attribute und Textinhalt gestattet, nicht aber Child-Elemente. Wenn die Option *"pure" Objekt erstellen* aktiviert ist, wird der ComplexType in ein JSON-Objekt konvertiert. Die Attribute des Typs werden in Eigenschaften des JSON-Objekts konvertiert, wobei den Eigenschaftsnamen das Präfix `@` vorangestellt wird. Für den Textinhalt des Typs wird eine Eigenschaft namens `$` generiert. Wenn die Option *"pure" Objekt erstellen* nicht aktiviert ist, wird der ComplexType in ein Objekt konvertiert, das andere Objekte und JSON SimpleTypes wie String und Zahl enthalten kann.
- *Ein lokales Schema für vordefinierte Typdefinitionen erstellen* Wenn diese Option aktiviert ist, wird die Typdefinition im Objekt selbst erstellt. Andernfalls ist die Typdefinition eine Referenz auf ein separates Objekt. In den beiden Codefragmenten im unten gezeigten JSON-Schema sehen Sie die beiden Ausgabeoptionen.

Typdefinition wird referenziert:

```
"properties": {
    "AccountManager": {
    "$ref": "#/definitions/xs:string"
    }
}
"xs:string": {
    "type": "string"
}
```

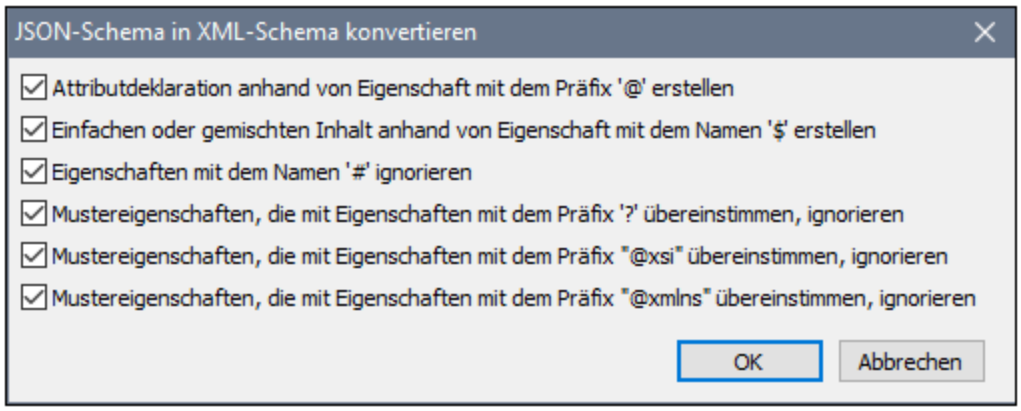
Typdefinition ist lokal:

```
"properties": {
    "AccountManager": {
    "type": "string"
    }
}
```

Optionen für die Konvertierung von JSON-Schemas in XML-Schemas

Wenn Sie den Befehl **XML-Schema von/in JSON-Schema konvertieren** auswählen, um ein JSON-Schema-Dokument in ein XML-Schema zu konvertieren, wird das Dialogfeld "JSON-Schema in XML-Schema konvertieren" (*Abbildung unten*) angezeigt. Aktivieren Sie die gewünschten Konvertierungsoptionen und klicken Sie auf **OK**. Daraufhin wird anhand des JSON-Schemas ein XML-Schema generiert und das generierte Dokument wird in einem neuen Fenster geöffnet.

© 2016-2022 Altova GmbH



Es stehen die folgenden Optionen zur Auswahl:

- ob JSON-Eigenschaftsnamen, die mit `'@'` und `'$'` beginnen, erstellt werden sollen oder nicht. Diese würde als Attribut- bzw. Text-Nodes erstellt.
- ob Eigenschaften mit dem Namen `'#'` als XML-Kommentar-Nodes erstellt werden sollen oder nicht.
- ob Mustereigenschaften, die mit Eigenschaften mit dem Präfixen `'?'`, `"@xsi"` und/oder `"@xmlns"` übereinstimmen, ignoriert werden sollen oder nicht. Falls diese nicht ignoriert werden, so werden Eigenschaften mit dem Präfixen `'?', "@xsi"` und `"@xmlns"` in Processing Instructions, Attribute mit dem Präfix `xsi:` bzw. Attribute mit dem Präfix `xmlns:` konvertiert.

28.11.11 In OIM xBRL-XML konvertieren

Konvertiert die folgenden Datenformate in OIM xBRL-XML.

- OIM xBRL-JSON
- OIM xBRL-CSV

Machen Sie die gewünschte Datei zur aktiven Datei und wählen Sie den Befehl anschließend aus.

Daraufhin wird die OIM xBRL-XML-Datei generiert, in einem neuen Fenster geöffnet und validiert. Sie können die generierte Datei anschließend in einem beliebigen Ordner speichern.

28.11.12 In OIM xBRL-JSON konvertieren

Konvertiert die folgenden Datenformate in OIM xBRL-JSON.

- XBRL-Datendatei
- OIM xBRL-XML
- OIM xBRL-CSV

Machen Sie die gewünschte Datei zur aktiven Datei und wählen Sie den Befehl anschließend aus.

Daraufhin wird die OIM xBRL-JSON-Datei generiert, in einem neuen Fenster geöffnet und validiert. Sie können die generierte Datei anschließend in einem beliebigen Ordner speichern.

28.11.13 In OIM xBRL-CSV konvertieren

Konvertiert die folgenden Datenformate in OIM xBRL-CSV.

- XBRL-Datendatei
- OIM xBRL-XML
- OIM xBRL-JSON

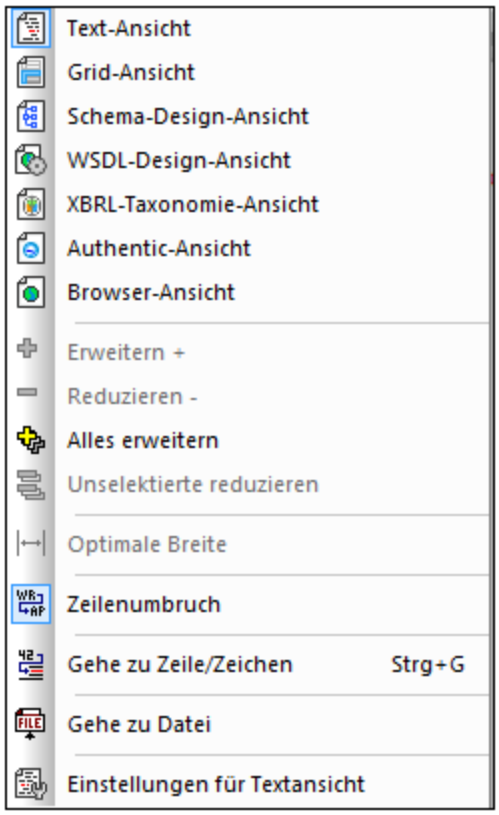
Machen Sie die gewünschte Datei zur aktiven Datei und wählen Sie den Befehl anschließend aus.

Sie werden nun nach einem Dateinamen gefragt, unter dem eine JSON-Datei gespeichert werden soll. Diese JSON-Datei enthält die Referenzen auf die CSV-Datendatei(en), die generiert werden. Nachdem 'Sie die JSON-Datei definiert und auf "Speichern" geklickt haben, werden die JSON- und die CSV-Datei generiert und die Validierungsergebnisse der generierten Dateien werden angezeigt.

© 2016-2022 Altova GmbH

28.12 Menü "Ansicht"

Im Menü **Ansicht** (*Abbildung unten*) kontrollieren Sie die Anzeige für das aktive Hauptfenster[116] und können die Art, wie ein Dokument angezeigt wird, ändern.



In diesem Abschnitt finden Sie eine Beschreibung aller Befehle im Menü "**Ansicht**".

28.12.1 Text-Ansicht

Mit diesem Befehl **wechseln** Sie für das aktuelle Dokument in die Textansicht[144]. In dieser Ansicht können Sie das Dokument in der Textform editieren. Die Text-Ansicht unterstützt eine Reihe von Editierfunktionen, die im Abschnitt Textansicht[144] näher beschrieben sind.

Anmerkung: Sie können Aspekte der Textansicht auf den verschiedenen Registern des Dialogfelds "Optionen" (**Extras | Optionen**[1602]) konfigurieren.

28.12.2 Grid Ansicht



Mit diesem Befehl **wechseln** Sie mit dem aktuellen Dokument in die Grid-Ansicht[169].
Wenn Sie von der Textansicht[144] in diese Ansicht wechseln, wird Ihr Dokument vorab auf Wohlgeformtheit überprüft.

XML	<?xml version="1.0" encoding="UTF-8"?>				
Company					
	xmlns	http://my-company.com/namespace			
	xmlns:xsi	http://www.w3.org/2001/XMLSchema-instance			
	xsi:schemaLocation	http://my-company.com/namespace AddressLast.xsd			
	Address				
		xsi:type	US-Address		
		Name	US dependency		
		Street	Noble Ave.		
		City	Dallas		
		Zip	04812		
		State	Texas		
	Person (3)		Manager	Degree	Programmer	First
		1	false	MA	true	Alfred
		2	true	Ph.D	false	Colin
		3	true	BA	false	Fred

28.12.3 Schema-Ansicht



Mit diesem Befehl **wechseln** Sie mit dem aktuellen Dokument - falls es sich dabei um ein XML-Schema-Dokument handelt - in die Schema-Ansicht. Nähere Informationen dazu erhalten Sie im Abschnitt Schema-Ansicht[171].

28.12.4 WSDL-Design-Ansicht



© 2016-2022 Altova GmbH

Mit diesem Befehl wechseln Sie mit dem aktuellen Dokument in die WSDL-Design-Ansicht, falls es ein WSDL-Dokument (mit der Dateierweiterung `.wsdl`) ist. Nähere Informationen dazu erhalten Sie im Abschnitt WSDL-Ansicht[253].

28.12.5 XBRL-Taxonomieansicht



Mit diesem Befehl wechseln Sie mit dem aktuellen Dokument in die XBRL-Taxonomieansicht, falls es sich beim Dokument um ein XBRL-Taxonomiedokument (mit der Dateierweiterung `.xsd`) handelt. Beachten Sie, dass XBRL-Instanzdokumente XML-Dateien sind, die Erweiterung `.xml` haben und daher nicht in der XBRL-Ansicht bearbeitet werden können, sondern wie normale XML-Dateien in anderen Ansichten bearbeitet werden müssen. Nähere Informationen dazu erhalten Sie im Abschnitt XBRL-Ansicht[266].

28.12.6 Authentic-Ansicht

Mit diesem Befehl wechseln Sie mit Ihrem Dokument in die Authentic[622]-Ansicht.

In der Authentic-Ansicht können Sie XML-Dokumente bearbeiten, die auf in Altova StyleVision erstellten StyleVision Power Stylesheet-Vorlagen basieren. Mit Hilfe dieser Vorlagen (StyleVision Stylesheets oder SPS-Dateien) werden XML-Dokumente in einem grafischen Format angezeigt, in dem das Dokument leichter als in einem mit Markup-Code versehenen Textformat bearbeitet werden kann.

Wenn ein XML-Dokument mit einer SPS-Datei verknüpft ist (**Authentic | StyleVision Stylesheet zuweisen**[1430]), kann das XML-Dokument in der Authentic-Ansicht angezeigt werden. Sie können auch eine SPS-Datei als neue Vorlage in der Authentic-Ansicht öffnen. Dafür gibt es zwei Methoden:

- Wählen Sie den Menübefehl **Datei | Neu** und klicken auf die Schaltfläche "**StyleVision Stylesheet auswählen**...".
- Wählen Sie den Befehl **Authentic | Neues Dokument** und navigieren Sie zur SPS-Datei.

Weitere Informationen finden Sie in der Dokumentation zur Authentic-Ansicht[606] und zu StyleVision.

28.12.7 Browser-Ansicht



Mit diesem Befehl wechseln Sie mit dem aktuellen Dokument in die Browser Ansicht[282]. Ein XML-fähiger Browser zeigt das XML-Dokument anhand der verfügbaren CSS- und/oder XSL-Stylesheets an.

Wenn Sie in die Browser Ansicht wechseln, wird das Dokument zuerst auf Gültigkeit geprüft, wenn Sie im Dialogfeld Optionen im Abschnitt "Datei"[1603] (**Extras | Optionen**) die Option "*Validieren beim Speichern der Datei*" einschalten. Weitere Informationen zu dieser Ansicht finden Sie im Abschnitt Browser Ansicht[282].

28.12.8 Erweitern



Dieser Befehl (*Tastenkürzel*: + am Ziffernblock) ist in der Grid-Ansicht aktiv und erweitert das ausgewählte Element um eine Ebene. Das Element bleibt nach dem Erweitern markiert, sodass Sie das Element durch wiederholtes Klicken auf die Taste '+' um weitere Ebenen erweitern können.

28.12.9 Reduzieren



Dieser Befehl (*Tastenkürzel*: - am Ziffernblock) ist in der Grid-Ansicht aktiv und reduziert das ausgewählte Element um eine Ebene. Sie können jedes Element durch Klicken auf die graue Leiste links vom Element erweitern bzw. reduzieren.

28.12.10 Alles erweitern



Dieser Befehl (Tastenkürzel: * oder x auf dem Ziffernblock) ist in der Grid- und der Text-Ansicht aktiv, wenn die Klappleiste aktiv ist. Mit diesem Befehl erweitern Sie alle Nachfahren-Nodes des ausgewählten Elements.

28.12.11 Unselektierte reduzieren



Dieser Befehl (Tastenkürzel: Strg + '-' auf dem Ziffernblock) ist in der Grid-Ansicht aktiv. Das ausgewählte Datenelement wird erweitert angezeigt, während alle anderen Datenelemente reduziert werden. Mit diesem Befehl können Sie sich auf ein Element und dessen Children konzentrieren und alle anderen Elemente ignorieren.

© 2016-2022 Altova GmbH

28.12.12 Optimale Breite



Dieser Befehl ist in der Grid-Ansicht aktiv. Sie stellen damit die Größe aller Spalten in der Grid-Ansicht so ein, dass jede Spalte so breit ist, dass der längste Text einer ihrer Zellen in einer Zeile darin Platz hat. Im Dialogfeld "Optionen" (**Extras | Optionen**) können Sie im Abschnitt "Ansicht" eine maximale optimale Breite definieren. Beachten Sie, dass die optimale Breite anhand der sichtbaren Zellen von Spalten berechnet wird. Dadurch können Sie die Ansicht optimieren, wenn einzelne Elemente reduziert oder erweitert werden.

28.12.13 Zeilenumbruch

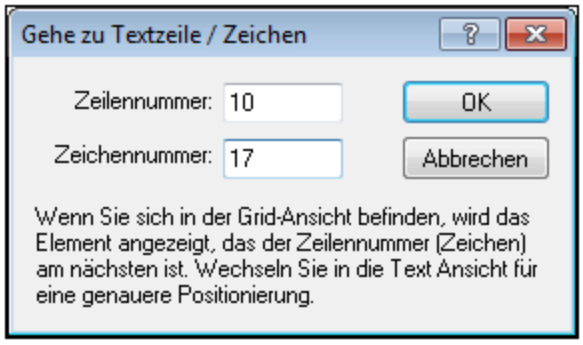


Mit diesem Befehl schalten Sie den Zeilenumbruch in der Textansicht ein bzw. aus. Wenn der Zeilenumbruch aktiviert ist, wird der Text am Rand des Fensters umbrochen.

28.12.14 Gehe zu Zeile/Zeichen



Dieser Befehl (Tastenkürzel: **Strg+G**) ist in der Text- und der Grid-Ansicht aktiv. Mit diesem Befehl rufen Sie ein Dialogfeld auf (*Abbildung unten*), in das Sie die Zeilen- und Zeichennummer, zu der Sie springen möchten, eingeben können. In der Textansicht springt der Cursor zur eingegebenen Position. In der Grid-Ansicht wird der Node, der der eingegebenen Zeilen- und/oder Zeichennummer am nächsten liegt, markiert.



Diese Funktion eignet sich, um schnell zu einer bestimmten Position, z.B. der Position eines in einer Fehlermeldung angegebenen Fehlers, zu navigieren.

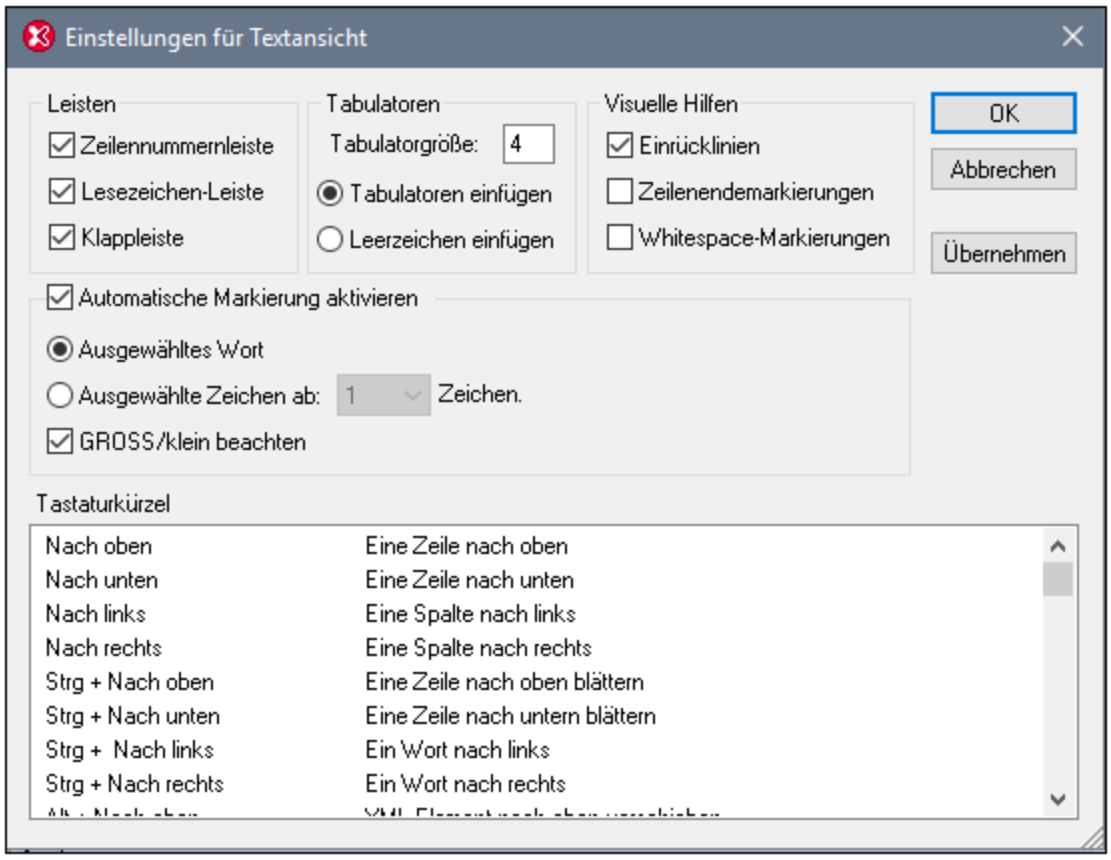
28.12.15 Gehe zu Datei

Dieser Befehl ist in der Text- und der Grid-Ansicht aktiv. Wenn der Cursor (in der Textansicht) innerhalb von Text platziert wird, der eine Datei referenziert oder wenn er (in der Grid-Ansicht) in einen Node platziert wird, der Text enthält, der eine Datei referenziert, so wird das referenzierte Dokument von der Datei aus, die Sie gerade bearbeiten, geöffnet.

28.12.16 Einstellungen für Textansicht

Der Befehl **Einstellungen für Textansicht** ist in der Textansicht aktiv. Mit diesem Befehl wird das Dialogfeld "Einstellungen für Textansicht" geöffnet (*Abbildung unten*), in dem Sie die Textansicht konfigurieren können.

Der Shortcut zum Befehl steht in Form einer Schaltfläche in der Symbolleiste "Text" zur Verfügung.



 © 2016-2022 Altova GmbH

Leisten

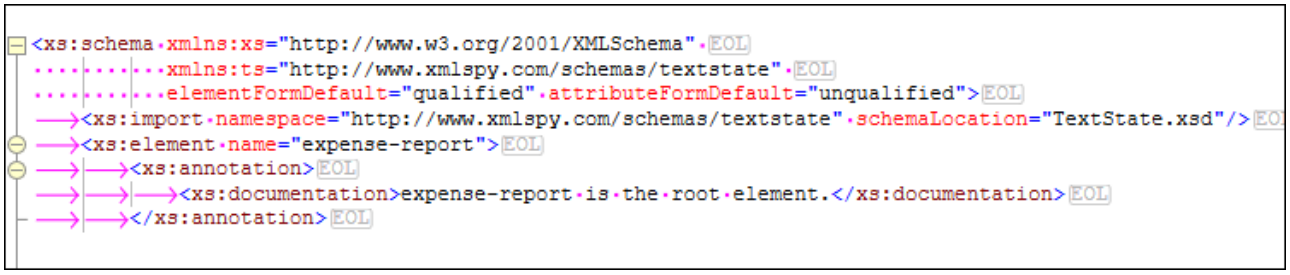
Im Bereich "Leisten" können Sie die Zeilennummernleiste, die Lesezeichenleiste und die Klappleiste ein- und ausblenden. Dabei handelt es sich um separate Leisten in der Textansicht, in denen (i) Zeilennummern, (ii) Lesezeichen bzw. (iii) die hierarchische Struktur (Symbole zum Ein- und Ausklappen von Nodes) angezeigt werden. Mit diesen Einstellungen legen Sie fest, ob die Leisten in der Textansicht angezeigt werden sollen oder nicht. Die Lesezeichenbefehle befinden sich im Menü **Bearbeiten**. Sie können Nodes in der Textansicht nur dann ein- ausklappen, wenn die *Klappleiste* eingeblendet ist.

Tabulatoren

Im Bereich "Tabulatoren" können Sie die Tabulatorgröße in Form der Anzahl von Leerzeichen definieren. Mit Hilfe der Optionsfelder unterhalb der Einstellung *Tabulatorgröße* legen Sie fest, ob Dokumente mit Tabulator- oder Leerzeicheneinrückungen angezeigt werden, wenn Pretty-Print mit Einrückung im Dialogfeld "'Optionen" im Abschnitt "Ansicht" 1617 (**Extras | Optionen**) aktiviert ist.

Visuelle Hilfen

Der Bereich "Visuelle Hilfen" enthält Einstellungen zum Ein- und Ausblenden der Einrücklinien (gepunktete vertikale Linien, die die Einrückung des Texts anzeigen, *siehe Abbildung unten*), Zeilenendezeichen sowie Whitespace-Markierungen (Tabulatoren und Leerzeichen). (Tabulatoren werden durch Pfeile angezeigt, während Leerzeichen durch Punkte gekennzeichnet werden (*in der Abbildung unten sind beide pink*). Die Farben von Whitespace-Markierungen können in den Textansichtsoptionen des Dialogfelds "Optionen" 1623 angepasst werden).



Automatische Markierung aktivieren

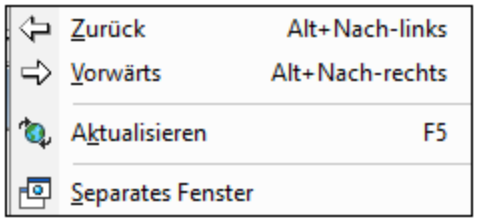
Wenn die Markierung aktiviert ist, werden alle Instanzen eines ausgewählten Bereichs in der Textansicht markiert. Über die Optionen in diesem Bereich können Sie festlegen, was als markierter Bereich gilt. Als markierter Bereich wird entweder ein ganzes Wort oder eine festgelegte Anzahl von Zeichen festgelegt, wobei die Groß- und Kleinschreibung dabei berücksichtigt wird oder nicht. Bei der Auswahl der Zeichen können Sie die Mindestanzahl der Zeichen für eine Übereinstimmung definieren (z.B. zwei oder mehr Zeichen). Alle Instanzen von Zeichensequenzen, die mit Ihrer Auswahl übereinstimmen, werden in der Textansicht markiert. Bei Wortsuchen werden Elementnamen, Attributnamen, Attributwerte ohne Anführungszeichen und die spitzen Klammern von Element-Tags als separate Wörter betrachtet.

Tastaturkürzel

Die Liste der Tastaturkürzel enthält die Tastaturkürzel und die damit verknüpften Befehle.

28.13 Menü "Browser"

Die Befehle im Menü **Browser** sind nur in der Browser Ansicht[282] aktiv. Die Schaltflächen **Zurück** und **Vorwärts** sind jedoch auch in der Schema-Ansicht aktiv.



Zurück, Vorwärts

Der Befehl **Zurück** (*Tastenkürzel:* **Alt + Pfeil nach links**) ist in der Browser- und der Schema-Ansicht aktiv.

- Mit diesem Befehl wird in der Browser-Ansicht eine zuvor angezeigte Seite aufgerufen. Mit der **Zurück**-Taste auf Ihrer Tastatur erreichen Sie denselben Effekt. Dieser Befehl ist hilfreich, wenn Sie auf einen Link in Ihrem XML-Dokument klicken und danach zu Ihrem XML-Dokument zurückkehren möchten.
- In der Schema-Ansicht gelangen Sie mit dem Befehl "Zurück" zur zuvor angezeigten Komponente oder Ansicht. Mit dem Befehl "Zurück" können Sie bis zu 500 zuvor angezeigte Positionen erneut anzeigen.

Der Befehl **Vorwärts** (*Tastenkürzel:* **Alt + Pfeil nach rechts**) ist in der Browser- und der Schema-Ansicht aktiv.

- Mit diesem Befehl gelangen sie in der Browser-Ansicht vorwärts durch vorher angezeigte Seiten.
- In der Schema-Ansicht gelangen Sie mit diesem Befehl vorwärts durch zuvor angezeigte Komponenten oder Ansichten. Mit diesem Befehl können Sie bis zu 500 zuvor angezeigte Positionen erneut anzeigen.

Aktualisieren

Der Befehl **Aktualisieren** (**F5**) ist in der Browser-Ansicht aktiv. Mit diesem Befehl aktualisieren Sie die Browser Ansicht durch erneutes Laden des aktuellen Dokuments sowie damit verknüpfter Dokumente (wie CSS und XSL Stylesheets und DTDs).

Separates Fenster

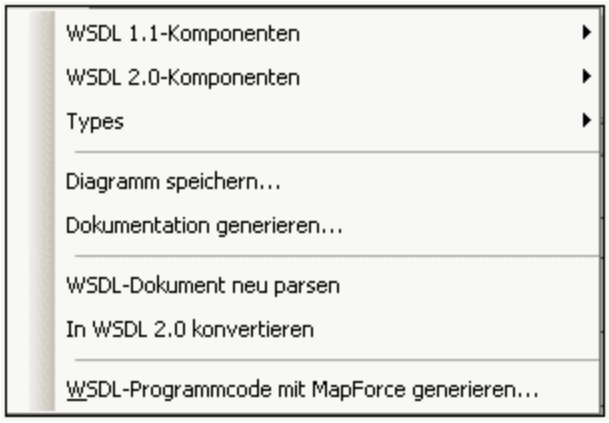
Dieser Befehl ist in der Browser-Ansicht aktiv. Mit diesem Befehl entkoppeln Sie die Browser-Ansicht vom Applikationsfenster. Dadurch kann die Browser-Ansicht als separates Fenster Seite an Seite mit der Bearbeitungsansicht des Dokuments angezeigt werden.

Um die separate Browser-Ansicht zu aktualisieren, nachdem Sie in der Bearbeitungsansicht eine Änderung vorgenommen haben, drücken Sie in der Bearbeitungsansicht die Taste **F5.** Um ein separates Browser-Fenster wieder im Applikationsfenster anzudocken, machen Sie die Browser-Ansicht zur aktiven Ansicht und wählen Sie den Befehl **Separates Fenster**.

© 2016-2022 Altova GmbH

28.14 Menü "WSDL"

Dieses Menü enthält Befehle, die zur Verfügung stehen, wenn Sie ein WSDL-Dokument in der WSDL-Ansicht[253], einem grafischen Editor zum Erstellen und Bearbeiten von WSDL-Dokumenten, anzeigen.

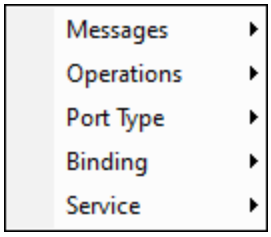


Eine Beschreibung der WSDL-Ansicht finden Sie im Abschnitt WSDL-Ansicht[253]. Eine kurze Anleitung für den Einstieg in WSDL finden Sie im WSDL Tutorial[789].

Siehe auch: Weitere Informationen zum Arbeiten mit WSDL-Dokumenten finden Sie in den Abschnitten WSDL-Ansicht[253] und WSDL Tutorial[789].

28.14.1 WSDL 1.1-Komponenten

Wenn Sie die Maus über den Menüeintrag **WSDL 1.1-Komponenten** platzieren, wird ein Untermenü eingeblendet (*Abbildung unten*), über das verschiedene Befehle zur Bearbeitung von WSDL 1.1-Komponenten ausgewählt werden können.



Jeder Befehl des Menüs WSDL 1.1 (*Abbildung oben*) enthält ein eigenes Untermenü, das Befehle im Zusammenhang mit dieser Komponente enthält. Die Befehle in den einzelnen Untermenüs werden in den Unterabschnitten dieses Abschnitts beschrieben.

28.14.1.1 Messages

Message einfügen

Eine neue Message wird in der WSDL Datei eingefügt. In der Eingabehilfe "Übersicht" wird der Message-Eintrag geöffnet und die neu erstellte Message erscheint dort markiert.

Message löschen

Löscht die ausgewählte Message aus dem Input- oder Outputelement.

Message Part (Parameter) einfügen

Ein Message Part (Parameter) wird zur ausgewählten Nachricht hinzugefügt.

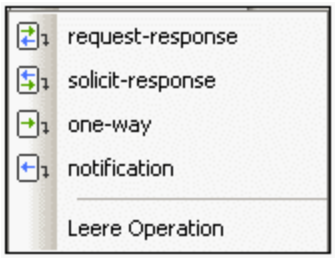
Message Part (Parameter) löschen

Ein Message Part (Parameter) wird aus der ausgewählten Nachricht gelöscht.

28.14.1.2 Operations

Operation anhängen

Hängt eine neue Operation an den ausgewählten PortType an. Die Art der anzuhängenden Operation können Sie im Untermenü des Befehls **Operation anhängen** (*Abbildung unten*) auswählen.



Operation löschen

Die ausgewählte PortType-Operation wird gelöscht.

Input-Element einfügen

Ein Input-Element wird zur ausgewählten PortType-Operation hinzugefügt.

Output-Element einfügen

Ein Output-Element wird zur ausgewählten PortType-Operation hinzugefügt.

© 2016-2022 Altova GmbH

Fault-Element einfügen

Ein Fault-Element wird zur ausgewählten PortType-Operation hinzugefügt.

Input/Output/Fault-Element löschen

Löscht das ausgewählte PortType-Input-, Output- oder Fault-Element.

Neue Message zu Input/Output/Fault-Element hinzufügen

Eine neue (Default) Message wird zum ausgewählten PortType-Input-, Output- oder Fault-Element hinzugefügt.

28.14.1.3 PortType

PortType einfügen

Ein neuer Porttyp wird im Hauptfenster zur Spalte"Porttypes" hinzugefügt.

PortType löschen

Der ausgewählte Porttyp wird im Hauptfenster aus der Spalte"Porttypes" gelöscht.

28.14.1.4 Binding

Binding einfügen

Ein neues Binding wird im Hauptfenster zur Spalte "Bindings" hinzugefügt.

Binding löschen

Das ausgewählte Binding wird im Hauptfenster aus der Spalte "Bindings" gelöscht.

Child anhängen

Ein neues Extensibility-Element wird an eine Input- oder Output Message angehängt. Falls ein Menüpunkt an der ausgewählten Stelle nicht angehängt werden darf, ist dieser ausgegraut. Weitere Informationen zu Extensibility-Objekten finden Sie in der W3C WSDL Spezifikation.

Folgende Extensibility-Objekte stehen zur Verfügung:

- soap:body
- soap:header
- soap:headerfault
- soap:fault
- mime:content
- mime:multipartrelated
- mime:part
- mime:mimeXml

- http:urlencoded
- http:urlreplacement

Extensibility löschen

Löscht das markierte Extensibility-Objekt.

28.14.1.5 Service

Service einfügen

Ein neuer Service wird im Hauptfenster in der Spalte "Services" hinzugefügt.

Service löschen

Der ausgewählte Service wird im Hauptfenster aus der Spalte "Services" gelöscht.

Port einfügen

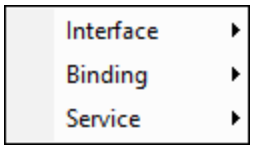
Ein neuer Port wird im Hauptfenster in der Spalte "Services" zum ausgewählten Service hinzugefügt.

Port löschen

Der ausgewählte Port wird aus dem ausgewählten Service gelöscht.

28.14.2 WSDL 2.0-Komponenten

Wenn Sie die Maus über den Menüeintrag **WSDL 2.0-Komponenten** platzieren, wird ein Untermenü eingeblendet (*Abbildung unten*), über das verschiedene Befehle zur Bearbeitung von WSDL 2.0-Komponenten ausgewählt werden können.

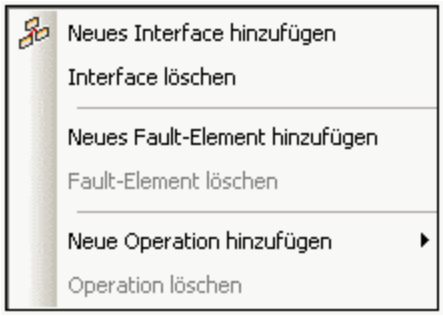


Jeder Befehl des Menüs WSDL 2.0 (*Abbildung oben*) enthält ein eigenes Untermenü, das Befehle im Zusammenhang mit dieser Komponente enthält. Die Befehle in den einzelnen Untermenüs werden in den Unterabschnitten dieses Abschnitts beschrieben.

© 2016-2022 Altova GmbH

28.14.2.1 Interface

Im Menü **Interface** (*Abbildung unten*) stehen die folgenden Befehle zur Verfügung.



Neues Interface hinzufügen

Fügt im Hauptfenster ein neues Interface-Kästchen zur Spalte "Interfaces" hinzu. Der Standardname des Interface erscheint im Interface-Kästchen markiert, sodass Sie ihn dort direkt bearbeiten können.

Interface löschen

Löscht das ausgewählte Interface.

Neues Fault-Element hinzufügen

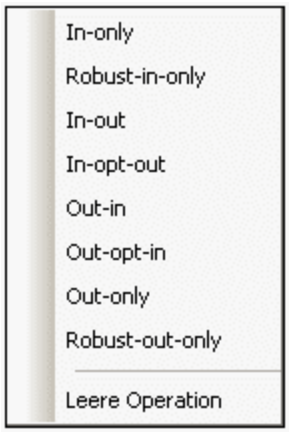
Fügt ein neues `fault` Element zum ausgewählten Interface hinzu. Der Standardname des `fault` Elements erscheint im Interface-Kästchen markiert, sodass Sie ihn dort direkt bearbeiten können.

Fault-Element löschen

Löscht das ausgewählte Fault-Element.

Neue Operation hinzufügen

Fügt ein neues `operation` Element zum ausgewählten Interface hinzu. Die Art der hinzuzufügenden Operation können Sie im Popout-Menü (*Abbildung unten*) auswählen. Es kann sich dabei um eine der Operationsarten handeln, die Sie in der Abbildung unten sehen.



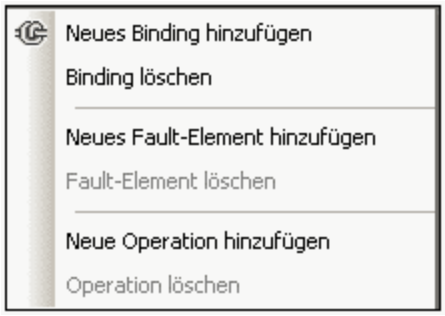
Der Standardname des `operation` Elements erscheint im Interface-Kästchen markiert, sodass Sie ihn dort direkt bearbeiten können.

Operation löschen

Löscht die ausgewählte Operation.

28.14.2.2 Binding

Im Menü **Binding** stehen die folgenden Befehle zur Verfügung (*Abbildung unten*).



Neues Binding hinzufügen

Fügt im Hauptfenster ein neues Binding-Kästchen zur Spalte "Bindings" hinzu. Der Standardname des Binding erscheint im Binding-Kästchen markiert, sodass Sie ihn dort direkt bearbeiten können.

Binding löschen

Löscht das ausgewählte Binding.

© 2016-2022 Altova GmbH

Neues Fault-Element hinzufügen

Fügt ein neues `fault` Element zum ausgewählten Binding hinzu. Ein `fault` Element in einem Binding enthält ein `ref` Attribut, das ein in einem Interface deklariertes Fault-Element referenziert. Das zu referenzierende Interface-Fault-Element im neu erstellten Fault-Element im Binding kann über eine Auswahlliste des neu erstellen Fault-Elements ausgewählt werden.

Fault-Element löschen

Löscht das ausgewählte Fault-Element.

Neue Operation hinzufügen

Fügt ein neues `operation` Element zum ausgewählten Binding hinzu. Ein `operation` Element in einem Binding enthält ein `ref` Attribut, das eine in einem Interface deklarierte Operation referenziert. Die zu referenzierende Interface-Operation in der neu erstellten Operation im Binding kann über eine Auswahlliste der neu erstellen Binding-Operation ausgewählt werden.

Operation löschen

Löscht die ausgewählte Operation.

28.14.2.3 Service

Im Menü **Service** stehen die folgenden Befehle zur Verfügung (*Abbildung unten*).



Neuen Service hinzufügen

Fügt im Hauptfenster ein neues Service-Kästchen zur Spalte "Services" hinzu. Der Standardname des Service erscheint im Service-Kästchen markiert, sodass Sie ihn dort direkt bearbeiten können. Die Interface-Referenz kann über die Auswahlliste für die *Interface*-Eigenschaft ausgewählt werden.

Service löschen

Löscht den ausgewählten Service.

Neuen Endpoint hinzufügen

Fügt ein neues `endpoint` Element zum ausgewählten Service hinzu. Der Standardname des Endpoint-Elements erscheint im Service-Kästchen markiert, sodass Sie ihn dort direkt bearbeiten können. Die Binding-Referenz kann über die Auswahlliste für die *Binding*-Eigenschaft ausgewählt werden. Die Adresse des Endpoint-Elements muss im Feld für die *Address*-Eigenschaft eingegeben werden.

Endpoint löschen

Löscht das ausgewählte Endpoint-Element.

28.14.3 Types, Diagramm speichern

Das Menü **Types** hat ein Untermenü, das die folgenden Befehle enthält. Diese sind weiter unten beschrieben.

- **Neues Schema**
- **Schema einbetten**
- **Schema(s) extrahieren**
- **Schema(s) in Schema-Ansicht bearbeiten**

Mit dem Befehl **Diagramm speichern** wird das Design-Diagramm als PNG-Datei gespeichert.

Types | Neues Schema

Diese Option ist dann verfügbar, wenn die WSDL Datei kein Schema-Element enthält. Bitte beachten Sie, dass bei Aufrufen der Menüoption **Datei | Neu** ein Schemaelement im Skelett der WSDL Datei inkludiert ist. In diesem Fall ist dieser Menüpunkt nicht verfügbar (siehe unten).

```
<types>
   <xs:schema/>
</types>
```

Types | Schema einbetten

Mit diesem Befehl wird ein "Öffnen"-Dialogfeld aufgerufen, in dem Sie zu der Schemadatei navigieren können, die Sie einbetten möchten. Nachdem Sie im Dialogfeld auf **OK** geklickt haben, wird das Schema innerhalb des Elements `types` als Inline-Schema erstellt. Wenn das ausgewählte Schema bereits importiert wurde, werden Sie gefragt, ob das bereits importierte Schema eingebettet werden soll. Wenn dies der Fall ist, wird es in ein Inline-Schema innerhalb des Elements `types` konvertiert.

Types | Schema(s) extrahieren

Bei Auswahl dieses Befehls wird jedes der (inline innerhalb des Elements `types` definierten) eingebetteten Schemas in der Schema-Ansicht als temporäre Datei geöffnet und für jede Datei wird ein "Speichern unter" Dialogfeld angezeigt. Wenn Sie eine Schema-Datei speichern, wird das Schema aus der WSDL-Datei extrahiert, unter dem angegebenen Pfad gespeichert und anschließend in die WSDL-Datei importiert. Es handelt sich nun nicht mehr um ein eingebettetes Schema, sondern um ein externes, importiertes Schema.

© 2016-2022 Altova GmbH

Types | Schema(s) in Schema-Ansicht bearbeiten

Öffnet eine Schemaskelett-Datei, falls die WSDL Datei keine Referenz auf ein spezifisches Schema enthält. Dies ist der Fall, wenn Sie die Menüoption **Datei | Neu** verwendet haben. Wenn eine Referenz zu einem spezifischen Schema existiert, wird das Schema in der eingebetteten Schema-Ansicht des graphischen WSDL Editors angezeigt.

28.14.4 Dokumentation generieren

Mit dem Befehl **Dokumentation generieren** wird die Dokumentation für die aktuelle WSDL Datei generiert. Die Ausgabe der Dokumentation kann als HTML-, MS Word, RTF- oder PDF.Datei erfolgen. Diese Dokumentation kann nach Wunsch geändert und verwendet werden; Sie benötigen dafür keine Genehmigung von Altova. Die Dokumentation wird für die Komponenten generiert, die Sie im Dialogfeld "WSDL-Dokumentation generieren" auswählen (Aufruf des Dialogfelds mit dem Befehl "Dokumentation generieren"). Miteinander in Zusammenhang stehende Komponenten werden in der Ausgabe auf dem Bildschirm durch Hyperlinks verbunden, um die Navigation zu vereinfachen. Beachten Sie, dass WSDL-Dokumentation auch für **importierte WSDL- und XML-Schema-Dateien** generiert werden kann. Die verschiedenen Optionen für die Generierung von Dokumentation werden im Abschnitt "Dokumentationsoptionen[1517]" beschrieben.

Anmerkung: Um Dokumentation im MS Word-Format generieren zu können, muss MS Word (Version 2000 oder höher) installiert sein.

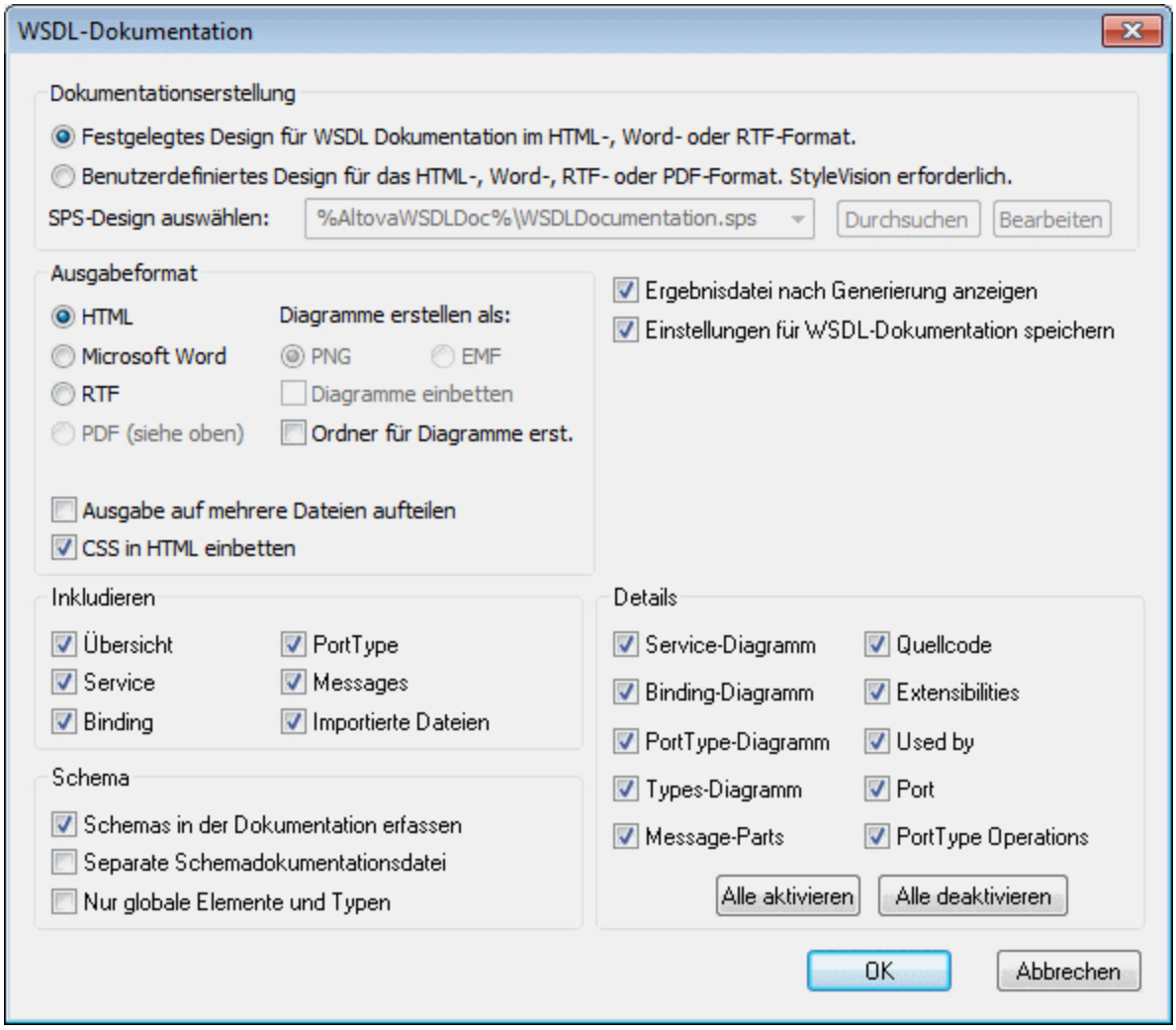
Sie können für das generierte Dokument entweder das festgelegte Standard-Design von XMLSpy verwenden oder für das Design ein StyleVision SPS verwenden. Die Verwendung eines StyleVision SPS hat den Vorteil, dass Sie das Design der generierten Dokumentation anpassen sowie als zusätzliches Ausgabeformat PDF generieren können. Wie man mit einem SPS arbeitet, wird in im Abschnitt "Benutzerdefiniertes Design[1393]" erklärt.

Anmerkung: Um ein SPS zum Generieren von WSDL-Dokumentation zu verwenden muss StyleVision auf Ihrem Computer installiert sein.

28.14.4.1 Dokumentationsoptionen

Mit dem Befehl **WSDL | Dokumentation generieren** wird das Dialogfeld WSDL-Dokumentation geöffnet (*siehe Abbildung unten*), in dem Sie die Optionen für die Dokumentation auswählen können.

Im Bereich "Dokumentationserstellung" des Dialogfelds können Sie wählen, ob Sie für die generierte Dokumentation das festgelegte XMLSpy-Design oder ein in einem StyleVision SPS erstelltes angepasstes Design verwenden möchten. Wählen Sie die gewünschte Option. Beachten Sie, dass die PDF-Ausgabe nur für mit einem StyleVision SPS erstellte Dokumentation und nicht für mit einem festgelegten Design generierte Dokumentation zur Verfügung steht. Wie Sie mit einem benutzerdefinierten Design arbeiten, wird im Abschnitt "Benutzerdefiniertes Design[1519]"[1393] beschrieben.



Die anderen Optionen im Dialogfeld WDSL-Dokumentation werden im Anschluss erklärt. Je nachdem ob ein WDSL 1.1 oder ein WSDL 2.0 Dokument aktiv ist, enthalten die Bereiche "Inkludieren" und "Details" unterschiedliche Elemente. Die obige Abbildung zeigt das Dialogfeld WSDL-Dokumentation für ein WSDL 1.1 Dokument.

- Das gewünschte Format wird im Bereich "Ausgabeformat" definiert: entweder HTML, Microsoft Word, RTF oder PDF. (Das Ausgabeformat PDF steht nur zur Verfügung wenn Sie ein StyleVision SPS zur Generierung der Dokumentation verwenden.) Wenn Sie auf OK klicken, werden Sie aufgefordert, den Namen der Ausgabedatei und den Ausgabepfad dafür anzugeben.
- Microsoft Word-Dokumente werden mit der Dateierweiterung `.doc` erstellt, wenn sie mit Hilfe eines festgelegten Designs generiert werden, und mit der Dateierweiterung `.docx`, wenn sie mit Hilfe eines StyleVision SPS generiert werden.
- Die Dokumentation kann entweder als eine einzige Datei generiert oder in mehrere Dateien aufgeteilt werden. Wenn mehrere Dateien generiert werden, entspricht jede Datei einer Komponente. Welche Komponenten in der Ausgabe enthalten sein sollen, wird mittels der Kotrollkästchen im Bereich "Inkludieren" festgelegt. In festgelegten Designs werden automatisch Links zwischen mehreren Dokumenten erstellt.

 © 2016-2022 Altova GmbH

- Für die HTML-Ausgabe können die CSS-Stildefinitionen entweder in einer separaten CSS-Datei gespeichert oder in der HTML-Datei (im Element `<head>`) eingebettet werden. Wenn eine separate CSS-Datei erstellt wird, erhält sie denselben Namen wie die HTML-Datei, jedoch mit der Erweiterung `.css`. Aktivieren oder deaktivieren Sie das Kontrollkästchen *CSS in HTML einbetten* um die gewünschte Option festzulegen.
- Die Option "*Diagramme einbetten*" ist für die Ausgabeoptionen MS Word, RTF und PDF aktiv. Wenn diese Option aktiviert ist, werden Diagramme entweder im PNG- oder im EMF-Format in die Ergebnisdatei eingebettet. Andernfalls werden die Diagramme als PNG- oder EMF-Dateien generiert und in der Ergebnisdatei über Objektlinks angezeigt.
- Wenn es sich bei der Ausgabedatei um eine HTML-Datei handelt, werden alle Diagramme als dokumentexterne PNG-Dateien generiert. Wenn das Kontrollkästchen *Ordner für Diagramme erstellen* aktiviert ist, wird im selben Ordner wie die HTML-Datei ein Ordner erstellt, in dem die PNG-Dateien gespeichert werden. Dieser Ordner erhält einen Namen im Format *`HTMLDateiname`*`_diagramme`. Wenn das Kontrollkästchen *Ordner für Diagramme erstellen* deaktiviert ist, werden die PNG-Dateien im selben Ordner wie die HTML-Datei gespeichert.
- Im Bereich **Inkludieren** können Sie auswählen, welche Datenelemente in der Dokumentation inkludiert werden sollen. Bei Auswahl der Option *Übersicht* werden alle Komponenten nach Komponententyp geordnet am Anfang der Datei aufgelistet. Wenn die Option *Importierte Dateien* (WSDL 1.1) oder *Importierte/Inkludierte Dateien* (WSDL 2.0) aktiviert ist, werden Komponenten in importierten Dateien (sowie im Fall von WSDL 2.0 inkludierte Dateien) in die Schemadokumentation inkludiert.
- Im Bereich "Schema" können Sie auswählen, ob Schemas in der Datei in die Dokumentation inkludiert werden sollen oder nicht. Soll die Dokumentation auch Schemas erfassen, haben Sie die Wahl zwischen folgenden Optionen: (i) ob die Schema-Dokumentation in einer eigenen Datei oder in der Hauptdokumentation ausgegeben werden soll und (ii) ob das gesamte Schema oder nur globale Elemente, SimpleTypes und ComplexTypes dokumentiert werden sollen.
- Im Bereich **Details** werden die Einzelheiten aufgelistet, die für die einzelnen Komponenten inkludiert werden können. Wählen Sie die gewünschten Informationen aus. Über die Schaltflächen **Alle aktivieren** und **Alle deaktivieren** können Sie alle Optionen im Bereich einfach aktivieren bzw. deaktivieren.
- Die Option "Ergebnisdatei nach der Generierung anzeigen" ist für alle Ausgabeoptionen aktiv. Wenn diese Option aktiviert ist, wird die erzeugte Datei in der Browser-Ansicht (HTML-Ausgabe), MS Word (MS Word Ausgabe) und den Standardapplikationen für `.rtf`-Dateien (RTF-Ausgabe) und `.pdf`-Dateien (PDF-Ausgabe) angezeigt.

Parameterwerte

Wenn das StyleVision SPS einen oder mehrere Parameterdefinitionen enthält, wird beim Klicken auf **OK** ein Dialogfeld aufgerufen, in dem alle Parameter aufgelistet werden, die im SPS definiert sind. Sie können in diesem Dialogfeld Parameterwerte eingeben um die Standardwerte zu überschreiben, die im SPS zugeordnet worden sind.

28.14.4.2 Benutzerdefiniertes Design

Sie können anstatt des festgelegten Standart XMLSpy Designs ein angepasstes Design für die WSDL-Dokumentation erstellen. Das angepasste Design wird in einem StyleVision SPS erstellt, das eine Design-Vorlage für das Ausgabedokument darstellt.

Erstellen des SPS

Ein StyleVision Power Stylesheet (oder SPS) wird mit Hilfe des StyleVision Produkts von Altova erstellt. Einem SPS zur Generierung von WSDL-Dokumentation muss ein XML-Schema zugrunde liegen, das die Struktur der WSDL-Dokumentation festlegt. Zwei Schemas, eines für WSDL 1.1 und das zweite für WSDL 2.0 werden mit Ihrem XMLSpy Paket mitgeliefert. Diese sind `WSDLDocumentation.xsd` bzw. `WDSL20Documentation.xsd` und befinden sich in den folgenden Ordnern des (Eigene Dateien)-Verzeichnisses[34]:

- `C:\Dokumente und Einstellungen\<Benutzername>\Eigene Dateien\Altova\XMLSpy2023\Dokumentation\WDSL`.
- `C:\Dokumente und Einstellungen\<Benutzername>\Eigene Dateien\Altova\XMLSpy2023\Dokumentation\WDSL20`.

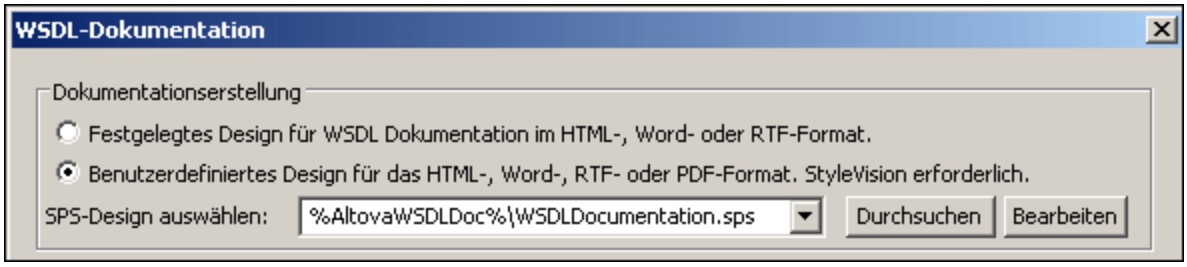
Beim Erstellen des SPS Designs in StyleVision werden Nodes aus dem Schema in die Design-Vorlage eingefügt und Formatvorlagen und Eigenschaften zugewiesen. Zusätzliche Komponenten wie Links, Tabellen und Bilder können ebenfalls zum SPS hinzugefügt werden. Auf diese Weise kann die gesamte Ausgabe im SPS gestaltet werden. Wie man ein SPS Design in StyleVision erstellt, wird im Benutzerhandbuch von StyleVision im Detail beschrieben.

Die Verwendung eines SPS zur Generierung von WSDL-Dokumentation hat den Vorteil, dass Sie die vollständige Kontrolle über das SPS Design haben. Beachten Sie auch, dass die PDF-Ausgabe der WDSL-Dokumentation nur dann verfügbar ist, wenn ein benutzerdefiniertes SPS verwendet wird; PDF-Ausgabe steht nicht zur Verfügung, wenn das festgelegte XMLSpy Design verwendet wird.

Festlegen des für die WSDL-Dokumentation verwendete SPS

Nachdem ein SPS erstellt worden ist, kann es zum Generieren von WDSL-Dokumentation verwendet werden. Das SPS, das Sie für die Generierung der WDSL-Dokumentation verwenden möchten wird im Dialogfeld WSDL-Dokumentation (aufgerufen über den Befehl **WSDL | Dokumentation generieren...**) ausgewählt. Wählen Sie im Bereich "Dokumentationserstellung" dieses Dialogfelds (*siehe Abbildung unten*) die Optionsschaltfläche *Beutzerdefiniertes Design für das HTML-, Word-, RTF- oder PDF-Format* aus. Sie können dann auf die Schaltfläche **Durchsuchen** klicken und das gewünschte SPS im Dateisystem auswählen. Klicken Sie im Dialogfeld auf **OK** und geben Sie im Dialogfeld Speichern, das erscheint, das Verzeichnis und den Namen der Ausgabedatei ein.

Anmerkung: Der Pfad zum Schema, auf dem die SPS-Datei basiert, (`WSDLDocumentation.xsd` or `WSDL20Documentation.xsd` (siehe oben) muss im SPS korrekt angegeben sein:



Zwei editierbare SPS Designs, je eines für WSDL 1.1 und WSDL 2.0, werden mit XMLSpy geliefert. Sie befinden sich im Unterverzeichnis `WDSL` bzw. `WSDL20` des (Eigene Dateien)-Verzeichnisses[34] `C:\Dokumente und Einstellungen\<Benutzername>\Eigene Dateien\Altova\XMLSpy2011\Dokumentation\`. Sie heißen:

- `WSDL\WSDLDocumentation.sps`
- `WSDL20\WSDL20Documentation.sps`

Diese Dateien sowie alle anderen SPS-Dateien, die Sie kürzlich aufgerufen haben, stehen in der Dropdown-Liste der Option Benutzerdefiniertes Design zur Verfügung (*siehe Abbildung oben*).

Durch Klicken auf die Schaltfläche **Bearbeiten** im Bereich "Dokumentationserstellung" wird StyleVision gestartet und das ausgewählte SPS in einem StyleVision Fenster geöffnet. Um eine Vorschau auf das resultierende Dokument in StyleVision zu erhalten, benötigen Sie eine XML-Arbeitsdatei. Zu diesem Zweck werden die XML-Beispieldateien namens `TimeService.xml` bzw. `TimeService20.xml` mit Ihrer Applikation mitgeliefert; sie befinden sich im Verzeichnis (Eigene) Dateien[34]:

```
C:\Dokumente und Einstellungen\<Benutzername>\Eigene
Dateien\Altova\XMLSpy2023\Dokumentation\WSDL(20)\Beispieldaten
```

Anmerkung: Um ein SPS zum Generieren von WSDL-Dokumentation verwenden zu können, muss StyleVision auf Ihrem Computer installiert sein.

28.14.5 WSDL-Dokument neu parsen

Parst das Dokument neu. Dies ist in einigen Situationen erforderlich, z.B. wenn ein mit dem WSDL-Dokument verknüpftes Schema geändert wurde. Wenn Sie das WSDL-Dokument neu parsen, wird es mit den Informationen aus dem geänderten Schema-Dokument aktualisiert.

28.14.6 In WSDL 2.0 konvertieren

Der Befehl **In WSDL 2.0 konvertieren** ist nur aktiv, wenn ein WSDL 1.1-Dokument in der WSDL-Ansicht aktiv ist. Er generiert anhand eines WSDL 1.1-Dokuments ein WSDL 2.0-Dokument. Bei Auswahl dieses Befehls erscheint ein "Datei speichern"-Dialogfeld, in dem Sie den Pfad der von XMLSpy generierten WSDL 2.0-Datei angeben können.

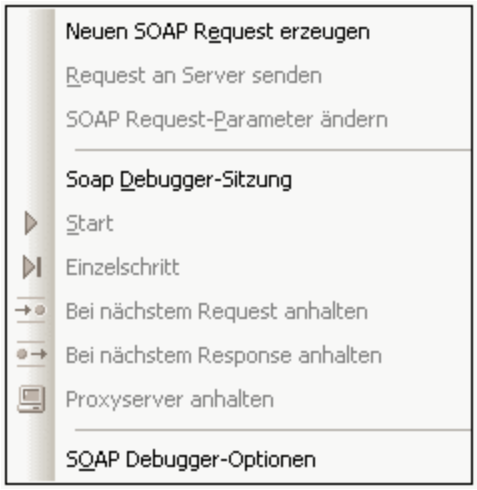
Nachdem Sie im Dialogfeld "Datei speichern" auf **OK** geklickt haben, wird ein WSDL 2.0-Dokument generiert, unter dem definierten Pfad gespeichert und in der WSDL-Ansicht auf einem neuen Register geöffnet. Die Datei kann dann wie jedes andere WSDL 2.0-Dokument nach Wunsch bearbeitet werden.

28.14.7 WSDL-Programmcode mit MapForce generieren

Mit diesem Befehl wird Altova MapForce gestartet, wenn die Applikation installiert ist. In MapForce können Sie ein Schema auf eine andere DTD, ein XML-Schema oder eine Datenbank mappen, um ein XML-Dokument zu generieren und Programmcode anhand der WSDL-Datei zu generieren.

28.15 Menü "Soap"

XMLSpy unterstützt die SOAP Versionen 1.1 und 1.2 und die WSDL Versionen 1.1 und 2.0.



Im Abschnitt SOAP: How To[802], der auf die Menübeschreibungen folgt, wird anhand des **nanonull.com timeservice Server**, der Ihnen von Altova zur Verfügung gestellt wird, die Funktionsweise des SOAP Debuggers beschrieben. Bitte verwenden Sie diesen Service, um den Debugger zu testen. Der AirportWeather Webservice, der auf den folgenden Seiten beschrieben wird, steht möglicherweise nicht immer zur Verfügung.

Verwenden Sie die SOAP Funktionalitäten:

- zum Testen Ihrer Webservices, ohne dass Client-Applikationen implementiert werden müssen
- zum schnellen Testen von Webservices von anderen Anbietern

Weitere Informationen

Weitere Informationen zu den Spezifikationen erhalten Sie unter:

SOAP http://www.w3.org/TR/SOAP/
WSDL http://www.w3.org/TR/wsdl

28.15.1 Neuen SOAP Request erzeugen

Mit diesem Befehl erzeugen Sie ein neues SOAP Request-Dokument. Dies erfordert die folgenden Schritte:

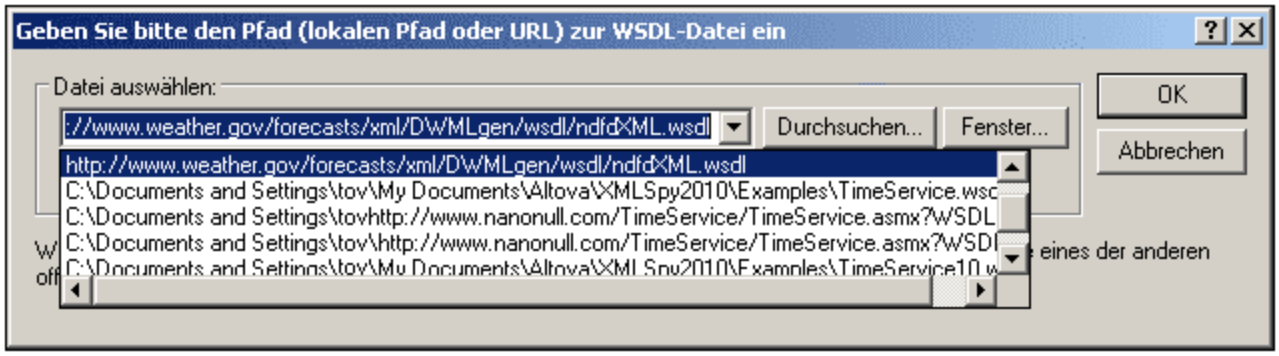
1. Geben Sie den Pfad zur WSDL-Datei an und stellen Sie eine Verbindung zum SOAP-Server her[1523].
2. Der Server gibt eine Liste von Operationen zurück. Wählen Sie die gewünschte SOAP-Operation aus[1523].
3. Der Server sendet ein SOAP Request-Formular im XML-Format zurück. Definieren Sie das SOAP Request-Formular[1524].

© 2016-2022 Altova GmbH

Im Folgenden wird die Vorgangsweise anhand der Erstellung eines SOAP Request für den SOAP-Service "US National Digital Forecast Database (NDFD)" gezeigt (http://www.nws.noaa.gov/xml/).

Herstellen einer Verbindung zum SOAP-Server

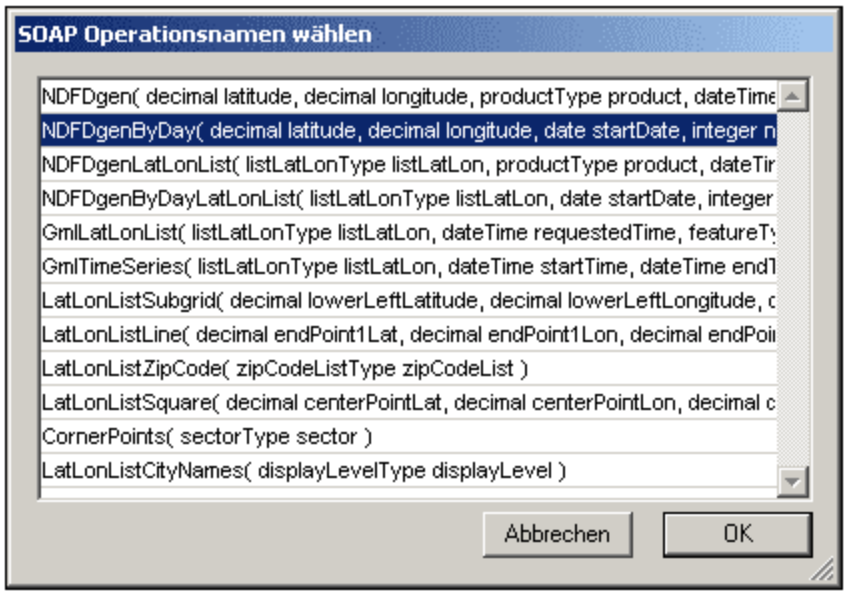
Die Verbindung wird über eine WSDL-Datei hergestellt. In unserem Beispiel lautet die URI der WSDL-Datei:`http://www.weather.gov/forecasts/xml/DWMLgen/wsdl/ndfdXML.wsdl`. Klicken Sie zum Herstellen der Verbindung auf den Befehl "Neuen SOAP Request erzeugen" und geben Sie im Dialogfeld, das angezeigt wird, die Datei-URI ein (*Abbildung unten*).



Klicken Sie zur Bestätigung auf **OK**.

Auswahl der gewünschten SOAP-Operation

Der Server sendet nun eine Liste von Operationen, die in einem Dialogfeld angezeigt werden (*Abbildung unten*).



Wählen Sie eine Operation aus und klicken Sie auf **OK**. Wir haben die Operation `NDFgenByDay` ausgewählt.

Definieren des SOAP Request

Der Server sendet daraufhin eine XML-Datei, die in der Textansicht von XMLSpy angezeigt wird. Für die ausgewählte Operation erhalten wir die folgende XML-Datei.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:m0="http://www.weather.gov/forecasts/xml/DWMLgen/schema/DWML.xsd">
   <SOAP-ENV:Body>
      <m:NDFDgenByDay
xmlns:m="http://www.weather.gov/forecasts/xml/DWMLgen/wsdl/ndfdXML.wsdl"
       SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
         <latitude xsi:type="xsd:decimal">0.0</latitude>
         <longitude xsi:type="xsd:decimal">0.0</longitude>
         <startDate xsi:type="xsd:date">1967-08-13</startDate>
         <numDays xsi:type="xsd:integer">0</numDays>
         <format xsi:type="m0:formatType">String</format>
      </m:NDFDgenByDay>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

We filled in the parameters as required by the XML (*in bold red below; fill in a start date that is the current date or one within the next week*):

```
<SOAP-ENV:Body>
   <m:NDFDgenByDay
xmlns:m="http://www.weather.gov/forecasts/xml/DWMLgen/wsdl/ndfdXML.wsdl"
    SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
         <latitude xsi:type="xsd:decimal">45</latitude>
         <longitude xsi:type="xsd:decimal">-90</longitude>
         <startDate xsi:type="xsd:date">2019-12-10</startDate>
         <numDays xsi:type="xsd:integer">1</numDays>
         <format xsi:type="m0:formatType">24 hourly</format>
   </m:NDFDgenByDay>
</SOAP-ENV:Body>
```

Damit ist die **Definition** dieses SOAP Request abgeschlossen. Im nächsten Schritt senden wir den Request[1524].

28.15.2 Request an Server senden

Eine Anleitung, wie Sie einen SOAP-Request erzeugen, finden Sie im vorhergehenden Kapitel Neuen SOAP Request erzeugen[1522], Nachdem Sie den SOAP-Request, bei dem es sich um ein XML-Dokument handelt, generiert haben, machen Sie ihn zu aktiven Dokument. Klicken Sie anschließend auf diesen Befehl, um den SOAP-Request an den SOAP-Server zu senden.

Nachdem der SOAP Request abgesendet wurde, geht eine Antwort (Response) vom SOAP Server ein. Diese Antwort ist ein XML-Dokument, das in der Textansicht von XMLSpy angezeigt wird. Als Beispiel dafür sehen

© 2016-2022 Altova GmbH

Sie unten in der Abbildung ein XML-Dokument, das als Antwort auf den SOAP Request, den wir im Abschnitt Neuen SOAP Request erzeugen[1522] definiert haben, zurückgesendet wurde.



Speichern und erneute Verwendung eines SOAP Request:

In XMLSpy können Sie einen SOAP Request speichern und zu einem späteren Zeitpunkt erneut senden. Dies geschieht folgendermaßen:

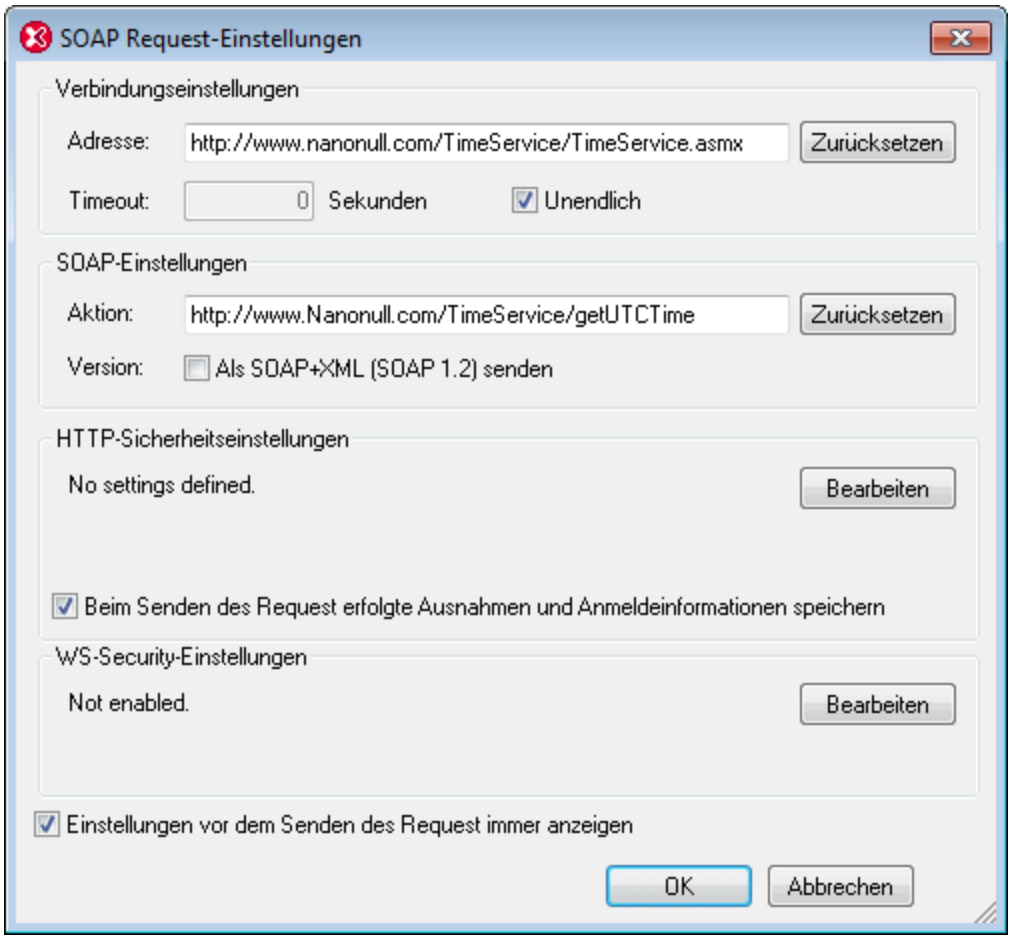
1. Speichern Sie das SOAP Request-XML-Dokument (**Datei | Speichern unter...**)
2. Schließen Sie die SOAP Request-Datei.
3. Öffnen Sie das SOAP Request XML-Dokument erneut und wählen Sie die Menüoption **SOAP | Request an Server senden.** (Jede XML-Datei kann als SOAP Request-Dokument verwendet werden).

28.15.3 SOAP Request-Einstellungen

Mit diesem Befehl wird das Dialogfeld "SOAP Request-Einstellungen" (*Abbildung unten*) geöffnet, in dem Sie die verschiedenen Einstellungen des SOAP Request definieren können. Diese Einstellungen sind im Folgenden beschrieben.

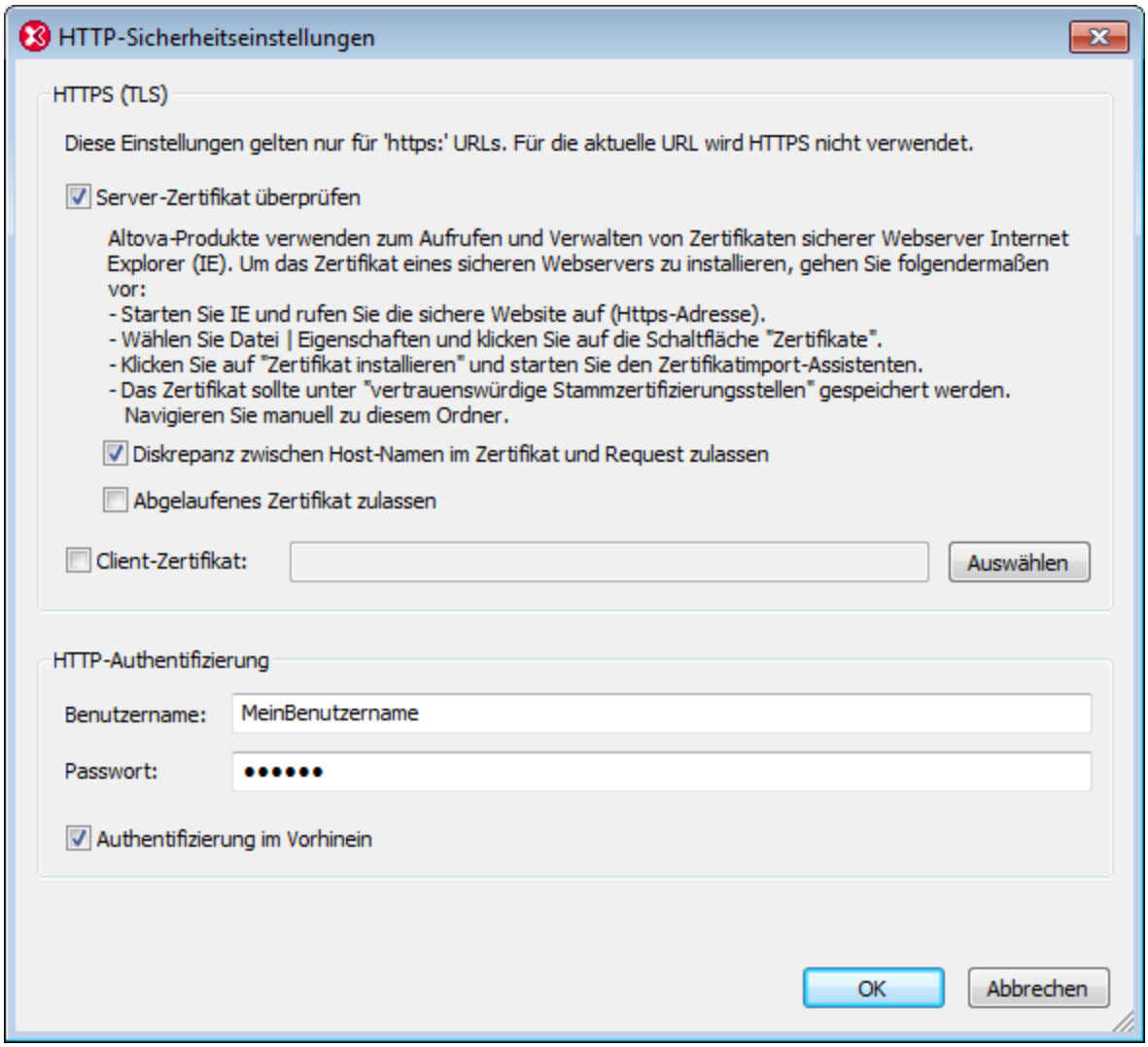
1. Machen Sie das SOAP Request-Dokument zum aktiven.

2. Wählen Sie die Menüoption **Soap | SOAP Request-Einstellungen**. Daraufhin wird das Dialogfeld "SOAP Request-Einstellungen" aufgerufen (*Abbildung unten*).



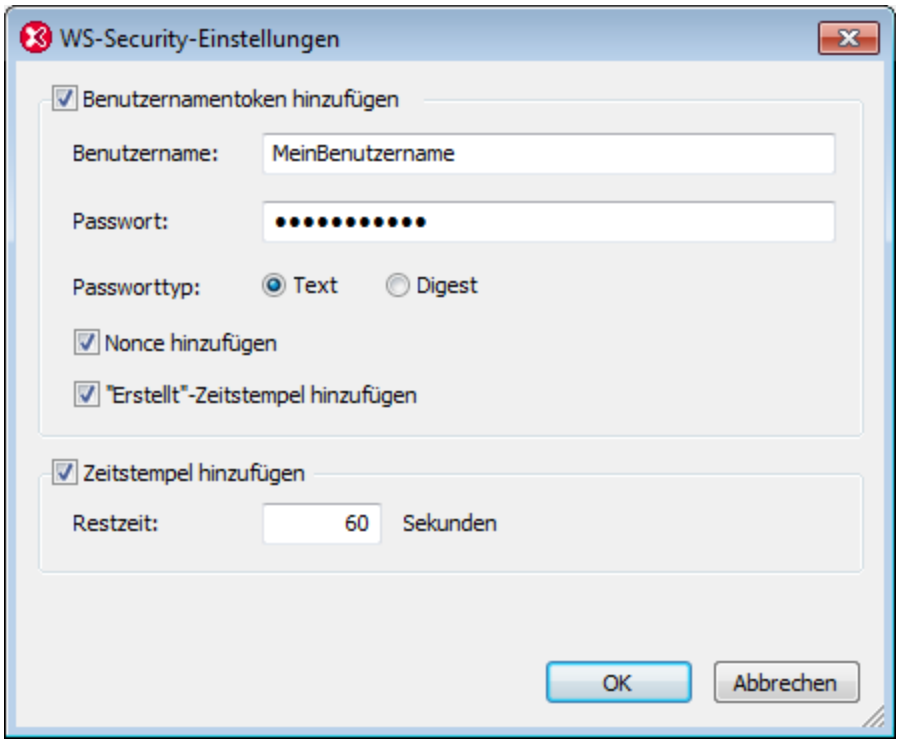
3. Geben Sie in das Feld *Adresse* den gewünschten Connection Endpoint ein. Wenn der SOAP Request in XMLSpy anhand einer WSDL-Datei erstellt wurde, so wird als Wert des Felds *Adresse* der Pfad zu dem in der WSDL-Datei ausgewählten Endpoint verwendet. Sie können einen Timeout-Wert für die Herstellung der Verbindung in Sekunden definieren. Um keinen Timeout festzulegen, aktivieren Sie das Kontrollkästchen *Unendlich.*
4. Geben Sie in das Feld *Aktion* die auszuführende SOAP-Aktion ein. Um den Request als SOAP 1.2 zu senden, aktivieren Sie das Kontrollkästchen *Als SOAP+XML (SOAP 1.2) senden*. Wenn der SOAP Request in XMLSpy anhand einer WSDL-Datei erstellt wurde, so wird die SOAP-Aktion vom Erweiterungselement unter der entsprechenden SOAP Binding-Operation in der WSDL-Datei empfangen. In diesem Fall wird die SOAP-Version ebenfalls im Vorhinein anhand der WSDL-Datei ausgewählt. (Die SOAP-Version wirkt sich auf den Wert des HTTP Header-Inhaltstyps aus: `text/xml` oder `application/soap+xml`.) Klicken Sie auf **Zurücksetzen**, um die SOAP-Aktion aus der WSDL-Datei abzurufen.
5. Im Bereich HTTP-Sicherheitseinstellungen sehen Sie eine Zusammenfassung der Sicherheitseinstellungslisten. Wenn die Option *Beim Senden des Request erfolgte Ausnahmen...speichern* aktiviert ist, werden alle Einstellungen beim Senden des Request gespeichert und können für den nächsten Request wiederverwendet werden. Klicken Sie auf die Schaltfläche **Bearbeiten**, um das Dialogfeld "HTTP-Sicherheitseinstellungen" (*Abbildung unten*) aufzurufen. Eine Anleitung zum Installieren von Server-Zertifikaten finden Sie weiter unten.

 © 2016-2022 Altova GmbH



Wenn Sie eine ***Diskrepanz zwischen Host-Namen*** (im Server-Zertifikat und in der tatsächlich verwendeten Adresse) oder ein ***abgelaufenes Server-Zertifikat*** zulassen möchten, so aktivieren Sie diese Optionen im Dialogfeld. Wenn für den Server ein ***Client-Zertifikat*** benötigt wird, können Sie den Pfad zum Client-Zertifikat definieren. Wenn eine Authentifizierung vom Server verlangt wird, definieren Sie den ***Benutzernamen*** und das ***Passwort*** für die Standard-Authentifizierung. Wenn der erste Client-Request an den Server die erforderlichen Authentifizierungsinformationen enthält, so wird dies als ***Authentifizierung im Vorhinein*** bezeichnet. Wenn dies vom Server verlangt wird, aktivieren Sie die Option *Authentifizierung im Vorhinein*. Lassen Sie diese Option andernfalls deaktiviert.

6. Zusätzlich zu den Sicherheitseinstellungen auf Transport-Ebene (HTTP-Sicherheitseinstellungen) können Sie auch Webservice-Sicherheitseinstellungen definieren, falls dies vom Webservice verlangt wird. Klicken Sie auf die **Bearbeiten**-Schaltfläche des Bereichs WS-Security-Einstellungen, um das gleichnamige Dialogfeld aufzurufen (*Abbildung unten*). Die Sicherheitsinformationen enthalten den Benutzernamen, das Passwort, einen automatisch generierten Nonce-Code-String und einen Zeitstempel. Sie können auch den Gültigkeitszeitraum der Sicherheitsinformationen definieren (*Zeitstempel hinzufügen*). Im Dialogfeld wird ein XML-Fragment erstellt, das die Sicherheitsinformationen enthält und dieses Fragment in den SOAP Request einbettet. *Siehe Codefragment unten*.



7. Klicken Sie auf **OK**, wenn Sie damit fertig sind.

Informationen zu vertrauenswürdigen Zertifikaten

Altova Produkte verwenden zum Aufrufen und Verwalten vertrauenswürdiger Zertifikate sicherer Webserver Internet Explorer (IE). Wenn Sie das Zertifikat eines Webservers in IE installieren, greift IE auf den Webserver zu, ohne eine Warnmeldung auszugeben oder den Vorgang abzubrechen. Um das Zertifikat eines sicheren Webservers zu installieren, gehen Sie folgendermaßen vor:

- Öffnen Sie die sichere Website in Internet Explorer 8.
- Wählen Sie den Befehl **Datei | Eigenschaften** und klicken Sie auf die Schaltfläche "Zertifikate".
- Klicken Sie auf **Zertifikat installieren** und starten Sie den Assistenten zum Import eines Zertifikats. (Dieser Assistent kann auch über **Extras | Internet Optionen| Inhalt | Zertifikate | Importieren** aufgerufen werden.)
- Das Zertifikat sollte in den Speicher vertrauenswürdiger Stammzertifizierungsstellen platziert werden, zu dem Sie manuell navigieren können.
- Wenn Sie die einzelnen Schritte im Assistenten fertig gestellt haben, schließen Sie die Dialogfelder "Zertifikate" bzw. "Eigenschaften" durch Klicken auf OK. Eventuell müssen Sie Internet Explorer neu starten.

Anmerkung: Ändern Sie die SOAP Action-Einstellungen nur, wenn Sie Zugriff auf alle SOAP-Methoden und ihre entsprechenden SOAP-Aktionen haben.

Webservice-Sicherheitsinformationen

Für einige Webservices ist eine Benutzerauthentifizierung erforderlich. (Die Webservice-Sicherheitsebene ist eine zusätzliche Sicherheitsebene zu der durch den Server implementierten HTTP-Sicherheitsebene). Die Webservice-Authentifizierungsinformationen werden im SOAP Request als XML-Fragment mit der unten

 © 2016-2022 Altova GmbH

aufgelisteten Struktur gespeichert. Dieses XML-Fragment im SOAP Request wird automatisch anhand der Webservice-Authentifizierungsinformationen, die Sie im Dialogfeld "WS-Security-Einstellungen" eingeben, generiert.

```
<wsse:Security xmlns:wsse="…" xmlns:wsu="…" SOAP-ENV:mustUnderstand="true">
   <wsse:UsernameToken>
      <wsse:Username>usr</wsse:Username>
      <wsse:Password Type="...#PasswordText">pwd</wsse:Password>
      <wsse:Nonce EncodingType="…#Base64Binary">UqrtD963797WBRgWiJPu2w==</wsse:Nonce>
      <wsu:Created>2014-11-17T16:08:07.016Z</wsu:Created>
   </wsse:UsernameToken>
   <wsu:Timestamp>
      <wsu:Created>2014-11-17T16:08:07.016Z</wsu:Created>
      <wsu:Expires>2014-11-17T16:09:07.016Z</wsu:Expires>
   </wsu:Timestamp>
</wsse:Security>
```

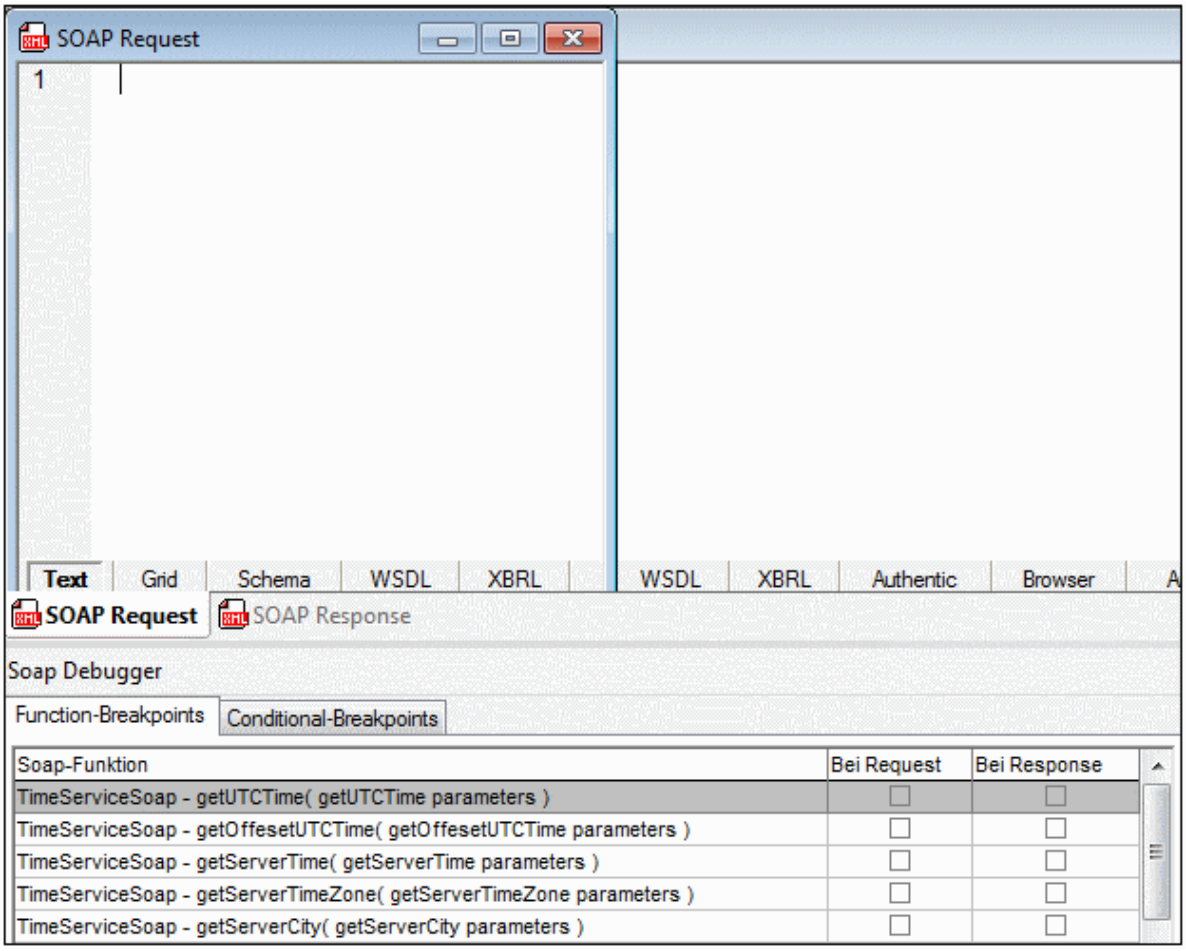
28.15.4 SOAP Debugger-Sitzung

Mit diesem Befehl starten Sie eine SOAP Debugger-Sitzung.

- Wenn Sie diesen Befehl ausführen, wird ein Dialogfeld angezeigt. Wählen Sie den Standort der WSDL-Datei (normalerweise eine URL). Sie können eine Datei auch über eine globale Ressource auswählen (klicken Sie auf das Symbol "Globale Ressourcen" und wählen Sie im angezeigten Dialogfeld eine globale Ressource aus) oder Sie können eine Datei in einem der offenen Fenster in XMLSpy auswählen.
- Definieren Sie im nächsten Dialogfeld die Quell- und Ziel-Ports, die für den Debugger Proxyserver und das Webservice benötigt werden.

  Der SOAP Debugger Proxyserver wird geöffnet, ist aber **noch nicht aktiv**. Klicken Sie auf eines der Symbole in der Symbolleiste. Der SOAP Debugger wird gestartet und wartet nun auf Ihre Befehle.

  Weitere Informationen erhalten Sie unter Soap Debugger 802 .



28.15.5 Starten



Mit diesem Befehl wird der SOAP Proxy Server gestartet. Die Prozedur wird solange in der WSDL-Datei fortgesetzt, bis ein Breakpoint erreicht wird. Das zugehörige SOAP Dokument wird dann in einem der SOAP Dokument-Fenster angezeigt.

28.15.6 Einzelschritt



© 2016-2022 Altova GmbH

Mit diesem Befehl können Sie sich schrittweise durch ein- und ausgehende SOAP Requests und Responses bewegen. Bei jedem Request und Response hält der Debugger automatisch an. Der Proxyserver wird auch gestartet, falls dieser inaktiv war.

28.15.7 Beim nächsten Request anhalten



Mit diesem Befehl hält der Debugger beim nächsten SOAP Request an und die Daten werden im SOAP Request Dokument-Fenster dargestellt. Sie können die Daten im Fenster bearbeiten bevor Sie diese an den Webservice senden.

28.15.8 Bei nächstem Response anhalten



Mit diesem Befehl hält der Debugger beim nächsten SOAP Response an und die Daten werden im SOAP Response Dokument Fenster dargestellt. Sie können die Daten im Fenster bearbeiten bevor Sie diese an den Client senden.

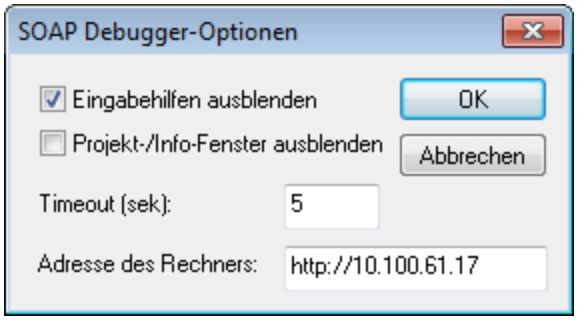
28.15.9 Proxyserver anhalten



Mit diesem Befehl halten Sie den Debugger Proxyserver an.

28.15.10 Soap Debugger-Optionen

Im Dialogfeld "SOAP-Debugger-Optionen" (*Abbildung unten*) können Sie die IP-Adresse des Rechners sowie andere unten aufgelistete Debugger-Optionen definieren. Rufen Sie das Dialogfeld mit dem Menübefehl **SOAP | SOAP Debugger-Optionen** auf.



- *Adresse des Rechners:* Die Adresse des Proxy-Servers, über den der Debugger ausgeführt wird. Der Debugger auf dem Proxy Server erhält Requests von Rechnern im Netzwerk und sendet diese an den Webservice. Da der Debugger in XMLSpy ausgeführt wird, dient der Rechner, auf dem XMLSpy installiert ist, auch als Proxy Server. Die IP-Adresse des Rechners wird automatisch ermittelt und in dieses Feld eingetragen. Sie müssen die IP-Adresse nur dann (als `http`-Adresse) in dieses Feld eingeben, wenn die IP-Adresse nicht automatisch ermittelt werden kann. Um die IP-Adresse Ihres Rechners herauszufinden, öffnen Sie ein Eingabeaufforderungsfenster, geben Sie den Befehl `ipconfig /all` ein und drücken Sie die **Eingabetaste**.
- *Timeout:* Dieser Wert ist die Zeitspanne, die der SOAP Debugger an einem Breakpoint stehen bleibt. Die Standardeinstellung ist 5 Sekunden.
- *Eingabehilfen ausblenden; Projekt-/Info-Fenster ausblenden:* Diese Optionen sind nützlich, um mehr Platz für das SOAP Debugger-Fenster zu haben.

© 2016-2022 Altova GmbH

28.16 Menü "XBRL"

Das Menü **XBRL**(*Abbildung unten*) enthält Befehle, die aktiv sind, wenn eine Taxonomie in der XBRL-Ansicht bearbeitet wird. Diese Befehle sind unten aufgelistet und in den Unterabschnitten dieses Abschnitts beschrieben.

- Arcroles[1533]: Definiert Arcroles
- Linkroles[1535]: Definiert Linkroles
- Namespaces-Präfixe[1537]: Verwaltet Taxonomie Namespaces
- Target Namespace definieren[1538] : Definiert und deklariert die Target Namespaces der Taxonomie.
- Parameterwerte[1538]: Zeigt Formula-Parameter und Tabellenparameter in einem Dialogfeld an, in dem sie bearbeitet werden können.
- Importieren/Referenzieren[1539]: Importiert eine XBRL-Taxonomie bzw. referenziert eine Linkbase.
- Formula-Komponente nach ID suchen[1541]: Sucht Formula-Komponenten auf Basis der vom Benutzer angegebenen ID.
- Dokumentation generieren[1541]: generiert Dokumentation zur aktuellen XBRL-Taxonomie.
- Ansichtseinstellungen[1545]: Definiert die Standardeinstellungen für die XBRL-Ansicht.
- XBRL mit MapForce anhand von DB, Excel, CSV generieren[1546]: Startet Altova MapForce, um eine XBRL-Instanzdatei zu generieren.
- XBRL mit StyleVision als HTML/PDF/Word-Dokument darstellen[1546]: Startet Altova StyleVision zur Erstellungen eines XBRL-Berichts.
- Formula ausführen[1546]: Führt Formeln (Formulas) und/oder Assertions anhand des mit dem aktiven XBRL-Instanzdokument verknüpften DTS aus.
- Tabelle generieren[1549]: Generiert XBRL-Tabellen anhand einer XBRL-Instanz.
- Duplikate ermitteln[1551]: Ermittelt doppelt vorhandene Facts in XBRL-Instanzen.
- XULE ausführen[1552]: Führt ein XULE-Dokument an einem XBRL-Instanzdokument aus.
- Inline XBRL transformieren[1553]: Generiert den Inline XBRL-Teil eines XHTML-Dokuments als XBRL.
- EDGAR am Server validieren[1554]: Validiert XBRL-Instanzen mittels EDGAR auf RaptorXML+XBRL Server.
- Verarbeitungsoptionen[1555]: Damit können Sie Optionen für die Verarbeitung von XBRL-Instanzen definieren.

Nähere Informationen zu XBRL finden Sie in den Abschnitten XBRL[829] und Bearbeitungsansichten | XBRL-Ansicht[266].

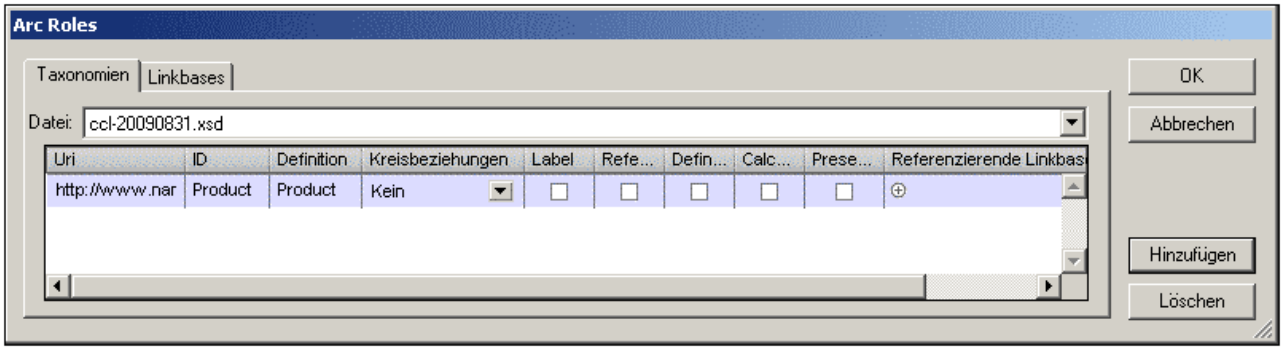
Informationen zum Konvertieren von OIM xBRL-Formaten finden Sie unter Menü "Konvertieren"[1470].

28.16.1 Arcroles

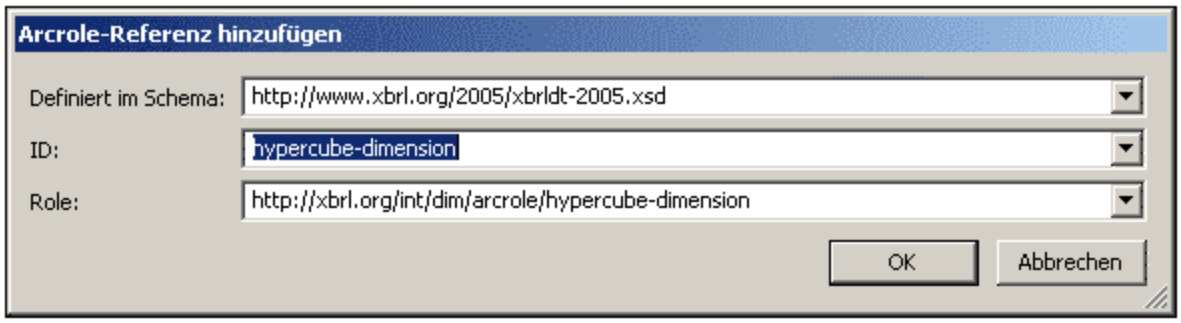
Mit dem Befehl **Arcroles** wird das Dialogfeld "Arc Roles" (*Abbildung unten*) aufgerufen, in dem Arcroles für eine Taxonomie erstellt werden können. Arcroles werden in der Concept Definitions-Datei im Element `appinfo` gespeichert. Sie definieren die Rolle eines Arc.

Auf dem Register **Taxonomien** des Dialogfelds "Arc Roles" (*Abbildung unten*) werden in der Auswahlliste nur Taxonomien angezeigt, die bearbeitet werden können oder die eine Arcrole oder eine Linkrole enthalten. Durch Klicken auf die Schaltfläche **Hinzufügen** können Sie eine Arcrole zu einer Taxonomie hinzufügen. Definieren Sie anschließend die URI, ID, Definition und Kreisbeziehungen der Arcrole. Um festzulegen, in welchen Beziehungsarten die Arcrole zur Verfügung stehen soll, aktivieren Sie die Kästchen der gewünschten

Beziehungsarten. Linkbases, die eine Arcrole referenzieren, können zur Spalte "Referenzierende Linkbase-Dateien" hinzugefügt werden.



Auf dem Register **Linkbases** sehen Sie eine andere Ansicht der Arcroles der Taxonomie. In dieser Ansicht können Sie Arcroles nach Linkbases hinzufügen und anzeigen (z.B. Calculation oder Presentation Linkbases). Wählen Sie in der Auswahlliste eine Linkbase aus und fügen Sie anschließend je nach Bedarf eine Arcrole hinzu oder löschen Sie eine. Wenn Sie auf die Schaltfläche **Hinzufügen** klicken, wird das Dialogfeld "Arcrole-Referenz hinzufügen" angezeigt (*Abbildung unten)*.

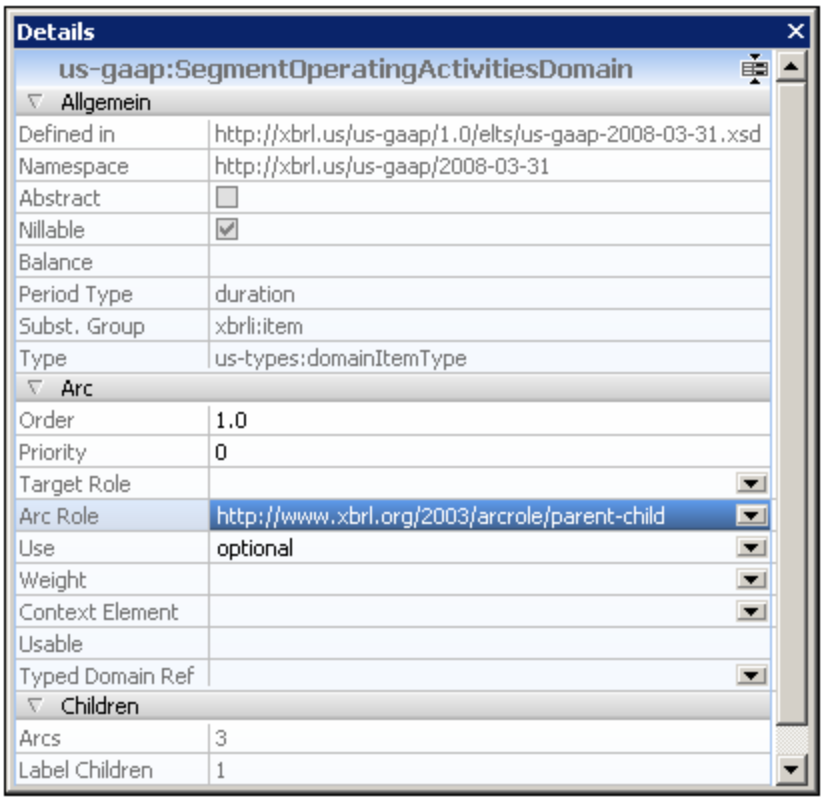


Bei den einzelnen Feldern in diesem Dialogfeld handelt es sich um Auswahllisten, über die Sie zwischen verfügbaren Optionen wählen können. Im Feld *Definiert im Schema* können Sie die Taxonomie auswählen, in der die Arcrole definiert ist. In den Feldern *ID* und *Role* können Sie die verfügbaren Arcroles auswählen. Nachdem Sie eine Arcrole ausgewählt und auf OK geklickt haben, wird die Referenz zur Linkbase hinzugefügt. Auf dem Register "Taxonomien" wird für die von Ihnen referenzierte Arcrole in der Spalte "Referenzierende Linkbase-Dateien" die referenzierende Linkbase angezeigt. Wenn Sie diese Arcrole für eine bestimmte Beziehungsart zur Verfügung stellen möchten, müssen Sie jedoch dennoch das Kontrollkästchen für die jeweilige Beziehungsart aktivieren.

Nachdem eine Arcrole in der Taxonomie definiert wurde, können Sie damit die Rolle eines Arc in einer Beziehungsart definieren, für die der Arc seiner Definition gemäß verfügbar ist. In der Abbildung oben wurde die Arcrole z.B. für Arcs in Label-Beziehungen zur Verfügung gestellt.

Die Arcrole eines Arc wird in der Eingabehilfe "Details" ausgewählt (*Abbildung unten; Arcrole ist markiert)*.

 © 2016-2022 Altova GmbH



Wählen Sie in der Eingabehilfe "Details" mit dem Element am `to` Ende eines ausgewählten Arc den gewünschten Eintrag aus der Dropdown-Liste des Arcrole-Eintrags aus.

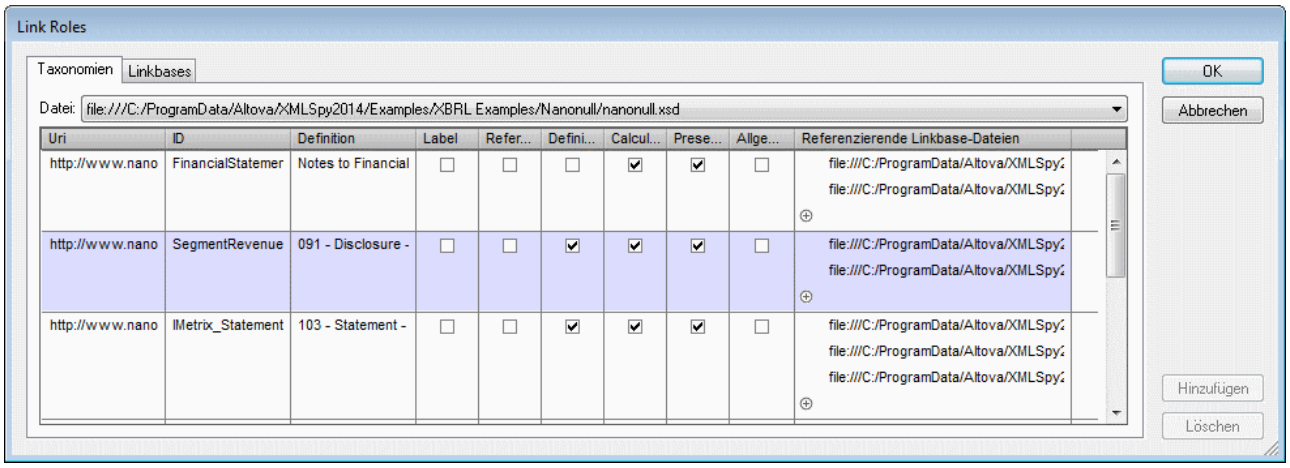
28.16.2 Linkroles

Mit dem Befehl **Linkroles** wird das Dialogfeld "Link Roles" (*Abbildung unten*) aufgerufen, in dem Linkroles für eine Taxonomie erstellt werden können. Linkroles werden in der Concept Definitions-Datei im `appinfo` Element gespeichert (*siehe Liste unten).* Linkroles werden nicht nur in `definitionLink` Elementen, sondern auch in den umschließenden Elementen anderer Beziehungsarten verwendet (z.B. in `calculationLink` und `presentationLink` Elementen).

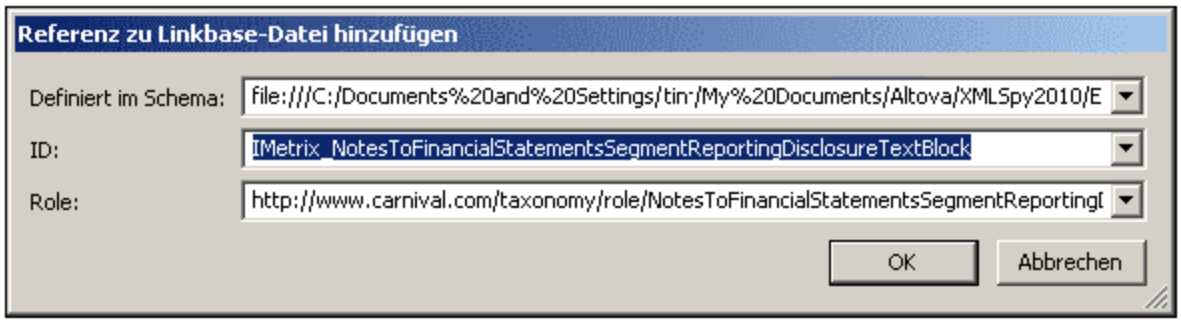
```
<xs:appinfo>
   <link:roleType id="SegmentRevenueAndOperatingIncome"
   roleURI="http://www.nanonull.com/taxonomy/role/SegmentRevenueAndOperatingIncome">
      <link:definition>006091 - Disclosure - Segment Revenue and Operating
Income</link:definition>
      <link:usedOn>link:calculationLink</link:usedOn>
      <link:usedOn>link:definitionLink</link:usedOn>
      <link:usedOn>link:presentationLink</link:usedOn>
   </link:roleType>
</xs:appinfo>
```

Beachten Sie in der Auflistung oben, dass sie `usedOn` Elemente enthält, die definieren, in welcher Art von Beziehung diese Linkrole verwendet werden kann.

Auf dem Register "Taxonomien" können Sie im Dialogfeld "Link Roles" (*Abbildung unten*) eine Linkrole zu einer Taxonomiedatei hinzufügen, indem Sie auf die Schaltfläche **Hinzufügen** klicken. Definieren Sie anschließend die URI und die ID der Linkrole (*siehe Liste oben*). Um festzulegen, für welche Beziehungsarten eine Linkrole zur Verfügung stehen soll, aktivieren Sie die Kästchen der gewünschten Beziehungsarten. In der Spalte "Referenzierende Linkbase-Dateien" können Sie für jede Linkrole die Linkbase-Dateien hinzufügen bzw. löschen, die die Linkrole referenzieren.



Auf dem Register **Linkbases** sehen Sie eine andere Ansicht der Linkroles der Taxonomie. In dieser Ansicht können Sie Linkroles nach Linkbases hinzufügen und anzeigen (z.B. Calculation oder Presentation Linkbases). Wählen Sie in der Auswahlliste eine Linkbase aus und fügen Sie anschließend je nach Bedarf eine Linkrole hinzu oder löschen Sie eine. Wenn Sie auf die Schaltfläche **Hinzufügen** klicken, wird das Dialogfeld "Referenz zu Linkbase-Datei hinzufügen" angezeigt (*Abbildung unten)*.

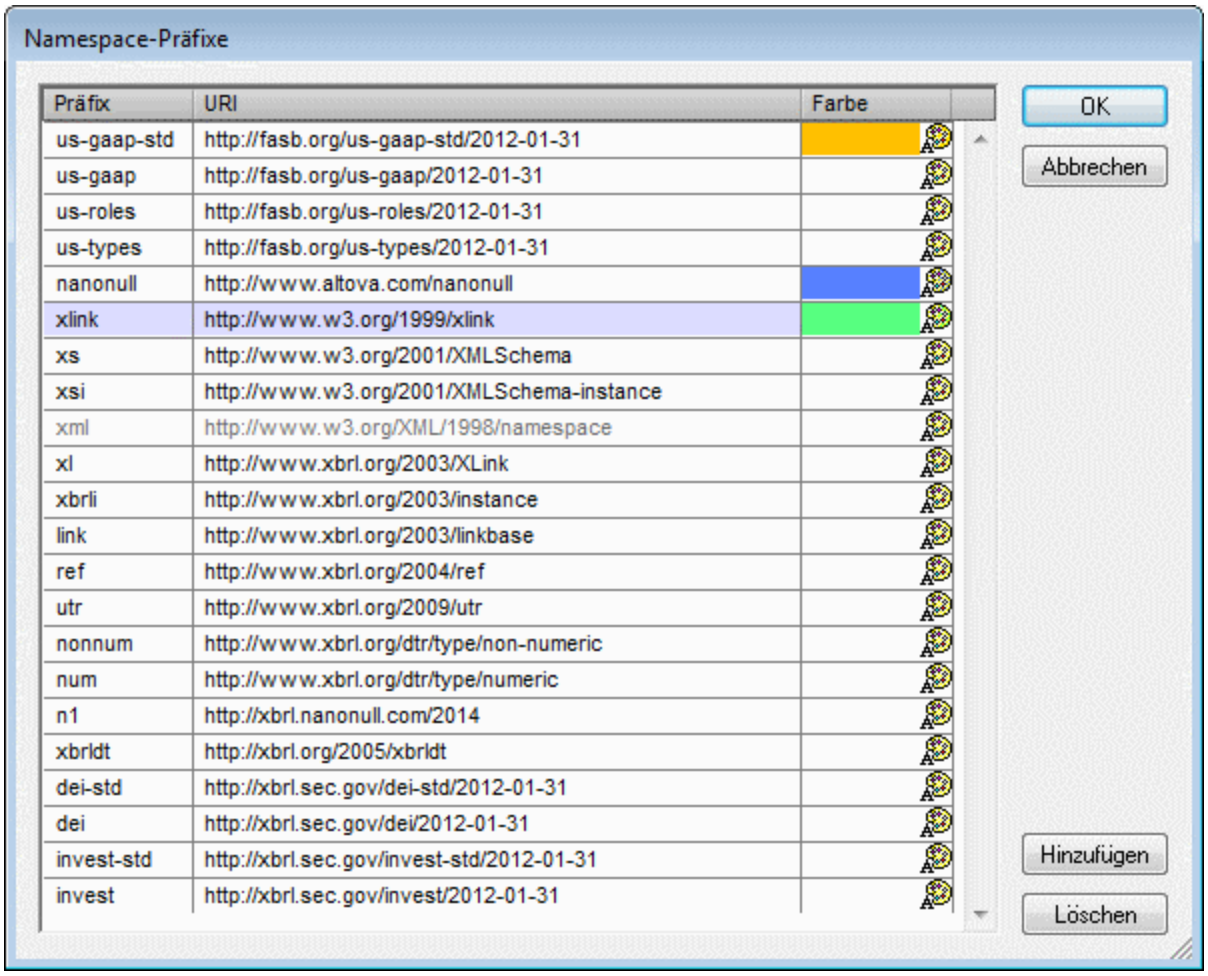


Bei den einzelnen Feldern in diesem Dialogfeld handelt es sich um Auswahllisten, über die Sie zwischen verfügbaren Optionen wählen können. Im Feld *Definiert im Schema* können Sie die Taxonomie auswählen, in der die Linkrole definiert ist. In den Feldern *ID* und *Role* können Sie die verfügbaren Linkroles auswählen. Nachdem Sie eine Linkrole ausgewählt und auf OK geklickt haben, wird die Referenz zur Linkbase hinzugefügt. Auf dem Register "Taxonomien" wird für die von Ihnen referenzierte Linkrole in der Spalte "Referenzierende Linkbase-Dateien" die referenzierende Linkbase angezeigt. Wenn Sie diese Linkrole für eine bestimmte Beziehungsart zur Verfügung stellen möchten, müssen Sie jedoch dennoch das Kontrollkästchen für die jeweilige Beziehungsart aktivieren.

Nachdem eine Linkrole in der Taxonomie erstellt wurde, wird sie beim Erstellen von Beziehungen[869] verwendet.

28.16.3 Namespace-Präfixe

Mit dem Befehl **Namespace-Präfixe** wird das Dialogfeld "Namespace-Präfixe" (*Abbildung unten)* aufgerufen, in dem alle Namespaces in der Taxonomie einschließlich der Namespaces aus importierten Taxonomien angezeigt werden. Sie können in diesem Dialogfeld Namespaces und Präfixe bearbeiten und Hintergrundfarben für bestimmte Namespaces definieren. Wenn eine Hintergrundfarbe für einen Namespace definiert wird, werden Elemente in diesen Namespaces im Hauptfenster und in den Eingabehilfen mit dieser Hintergrundfarbe angezeigt. Beachten Sie, dass eine Farbeinstellung für einen bestimmten Namespace auf diesen Namespace in allen Taxonomiedokumente, die in der XBRL-Ansicht geöffnet sind, angewendet wird.

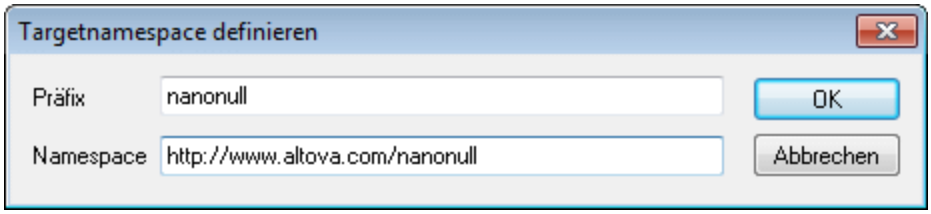


Um einen Namespace hinzuzufügen oder zu löschen, verwenden Sie die Schaltfläche **Hinzufügen** bzw. **Löschen**. Eine Farbe wird einem Namespace über die Farbpalette für diese Namespace zugewiesen. Wenn Sie mit den Bearbeitungen im Dialogfeld "Namespaces" fertig sind, klicken Sie auf OK, um sie zu speichern.

Auch der **Target Namespace** der Taxonomie wird im Dialogfeld "Namespaces" aufgelistet. Der Target Namespace kann auch über den Befehl **Target Namespace definieren**[1538] definiert werden. Nähere Informationen zu Target Namespaces finden Sie im XBRL-Abschnitt der Dokumentation[856].

28.16.4 Target Namespace definieren

Mit dem Befehl "Target Namespace definieren" kann ein Target Namespace für ein Taxonomiedokument definiert werden. Bei Aufruf des Befehls wird das Dialogfeld "Target Namespace definieren" angezeigt (*Abbildung unten*), in dem Sie den gewünschten Target Namespace und ein Präfix dafür eingeben können. Klicken Sie zum Abschluss auf **OK.**



Der Target Namespace wird definiert und auch deklariert:

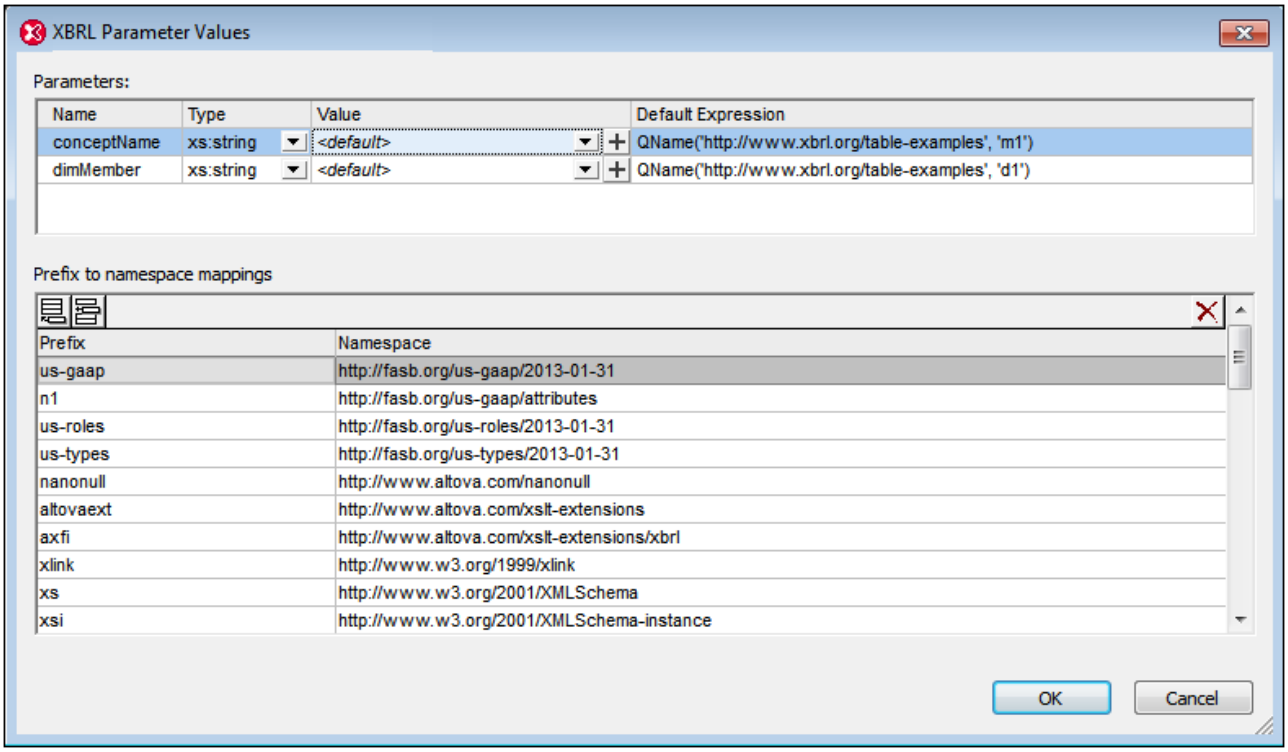
```
<xs:schema targetNamespace="http://www.altova.com/XBRL/Taxonomies"
    xmlns:ns1="http://www.altova.com/XBRL/Taxonomies" >
    ...
</xs:schema>
```

In der Auflistung oben ist der Target Namespace mit dem Attribut `targetNamespace` definiert und anschließend mit dem Präfix `ns1` deklariert.

28.16.5 Parameterwerte

Mit dem Befehl **Parameterwerte** wird das Dialogfeld "XBRL-Parameterwerte" angezeigt (*Abbildung unten*). Darin werden auf den Registern "Formula" (Formula-Parameter) und "Tabelle" (Tabellenparameter) definierte Parameter angezeigt. Sie können im Dialogfeld "XBRL-Parameterwerte" den Datentyp des Parameters bearbeiten und einen Parameterwert definieren, der den Standardwert außer Kraft setzt. Der von Ihnen eingegebene Parameterwert setzt den über das Diagramm eingegebenen Standardwert außer Kraft. Da die Tabellenparameter mehrere Werte erhalten können, können Sie durch Klicken auf das + Symbol in der Spalte *Wert* zusätzliche Parameterwerte für einen Parameter hinzufügen.

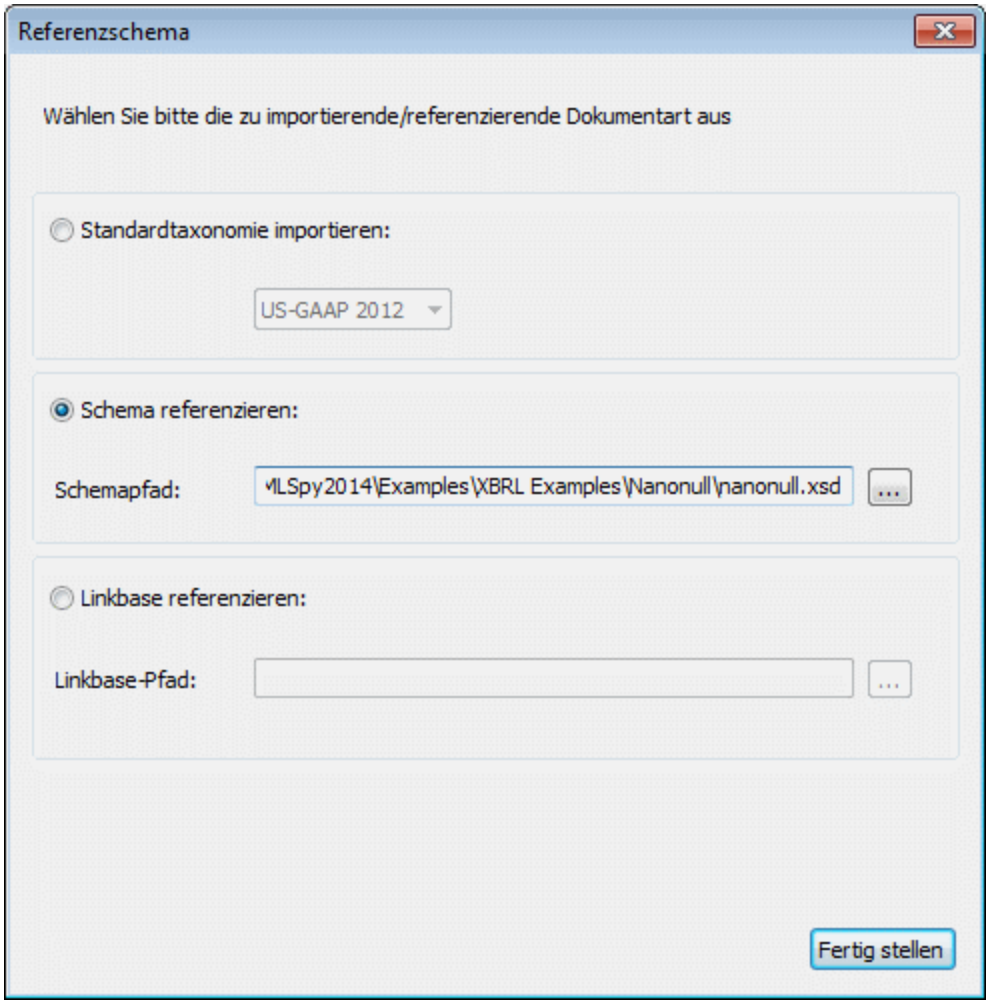
© 2016-2022 Altova GmbH



Die in diesem Dialogfeld zugewiesenen Werte von globalen Parametern werden nur für Tabellenparameter ausgewertet. Formula-Parameter werden zwar angezeigt, können in diesem Dialogfeld aber nicht bearbeitet werden.

28.16.6 Importieren/Referenzieren

Mit dem Befehl **Importieren/Referenzieren** wird ein Dialogfeld aufgerufen (*Abbildung unten*), in dem Sie das zu importierende Schema oder die zu referenzierende Linkbase festlegen können.



Das Dialogfeld enthält die folgenden Optionsfelder:

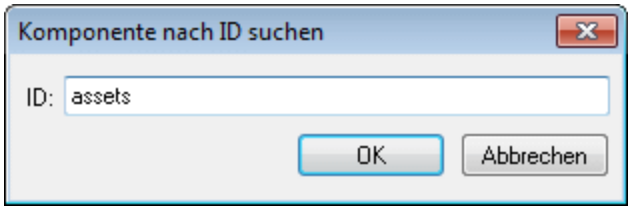
- *Standardtaxonomie importieren:* Mit dieser Option können Sie eine US-GAAP-Taxonomie oder eine IFRS-Taxonomie schnell und korrekt importieren. Wählen Sie die Standardtaxonomie im Dropdown-Menü der Auswahlliste aus und klicken Sie auf "Weiter". Daraufhin wird das jeweils nächste Dialogfeld angezeigt, in dem Sie Ihre Definition der gewünschten Taxonomie fertig stellen können. Dieser Vorgang wird im Abschnitt Erstellen einer neuen Taxonomie[851] beschrieben.
- *Importieren einer beliebigen Taxonomie (Schema referenzieren):* Mit Hilfe dieser Option können Sie jede beliebige Taxonomie importieren, indem Sie den Pfad zur Taxonomiedatei (`.xsd`-Datei) angeben.
- *Linkbase referenzieren:* Sie können eine Linkbase definieren, die in die Taxonomie inkludiert werden soll. Geben Sie dazu den Pfad zur Linkbase-Datei ein und klicken Sie auf **Fertig stellen**. Daraufhin wird in der Taxonomie eine Referenz zur Linkbase-Datei erstellt. Der Beziehungstyp der neu referenzierten Linkbase kann dann durch Rechtsklick auf den Dateinamen und Auswahl des Befehls **Linkbase-Art definieren**[859] definiert werden.

```
<xsl:template match="*">
   <xsl:copy>
      <xsl:copy-of select="@*[not(.='')]"/>
      <xsl:apply-templates/>
```

```
    </xsl:copy>
</xsl:template>
```

28.16.7 Komponente nach ID suchen

In Taxonomien mit großen Formula- oder Tabellen-Linkbases, die mehrere Komponenten derselben Art enthalten, empfiehlt sich oft eine Komponentensuche anhand der ID. Über den Menübefehl **XBRL | Komponente nach ID suchen** können Sie eine Suche nach ID durchführen. Bei Aufruf des Befehls erscheint ein Dialogfeld, in das Sie die zu suchende ID eingeben müssen (*Abbildung unten*).



Klicken Sie zum Starten der Suche auf **OK**.

28.16.8 Dokumentation generieren

Mit dem Befehl **XBRL | Dokumentation generieren** wird die Dokumentation für die aktuelle XBRL-Taxonomie generiert. Die Ausgabe der Dokumentation kann als HTML-, MS Word, RTF- oder PDF-Datei erfolgen. Diese Dokumentation kann nach Wunsch geändert und verwendet werden; Sie benötigen dafür keine Genehmigung von Altova. Die Dokumentation wird für die Komponenten generiert, die Sie im Dialogfeld "XBRL-Taxonomiedokumentation" auswählen (Aufruf des Dialogfelds mit dem Befehl "Dokumentation generieren"). Miteinander in Zusammenhang stehende Komponenten werden in der Ausgabe auf dem Bildschirm durch Hyperlinks verbunden, um die Navigation zu vereinfachen. Die verschiedenen Optionen für die Generierung von Dokumentation werden im Abschnitt "Dokumentationsoptionen[1541]" beschrieben.

Anmerkung: Um Dokumentation im MS Word-Format generieren zu können, muss MS Word (Version 2000 oder höher) installiert sein.

Sie können für das generierte Dokument entweder das festgelegte Standard-Design von XMLSpy verwenden oder für das Design ein StyleVision SPS verwenden. Die Verwendung eines StyleVision SPS hat den Vorteil, dass Sie das Design der generierten Dokumentation anpassen sowie als zusätzliches Ausgabeformat PDF generieren können. Wie man mit einem SPS arbeitet, wird in im Abschnitt "Benutzerdefiniertes Design[1543]" erklärt.

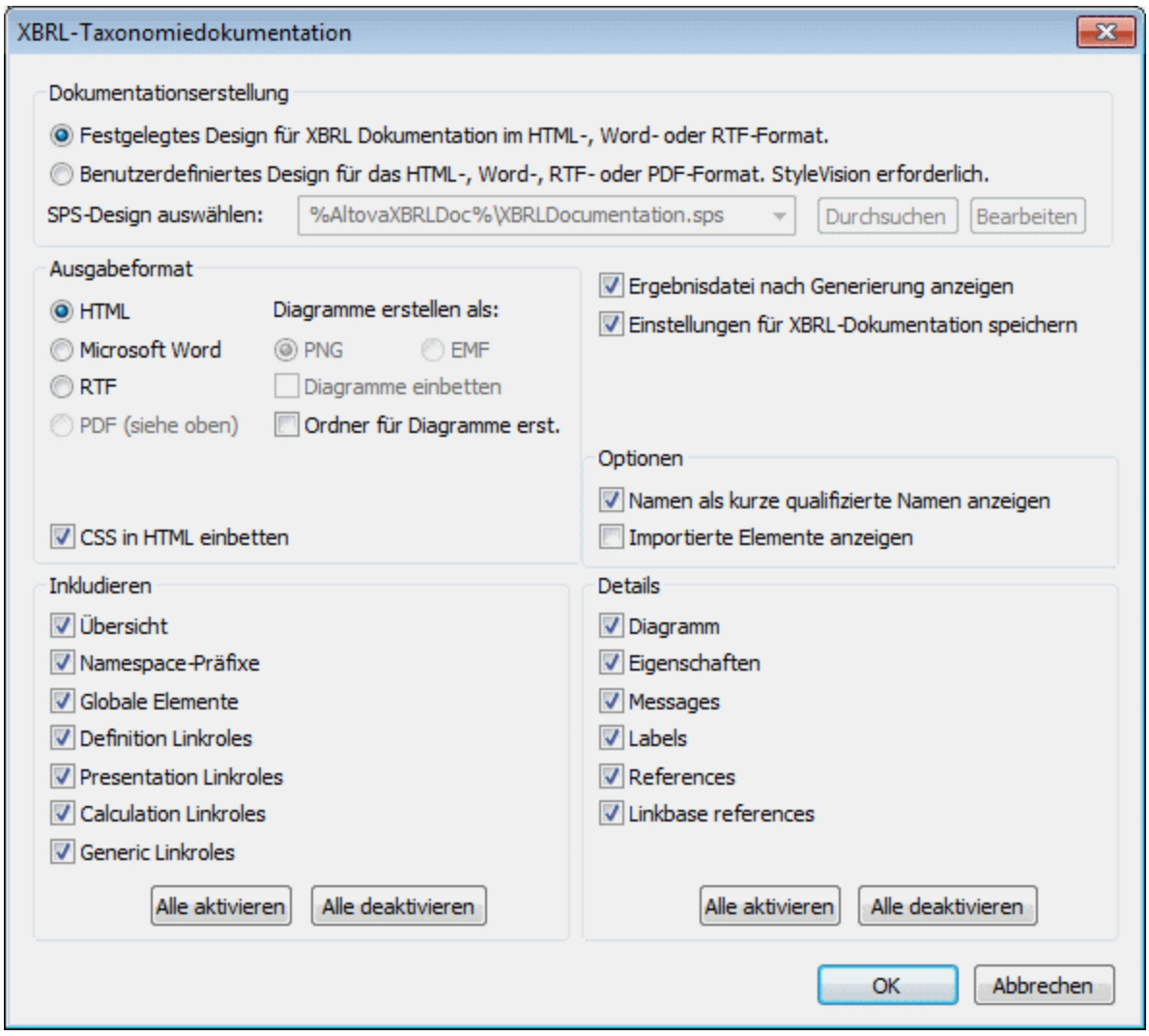
Anmerkung: Um ein SPS zum Generieren von Schema-Dokumentation zu verwenden muss StyleVision auf Ihrem Computer installiert sein.

28.16.8.1 Dokumentationsoptionen

Mit dem Befehl **XBRL | Dokumentation generieren** wird das Dialogfeld Schema-Dokumentation geöffnet (*siehe Abbildung unten*), in dem Sie die Optionen für die Dokumentation auswählen können.

Im Bereich "Dokumentationserstellung" des Dialogfelds können Sie wählen, ob Sie für die generierte Dokumentation das festgelegte XMLSpy-Design oder ein in einem StyleVision SPS erstelltes angepasstes Design verwenden möchten. Wählen Sie die gewünschte Option. Beachten Sie, dass PDF-Ausgabe nur für mit einem StyleVision SPS erstellte Dokumentation und nicht für mit einem festgelegten Design generierte Dokumentation zur Verfügung steht. Wie Sie mit einem benutzerdefinierten Design arbeiten, wird im Abschnitt "Benutzerdefiniertes Design" beschrieben.

Anmerkung: Um ein SPS zum Generieren von Schema-Dokumentation zu verwenden muss StyleVision auf Ihrem Computer installiert sein.



Bei Aufruf des Befehls Dokumentation generieren wird das Dialogfeld XBRL-Taxonomiedokumentation geöffnet:

- Das gewünschte Format wird im Bereich "Ausgabeformat" definiert: entweder HTML, Microsoft Word, RTF oder PDF. (Das Ausgabeformat PDF steht nur zur Verfügung wenn Sie ein StyleVision SPS zur Generierung der Dokumentation verwenden.) Wenn Sie auf OK klicken, werden Sie aufgefordert, den Namen der Ausgabedatei und den Ausgabepfad dafür anzugeben.

© 2016-2022 Altova GmbH

- Microsoft Word-Dokumente werden mit der Dateierweiterung `.doc` erstellt, wenn sie mit Hilfe eines festgelegten Designs generiert werden, und mit der Dateierweiterung `.docx`, wenn sie mit Hilfe eines StyleVision SPS generiert werden.
- Für die HTML-Ausgabe können die CSS-Stildefinitionen entweder in einer separaten CSS-Datei gespeichert oder in der HTML-Datei (im Element `<head>`) eingebettet werden. Wenn eine separate CSS-Datei erstellt wird, erhält sie denselben Namen wie die HTML-Datei, jedoch mit der Erweiterung `.css`. Aktivieren oder deaktivieren Sie das Kontrollkästchen *CSS in HTML einbetten* um die gewünschte Option festzulegen.
- Die Option "Diagramme einbetten" ist für die Ausgabeoptionen MS Word, RTF und PDF aktiv. Wenn diese Option aktiviert ist, werden Diagramme entweder im PNG- oder im EMF-Format in die Ergebnisdatei eingebettet. Andernfalls werden die Diagramme als PNG- oder EMF-Dateien generiert und in der Ergebnisdatei über Objektlinks angezeigt.
- Wenn es sich bei der Ausgabedatei um eine HTML-Datei handelt, werden alle Diagramme als dokumentexterne PNG-Dateien generiert. Wenn das Kontrollkästchen *Ordner für Diagramme erstellen* aktiviert ist, wird im selben Ordner wie die HTML-Datei ein Ordner erstellt, in dem die PNG-Dateien gespeichert werden. Dieser Ordner erhält einen Namen im Format *`HTMLDateiname`*`_diagramme`. Wenn das Kontrollkästchen *Ordner für Diagramme erstellen* deaktiviert ist, werden die PNG-Dateien im selben Ordner wie die HTML-Datei gespeichert.
- Im Bereich **Inkludieren** können Sie auswählen, welche Datenelemente in der Dokumentation inkludiert werden sollen. Bei Auswahl der Option "Übersicht" werden alle Komponenten nach Komponententyp geordnet am Anfang der Datei aufgelistet. Über die Schaltflächen **Alle aktivieren** und **Alle deaktivieren** können Sie alle Optionen im Bereich einfach aktivieren bzw. deaktivieren.
- Im Bereich **Details** werden die Einzelheiten aufgelistet, die für die einzelnen Komponenten inkludiert werden können. Wählen Sie die gewünschten Informationen aus. Über die Schaltflächen **Alle aktivieren** und **Alle deaktivieren** können Sie alle Optionen im Bereich einfach aktivieren bzw. deaktivieren. Das Kontrollkästchen *Messages* ist nur aktiviert, wenn im Inkludieren-Bereich die Option *Generic Linkroles* aktiviert ist. Alle anderen Kontrollkästchen sind aktiviert, wenn im Inkludieren-Bereich *Globale Elemente* oder *Generic Linkroles* aktiviert ist.
- Die Option "Ergebnisdatei nach der Generierung anzeigen" ist für alle Ausgabeoptionen aktiv. Wenn diese Option aktiviert ist, wird die erzeugte Datei in der Browser-Ansicht (HTML-Ausgabe), MS Word (MS Word Ausgabe) und den Standardapplikationen für `.rtf`-Dateien (RTF-Ausgabe) und `.pdf`-Dateien (PDF-Ausgabe) angezeigt. Sie können auch auswählen, ob die Links im Ausgabedokument absolut oder relativ sein sollen.
- Im Bereich "Optionen" können Sie auswählen, ob (i) Elementnamen nur mit einem Präfix (kurzer qualifizierter Name) oder in erweiterter Form (mit dem gesamten Namespace) angezeigt werden sollen; und (ii) ob auch importierte Elemente angezeigt werden sollen.

Parameterwerte

Wenn das StyleVision SPS einen oder mehrere Parameterdefinitionen enthält, wird beim Klicken auf **OK** ein Dialogfeld aufgerufen, in dem alle Parameter aufgelistet werden, die im SPS definiert sind. Sie können in diesem Dialogfeld Parameterwerte eingeben um die Standardwerte zu überschreiben, die im SPS zugeordnet worden sind.

28.16.8.2 Benutzerdefiniertes Design

Sie können anstatt des festgelegten Standard XMLSpy Designs ein angepasstes Design für die XBRL-Taxonomiedokumentation erstellen. Das angepasste Design wird in einem StyleVision SPS erstellt, das eine Design-Vorlage für das Ausgabedokument darstellt.

Erstellen des SPS

Ein StyleVision Power Stylesheet (oder SPS) wird mit Hilfe des StyleVision Produkts von Altova erstellt. Einem SPS zur Generierung von XBRL-Taxonomiedokumentation muss ein XML-Schema zugrunde liegen, das die Struktur der XBRL-Taxonomiedokumentation festlegt. Dieses Schema heißt `XBRLDocumentation.xsd` und wird mit Ihrem XMLSpy Paket mitgeliefert. Es wird im Verzeichnis `C:\Dokumente und Einstellungen\<Benutzername>\Eigene Dateien\Altova\XMLSpy2023\Dokumentation\XBRL`. gespeichert.

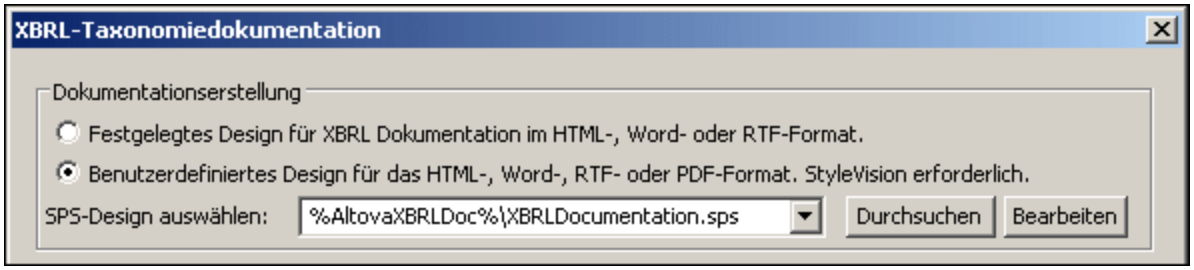
Beim Erstellen des SPS Designs in StyleVision werden Nodes aus dem Schema `XBRLDocumentation.xsd` in die Design-Vorlage eingefügt und Formatvorlagen und Eigenschaften zugewiesen. Zusätzliche Komponenten wie Links, Tabellen und Bilder können ebenfalls zum SPS hinzugefügt werden. Auf diese Weise kann die gesamte Ausgabe im SPS gestaltet werden. Wie man ein SPS Design in StyleVision erstellt wird im Benutzerhandbuch von StyleVision im Detail beschrieben.

Die Verwendung eines SPS zur Generierung von XBRL-Taxonomiedokumentation hat den Vorteil, dass Sie die vollständige Kontrolle über das SPS Design haben. Beachten Sie auch, dass die PDF-Ausgabe der XBRL-Taxonomiedokumentation nur dann verfügbar ist, wenn ein benutzerdefiniertes SPS verwendet wird; PDF-Ausgabe steht nicht zur Verfügung, wenn das festgelegte XMLSpy Design verwendet wird.

Festlegen des für die XBRLTaxonomiedokumentation verwendete SPS

Nachdem ein SPS erstellt worden ist, kann es zum Generieren von XBRL-Taxonomiedokumentation verwendet werden. Das SPS, das Sie für die Generierung der XBRL-Taxonomiedokumentation verwenden möchten wird im Dialogfeld "XBRL-Taxonomiedokumentation" (aufgerufen über den Befehl **XBRL | Dokumentation generieren...**) ausgewählt. Wählen Sie im Bereich "Dokumentationserstellung" dieses Dialogfelds (*siehe Abbildung unten*) die Optionsschaltfläche *Beutzerdefiniertes Design für das HTML-, Word-, RTF- oder PDF-Format* aus. Sie können dann auf die Schaltfläche **Durchsuchen** klicken und das gewünschte SPS im Dateisystem auswählen. Klicken Sie im Dialogfeld auf **OK** und geben Sie im Dialogfeld Speichern, das erscheint, das Verzeichnis und den Namen der Ausgabedatei ein.

Anmerkung: Der Pfad zum Schema, auf dem die SPS-Datei basiert, (`WSDLDocumentation.xsd` (siehe oben) muss im SPS korrekt angegeben sein:



Mit XMLSpy wird ein editierbares SPS Design für die XBRL-Taxonomiedokumentation geliefert. Es heißt `XBRLDocumentation.sps` und befindet sich im Verzeichnis `C:\Dokumente und Einstellungen\<Benutzername>\Eigene Dateien\Altova\XMLSpy2011\Dokumentation\XBRL\Beispieldaten`. Diese SPS-Datei sowie alle anderen SPS-Dateien, die Sie kürzlich aufgerufen haben, stehen in der Dropdown-Liste der Option "Benutzerdefiniertes Design" zur Verfügung (*siehe Abbildung oben*).

Durch Klicken auf die Schaltfläche **Bearbeiten** im Bereich "Dokumentationserstellung" wird StyleVision gestartet und das ausgewählte SPS in einem StyleVision Fenster geöffnet. Um eine Vorschau auf das

© 2016-2022 Altova GmbH

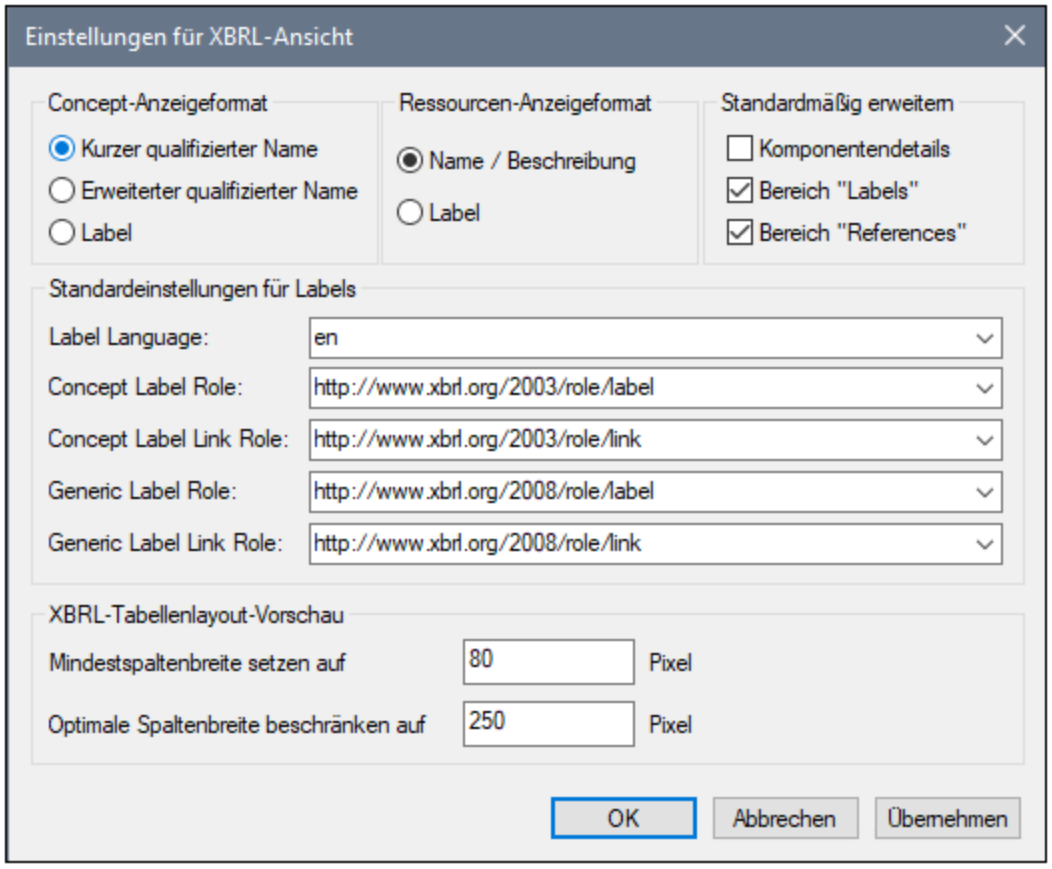
resultierende Dokument in StyleVision zu erhalten, benötigen Sie eine funktionierende XML-Datei. Zu diesem Zweck wird eine Beispieldatei namens `nanonull.xml` mit Ihrer Applikation mitgeliefert; sie befindet sich im Verzeichnis:

```
C:\Dokumente und Einstellungen\<Benutzername>\Eigene
Dateien\Altova\XMLSpy2023\Dokumentation\XBRL\Beispieldaten
```

Anmerkung: Um ein SPS zum Generieren von XBRL-Taxonomiedokumentation verwenden zu können, muss StyleVision auf Ihrem Computer installiert sein.

28.16.9 Ansichtseinstellungen

Mit dem Befehl **Ansichtseinstellungen** wird das Dialogfeld "Einstellungen für XBRL-Ansicht" (*Abbildung unten*) aufgerufen, in dem Sie die Standardeinstellungen für die XBRL-Ansicht definieren können.



Es stehen die folgenden Einstellungen zur Verfügung:

- *Concept-Anzeigeformat:* Kann auf den kurzen, den erweiterten qualifizierten Namen oder auf Labels gesetzt werden. Diese Einstellungen werden auf das Hauptfenster und die Eingabehilfe "Details", nicht aber die Eingabehilfe "Globale Elemente" angewendet. Die Einstellungen für die Anzeige der Elementnamen in der Eingabehilfe "Globale Elemente" werden über die Menüleiste der Eingabehilfen[276] definiert.

- *Ressourcen-Anzeigeformat:* Ressourcen können auf den Registern "Formula" und "Tabelle" entweder nach ihrem Namen oder ihrem Label angezeigt werden. (Wenn kein Name zugewiesen wurde, wird die Beschreibung der Ressource verwendet).
- *Standardmäßig erweitern:* Im Hauptfenster können die Elementdetails, der Kasten "Labels" und der Kasten "References" standardmäßig erweitert angezeigt werden. Beachten Sie, das die erweiterten Kästen in diesem Fall nur dann sichtbar sind, wenn die Elementdetails (entweder standardmäßig oder manuell) erweitert sind. Bei jeder Aktualisierung der Ansicht (z.B. beim Wechseln von der Textansicht in die XBRL-Ansicht) wird die XBRL-Ansicht auf die Standardeinstellung zurückgesetzt.
- *Standardeinstellungen für Labels:* Hier kann definiert werden, welche Standardsprache und Standard Label Roles verwendet werden sollen, wenn keine Labels definiert sind. In der Auswahlliste für die jeweilige Eigenschaft wird eine Liste verfügbarer Werte angezeigt.
- *XBRL-Tabellenlayout-Vorschau:* Hier kann die minimale und maximale Spaltenbreite in Pixel definiert werden.

28.16.10 Mit MapForce XBRL anhand von DB, Excel, CSV generieren

Dieser Befehl startet die Generierung einer XBRL-Instanzdatei auf Basis der aktuell aktiven Taxonomie. Die Daten für die Instanzdatei stammen aus einem MS Excel-Datenblatt, einer Datenbank oder einer CSV-Datei. Die XBRL-Instanzdatei wird mit Altova MapForce generiert, welches auf Ihrem Rechnet installiert sein muss. Der Befehl startet Altova MapForce und lädt die Taxonomie. Anschließend können Sie die Quelldatei definieren und die gewünschte Instanzdatei grafisch erstellen.

28.16.11 XBRL mit StyleVision anhand von HTML/PDF/Word darstellen

Dieser Befehl lädt die aktuell aktive Taxonomie in Altova StyleVision, sodass Sie anhand der Taxonomie ein Design für die Erstellung von Berichten entwerfen können. Um diesen Befehl verwenden zu können, muss Altova StyleVision auf Ihrem Rechner installiert sein.

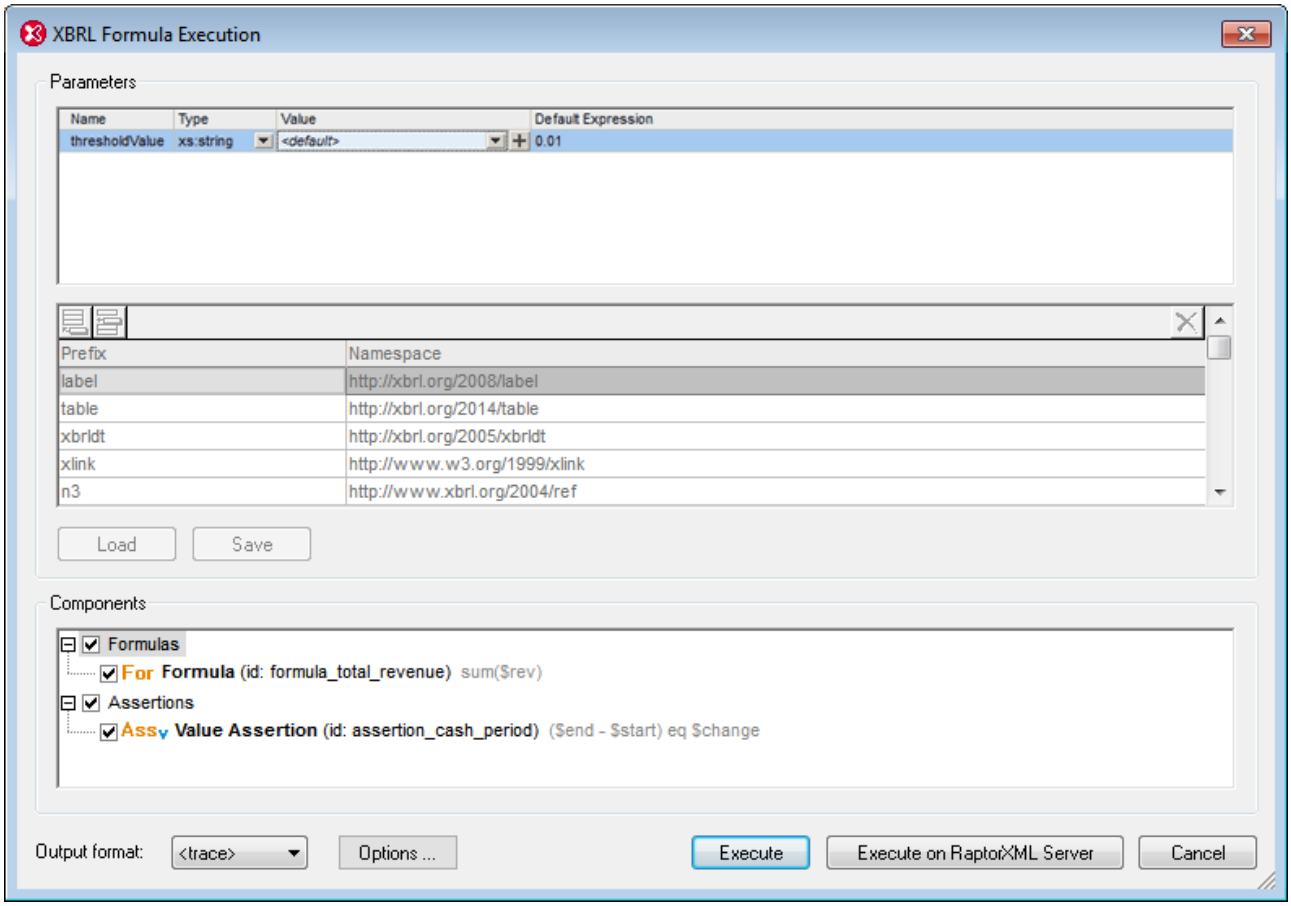
28.16.12 Formula (am Server) ausführen

Die Befehle **Formula ausführen** und **Formulas am Server ausführen (Hochleistungsausführung)** sind aktiv, wenn ein XBRL-Instanzdokument in der Text- oder Grid-Ansicht das aktive Dokument ist. Diese Befehle führen Formulas (Formeln) und/oder Assertions aus, die in dem mit der XBRL-Instanzdatei verknüpften DTS definiert sind. (DTS steht fürr Discoverable Taxonomy Set, eine Sammlung von Taxonomien.) XBRL-Formulas werden mit den Daten in der XBRL-Instanzdatei ausgewertet. Die Ergebnisse werden in eine XBRL-Instanzdatei geschrieben. Assertions werden separat ausgewertet und die Ergebnisse werden in eine JSON- oder XML-Datei geschrieben.

Der Befehl **Formula am Server ausführen (Hochleistungsausführung)** verwendet einen mit dem Programm verknüpften RaptorXML+XBRL Server[1088] zum Ausführen der Formulas. Mit dem Befehl **Extras | Raptor-Server verwalten**[1579] können Sie einen RaptorXML+XBRL Server konfigurieren.

© 2016-2022 Altova GmbH

Falls das DTS keine Formulas oder Assertions enthält, wird eine entsprechende Meldung angezeigt. Falls das DTS eine gültige Formula oder Assertion enthält, wird das Dialogfeld "XBRL Formula-Ausführung" (*Abbildung unten*) angezeigt.



Parameter
Wenn im DTS Parameter definiert sind, werden die einzelnen Parameter im Bereich "Parameter" angezeigt und Sie können dafür Werte eingeben. Parameternamen sind schreibgeschützt. Obligatorische Parameter werden mit einem roten Ausrufezeichen gekennzeichnet und die **OK**-Schaltfläche bleibt so lange deaktiviert, bis dem Parameter ein Wert zugewiesen wurde. Optionale Parameter haben einen Standardwert. Wenn ein obligatorischer Typ definiert ist, wird der Typ angezeigt. Parameter, für die mehrere Werte benötigt werden, sind durch ein **+** Symbol gekennzeichnet, auf welches Sie klicken können, um einen Wert hinzuzufügen. Beachten Sie, dass optionale Parameter ohne Wert nicht zur Ausführung an den Prozessor übergeben werden. Standardwerte sind schreibgeschützt und werden ausgeführt, wenn der Benutzer keinen Wert eingibt.

Namespace Mappings
In dieser Tabelle werden die Präfixe definiert, die in den QNames von Parametern und Typen verwendet werden. Hier können Sie zusätzliche Namespaces für die Parameterauswertung definieren.

Speichern und Laden von Parametern
Die Parametereinstellungen einschließlich der Namespace Mappings können durch Klicken auf **Speichern** im JSON- oder XML-Format gespeichert werden. Das Dateiformat hängt von der Dateierweiterung ab, die die Datei erhält. Beachten Sie, dass optionale Parameter ohne Wert nicht gespeichert werden. Nachdem eine

Parameterdatei gespeichert wurde, kann sie über die Schaltfläche **Laden** in das Dialogfeld geladen werden.

XBRL-Verarbeitungsoptionen
Über die Schaltfläche **Optionen** wird das Dialogfeld "XBRL-Verarbeitungsoptionen" 1555 aufgerufen, in dem Sie die Entfernung von Duplikaten aktivieren können (so dass doppelt vorhandene Facts automatisch ignoriert werden).

Komponenten

Dieser Bereich enthält eine Strukturansicht, über die auszuführende Formula- und/oder Assertion-Komponenten ausgewählt werden können. Zu jedem Eintrag wird ein Symbol und eine Beschreibung angezeigt sowie, falls vorhanden, die ID und der Ausdruck. Um eine Komponente für die Ausführung auszuwählen, aktivieren Sie ihr Kontrollkästchen. Assertion-Ausführungen können entweder im JSON- oder XML-Format ausgegeben werden; wählen Sie das gewünschte Ausgabeformat in der Auswahlliste "*Ausgabeformat*" aus. Das Ausgabeformat einer XBRL Formula-Ausführung ist immer XML.

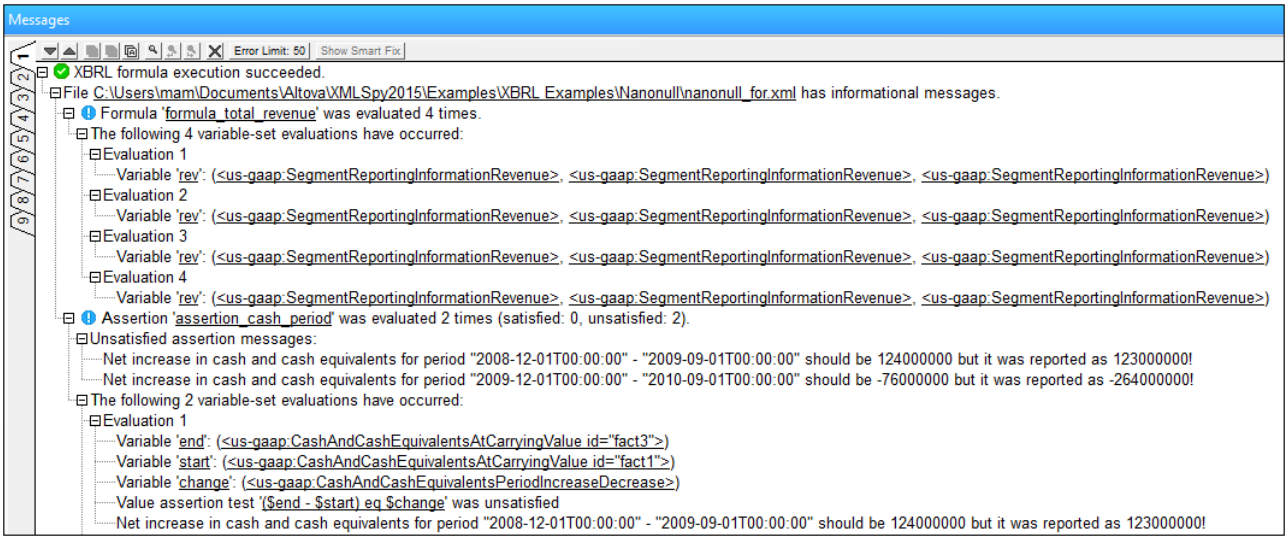
Ausführung

Sie können auswählen, ob die Ausführung mit dem internen Prozessor von XMLSpy oder mit Altova RaptorXML Server 1088 ausgeführt werden soll. Im Fall eines Ausführungsfehlers wird im Ausgabefenster eine Fehlermeldung angezeigt. Andernfalls wird eine Meldung angezeigt, dass die Ausführung erfolgreich war. Die Ausgabedateien `assertions-ouput-file.xml/json` und/oder `formula-output-file.xml` werden in neuen Dokumentfenstern geöffnet und nicht auf der Festplatte gespeichert. Sie müssen die Datei explizit im gewünschten Ordner speichern.

Trace

Wenn Sie in der Auswahlliste *Ausgabeformat* (am unteren Rand des Dialogfelds) **<trace>** auswählen, werden während der Ausführung der Formula zusätzliche Debugging-Informationen für "alle Variablengruppenauswertungen" gesammelt und anschließend im Fenster "Meldungen" angezeigt (*siehe Abbildung unten*). Beim Trace-Vorgang werden die einzelnen Variablengruppenauswertungen für jede einzelne Formula/Assertion an dem jeweiligen Punkt, an dem die eigentliche Zuweisung der Variablen in dieser Auswertung angezeigt werden, aufgelistet. Wenn die Variablen Instanz-Facts referenzieren, gelangen Sie bei Klick auf den jeweiligen Wert zum entsprechenden Fact-Element in der Instanz. Wenn Sie auf die Formula/Assertion oder den Variablennamen klicken, gelangen Sie zur entsprechenden Definition in den Formula Linkbase-Dateien. Im Fall von Validierungs-Assertions werden auch die Assertion-Meldungen, die in diesem Auswertungsschritt generiert wurden, angezeigt.

© 2016-2022 Altova GmbH



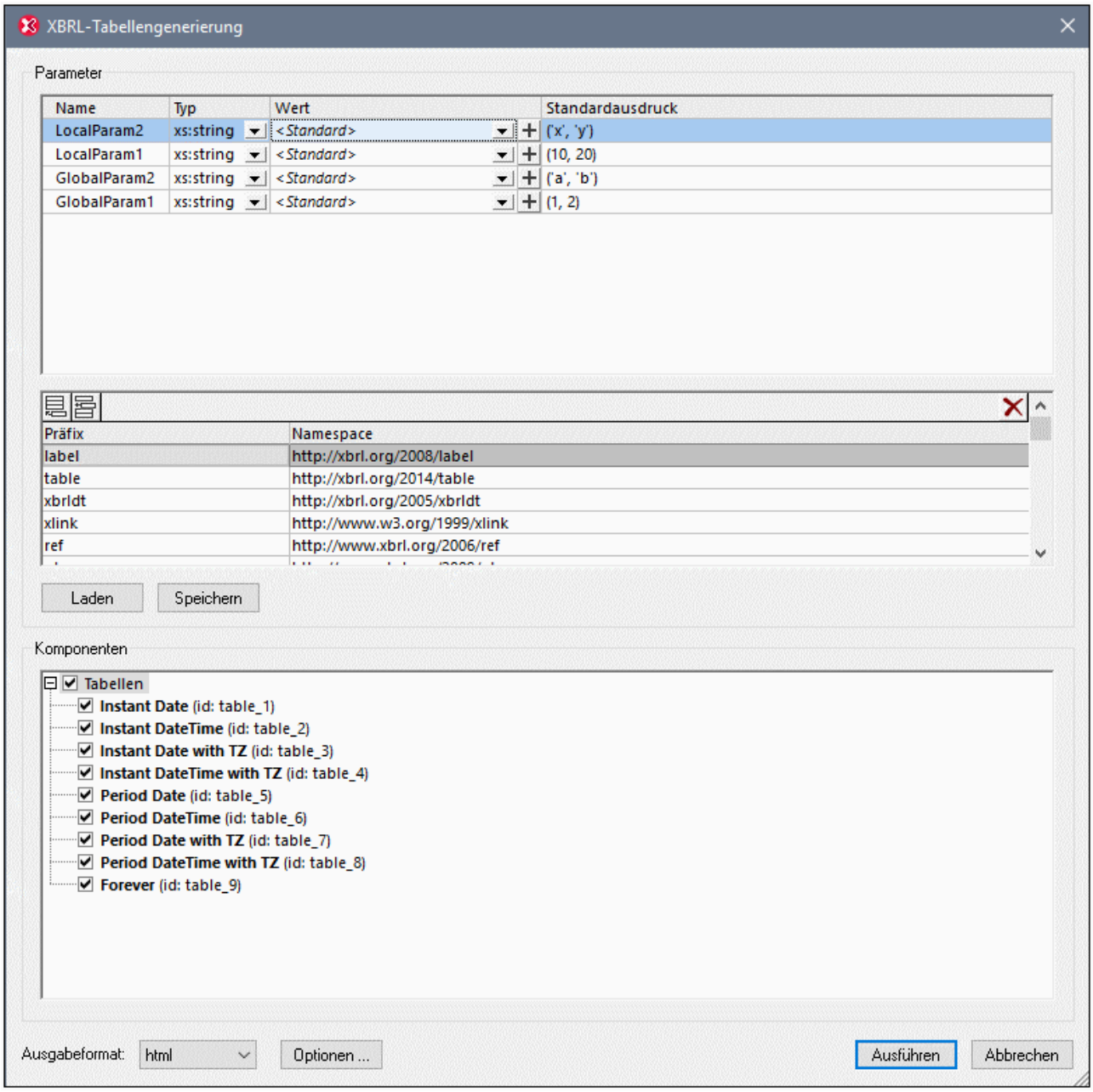
Anmerkung: Bei Durchführen eines Trace-Vorgangs wird unter Umständen viel Arbeitsspeicher benötigt und die Ausführung kann lange dauern. Bei großen XBRL-Instanzen werden die Assertions unter Umständen Millionen mal ausgewertet. Bei jeder Auswertung müssen manchmal die Werte von bis zu 49 Variablen gespeichert werden. Diese Funktion sollte daher nur zum Debuggen kleiner/reduzierter Beispiele verwendet werden, da die Ausführung sonst sehr lange dauern und zu wenig Arbeitsspeicher vorhanden sein könnte. Aus diesem Grund wurde im Code für jeden Trace-Vorgang ein Limit von 1000 Auswertungen definiert.

28.16.13 Tabelle (am Server) generieren

Die Befehle **Tabelle generieren** und **Tabelle am Server generieren (Hochleistungsausführung)** sind aktiv, wenn (i) ein XBRL-Instanzdokument das aktive Dokument in der Text- oder Grid-Ansicht ist oder (ii) wenn eine XBRL-Taxonomie das aktive Dokument in der XBRL-Ansicht, der Textansicht, Grid-Ansicht oder Schema-Ansicht ist. Mit diesen Befehlen wird ein XML- oder HTML-Dokument generiert, das XBRL-Tabellen enthält, die in dem mit dem aktiven Dokument verknüpften DTS definiert sind. (DTS steht fürr Discoverable Taxonomy Set, eine Sammlung von Taxonomien.) Im Fall von XBRL-Instanzdateien werden Tabellen mit Daten in der XBRL-Instanzdatei generiert.

Der Befehl **Tabelle am Server generieren (Hochleistungsausführung)** verwendet einen mit dem Programm verknüpften RaptorXML+XBRL Server[1088] zum Generieren der Tabelle. Mit dem Befehl **Extras | Raptor-Server verwalten**[1579] können Sie einen RaptorXML+XBRL Server konfigurieren.

Wenn im DTS keine Tabellen definiert sind, wird eine entsprechende Meldung angezeigt. Wenn es im DTS eine gültige Tabellendefinition gibt, wird das Dialogfeld "XBRL-Tabellengenerierung" (*Abbildung unten*) angezeigt.



Parameter

Wenn im DTS Parameter definiert sind, werden die einzelnen Parameter im Bereich "Parameter" angezeigt und Sie können dafür Werte eingeben. Parameternamen sind schreibgeschützt. Obligatorische Parameter werden mit einem roten Ausrufezeichen gekennzeichnet und die **OK**-Schaltfläche bleibt so lange deaktiviert, bis dem Parameter ein Wert zugewiesen wurde. Optionale Parameter haben einen Standardwert. Wenn ein obligatorischer Typ definiert ist, wird der Typ angezeigt. Parameter, für die mehrere Werte benötigt werden, sind durch ein **+** Symbol gekennzeichnet, auf welches Sie klicken können, um einen Wert hinzuzufügen. Beachten Sie, dass optionale Parameter ohne Wert nicht zur Ausführung an den Prozessor übergeben werden. Standardwerte sind schreibgeschützt und werden ausgeführt, wenn der Benutzer keinen Wert eingibt.

© 2016-2022 Altova GmbH

Namespace Mappings
In dieser Tabelle werden die Präfixe definiert, die in den QNames von Parametern und Typen verwendet werden. Hier können Sie zusätzliche Namespaces für die Parameterauswertung definieren.

Speichern und Laden von Parametern
Die Parametereinstellungen einschließlich der Namespace Mappings können durch Klicken auf **Speichern** im JSON- oder XML-Format gespeichert werden. Das Dateiformat hängt von der Dateierweiterung ab, die die Datei erhält. Beachten Sie, dass optionale Parameter ohne Wert nicht gespeichert werden. Nachdem eine Parameterdatei gespeichert wurde, kann sie über die Schaltfläche **Laden** in das Dialogfeld geladen werden.

XBRL-Verarbeitungsoptionen
Über die Schaltfläche **Optionen** wird das Dialogfeld "XBRL-Verarbeitungsoptionen" 1555 aufgerufen, in dem Sie die Entfernung von Duplikaten aktivieren können (so dass doppelt vorhandene Facts automatisch ignoriert werden).

Komponenten

Dieser Bereich enthält eine Strukturansicht, über die auszuführende Tabellenkomponenten ausgewählt werden können. Zu jedem Eintrag wird ein Symbol und eine Beschreibung angezeigt sowie, falls vorhanden, die ID. Um eine Tabellenkomponente für die Ausführung auszuwählen, aktivieren Sie ihr Kontrollkästchen. Ausführungen können entweder im XML- oder HTML-Format ausgegeben werden; wählen Sie das gewünschte Ausgabeformat in der Auswahlliste "*Ausgabeformat*" aus. Klicken Sie auf die Schaltfläche **Optionen,** um das Dialogfeld "XBRL-Verarbeitungsoptionen" 1555, in dem Sie Optionen für die Generierung von XBRL-Tabellen definieren und festlegen können, ob Duplikate ignoriert werden sollen, aufzurufen. Wenn bevorzugte Label-Optionen verfügbar sind, werden diese verwendet; andernfalls werden die im Dialogfeld XBRL-Verarbeitungsoptionen 1555 definierten Standardeinstellungen verwendet.

Ausführung

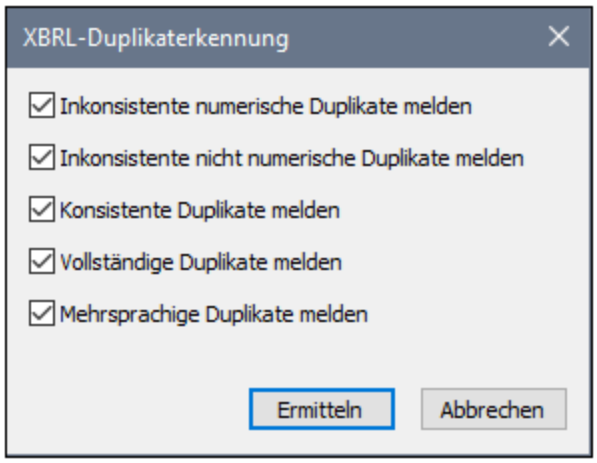
Im Fall eines Ausführungsfehlers wird im Ausgabefenster eine Fehlermeldung angezeigt. Andernfalls wird eine Meldung angezeigt, dass die Ausführung erfolgreich war. Die Ausgabedatei `table-ouput-file.xml/html` wird in einem neuen Dokumentfenster geöffnet und nicht auf der Festplatte gespeichert. Sie müssen die Datei explizit im gewünschten Ordner speichern.

28.16.14 Duplikate (am Server) ermitteln

Die Befehle **Duplikate ermitteln** und **Duplikate am Server ermitteln (Hochleistungsausführung)** sind aktiv, wenn ein XBRL-Instandokument in der Text- oder Grid-Ansicht das aktive Dokument ist. Mit diesen Befehlen wird das Instanzdokument auf doppelt vorhandene Facts überprüft und gefundene Duplikate werden im Fenster "Meldungen" aufgelistet. Beim Befehl **Duplikate am Server ermitteln (Hochleistungsausführung)** wird die Duplikatüberprüfung mit Hilfe eines verknüpften RaptorXML+XBRL Servers 1088 vorgenommen. Mit dem Befehl **Extras | Raptor Server verwalten** 1579 richten Sie einen RaptorXML+XBRL Server ein. Bei Auswahl eines der beiden Befehle wird das Dialogfeld "XBRL-Duplikaterkennung" (*Abbildung unten*) geöffnet. In diesem Dialogfeld können Sie auswählen, welche Arten von Duplikaten ermittelt werden sollen.

Eine ausführliche Erläuterung zu den verschiedenen Arten von Duplikaten finden Sie in der folgenden Spezifikation: Handling Duplicate Facts in XBRL and Inline XBRL 1.0. Im Folgenden sehen Sie eine kurze Übersicht darüber:

- *Vollständige Duplikate* sind Duplikate, die hinsichtlich Namen, Kontext, Wert identisch sind;
- *Konsistente Duplikate* sind doppelt vorhandene numerische Facts, die bis zu der für die Rundung definierten Dezimalstelle denselben Wert haben;
- Bei *mehrsprachigen Duplikaten* kommt dasselbe Fact in mehreren Sprachen vor;
- *Inkonsistente Duplikate* sind Duplikate, die die in der XBRL 2.1-Spezifikation festgelegten Bedingungen für doppelt vorhandene Facts erfüllen, aber keine vollständigen, konsistenten oder mehrsprachigen Duplikate sind (z.B. ein numerisches Duplikat mit einem anderen numerischen Wert).



28.16.15 XULE ausführen

Mit dem Befehl **XULE ausführen** werden XULE-Regeln an einem XBRL-Instanzdokument ausgeführt. Die XULE-Regeln können sich in einer einzigen `.xule`-Datei oder in einem ZIP-Archiv (`.zip`) mit XULE-Dokumenten befinden. *Nähere Informationen dazu finden Sie unter XBRL | Abschnitt XULE* 938.

Der Befehl ist in den folgenden Fällen aktiv:

- Wenn ein XULE-Dokument das aktive Dokument ist. Ein XULE-Dokument hat normalerweise die Dateierweiterung `.xule`. In diesem Fall werden Sie aufgefordert, eine XBRL-Instanz auszuwählen, an der das XULE-Dokument verarbeitet werden soll.
- Wenn ein XBRL-Instanzdokument (normalerweise mit der Dateierweiterung `.xbrl` oder `.xml`) das aktive Dokument ist. In diesem Fall werden Sie aufgefordert, das zu verwendenden XULE-Dokument oder das ZIP-Archiv der XULE-Dateien (das XULE Ruleset) auszuwählen.

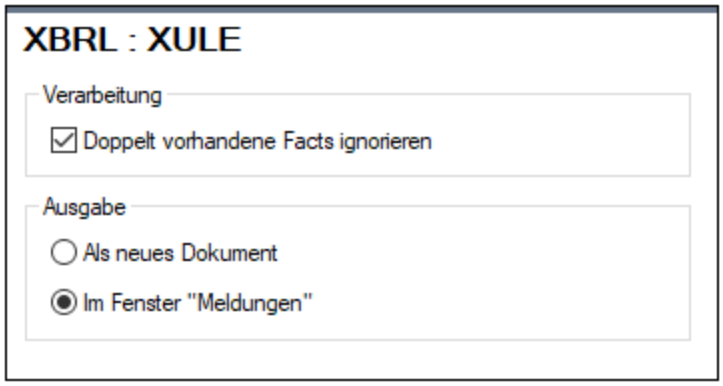
Anmerkung: Wenn das XULE-Dokument und das XBRL-Instanzdokument beide Teil eines XMLSpy-Projekts 1080 sind, können Sie die XBRL-Zielinstanzdatei in den Eigenschaften des XMLSpy-Projekts definieren 1344. Wenn Sie anschließend mit der rechten Maustaste auf die XULE-Datei klicken und den Befehl **XULE ausführen** auswählen, wird das XULE-Dokument an dem als Ziel für die XULE-Ausführen definierten XBRL-Dokument ausgeführt.

© 2016-2022 Altova GmbH

Ausführungsoptionen

XULE-Ausgabe

Die Ausgabe der XULE-Ausführung wird entweder (i) an das Fenster "Meldungen" oder (ii) ein neues in einem neuen XMLSpy-Fenster angezeigtes Dokument gesendet und temporär im Arbeitsspeicher gespeichert. Dieses Dokument kann mit dem Befehl **Datei | Speichern unter**[1286] in einer Datei gespeichert werden. Um festzulegen, ob das Resultat in einem Fenster oder einem neuen Dokument ausgegeben werden soll, wählen Sie die gewünschte Option auf dem XBRL XULE-Optionsregsiter[1644] (**Extras | Optionen | XBRL | XULE**) aus; *siehe Abbildung unten*.



Doppelt vorhandene Facts

Doppelt vorhandene Facts sind mehrere Referenzen auf dasselbe Fact. Sie können doppelt vorhandene Facts im Ergebnis vermeiden, indem Sie auf dem XBRL XULE-Optionsregsiter[1644] die Option *Doppelt vorhandene Facts ignorieren* (**Extras | Optionen | XBRL | XULE**) aktivieren; siehe Abbildung oben. In diesem Fall werden doppelt vorhandene Facts nur einmal ausgegeben.

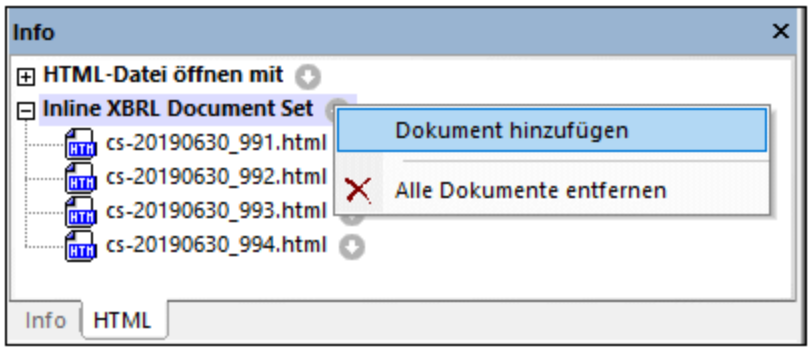
28.16.16 Inline XBRL transformieren

Die Befehle **Inline XBRL transformieren** ist aktiv, wenn in der Text-, Grid- oder Browser-Ansicht ein XHTML-Dokument, das Inline XBRL enthält, das aktive Dokument ist. Dieser Befehl extrahiert die Inline XBRL-Daten aus dem aktiven XHTML-Dokument und generiert ein XBRL-Dokument, das die extrahierten Daten enthält. Das generierte XBRL-Dokument wird in einem neuen Fenster geöffnet und kann in einer Datei gespeichert werden. Damit dieser Befehl ordnungsgemäß funktioniert, müssen alle vom Inline XBRL-Dokument referenzierten Ressourcen für die Verarbeitung zur Verfügung stehen.

Anmerkung: Im Dialogfeld XBRL-Verarbeitungsoptionen[1555], das mit dem Menübefehl **XBRL | Verarbeitungsoptionen**[1555] aufgerufen wird, finden Sie eine Einstellung zum Ignorieren von Duplikaten.

Verarbeiten mehrerer Inline XBRL-Dokumente

Sie können mehrere Inline XBRL-Dokumente verarbeiten, indem Sie die zusätzlichen Inline XBRL-Dokumente zum Inline XBRL Document Set des HTML-Registers des Info-Fensters[120] *(siehe Abbildung unten)* hinzufügen. Beachten Sie, dass dieses Register im Info-Fenster nur angezeigt wird, wenn ein HTML-Dokument im Hauptfenster aktiv ist.



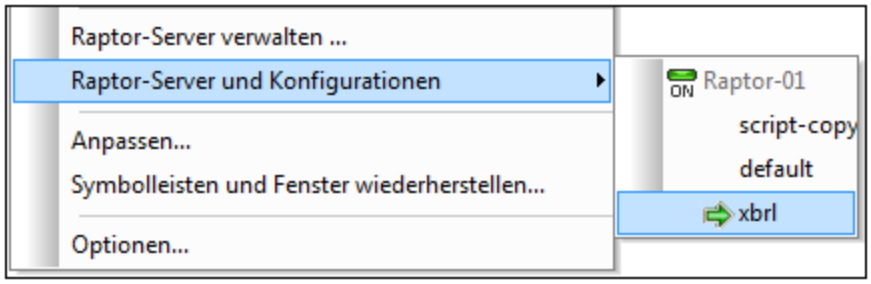
Klicken Sie auf die Menüschaltfläche von *Inline XBRL Document Set,* anschließend auf **Dokument hinzufügen** *(siehe Abbildung)* und navigieren Sie zu den gewünschten Inline XBRL-Dateien. Wenn Sie den Befehl **Inline XBRL transformieren** ausführen, werden die aktive Datei sowie die Dateien des Document Set verarbeitet. Die extrahierten Inline XBRL-Daten aus allen diesen Dateien werden zu einem einzigen XBRL-Dokument kombiniert, das in einem neuen Fenster geöffnet wird.

28.16.17 EDGAR am Server validieren

Mit dem Befehl **EDGAR am Server validieren (Hochleistungsvalidierung)** wird das aktive XBRL-Dokument unter Verwendung des derzeit aktiven RaptorXML+XBRL Servers[1583] und seiner aktiven Konfiguration[1583] validiert. Wenn Sie eine Validierung über EDGAR ausführen, validiert Raptor das XBRL-Instanzdokument mit Hilfe eines internen EDGAR-Skripts[1106]. Mit dem Befehl wird die Validierung sofort durchgeführt und die Ergebnisse werden im Fenster "Meldungen" angezeigt.

Anmerkung: Die tatsächliche Leistung hängt von der Anzahl der PC-Prozessorkerne ab, die von RaptorXML+XBRL Server für die Validierung verwendet werden: Je mehr Kerne verwendet werden, desto schneller ist die Verarbeitung.

Wenn Sie mehrere Konfigurationen auf mehreren Servern definiert haben, können Sie einen Server und eine seiner Konfigurationen als aktive Konfiguration auswählen. Die aktive Konfiguration wird in der Folge für Validierungen verwendet. Wenn Sie den Cursor über den Befehl **Extras | Raptor-Server und Konfigurationen** (*siehe Abbildung unten*) platzieren, wird ein Untermenü mit allen hinzugefügten Servern und deren Konfiguration angezeigt. Wählen Sie die Konfiguration, die Sie zur aktiven Konfiguration machen wollen, aus. In der Abbildung unten wurde die Konfiguration `xbrl` des Servers mit dem Namen `Raptor-01` als aktive Konfiguration ausgewählt (gekennzeichnet durch einen grünen Pfeil).



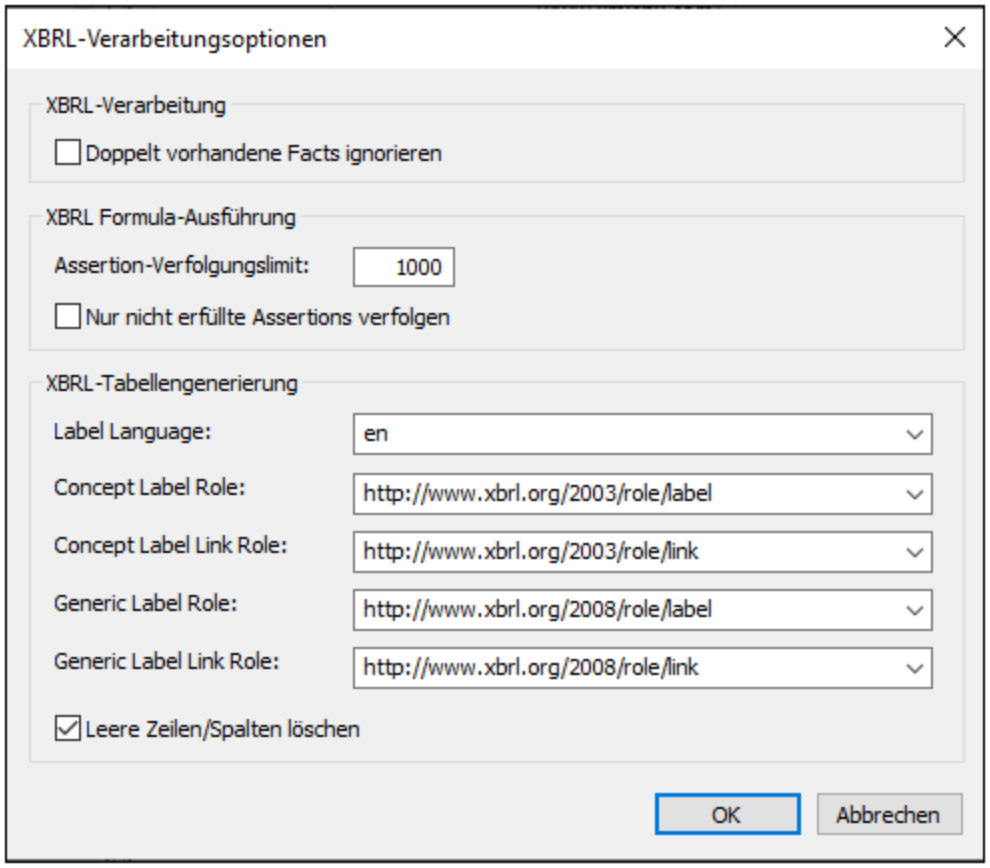
Nähere Informationen dazu finden Sie im Abschnitt RaptorXML(+XBRL) Server[1088].

 © 2016-2022 Altova GmbH

28.16.18 Verarbeitungsoptionen

Mit dem Befehl **XBRL-Verarbeitungsoptionen** wird das Dialogfeld "XBRL-Verarbeitungsoptionen" (*Abbildung unten*) aufgerufen. Hier können Sie folgende Einstellungen vornehmen:

- dass doppelt vorhandene Facts in folgenden Fällen ignoriert werden: (i) bei der XBRL-Formula-Ausführung, (ii) bei der XBRL-Tabellengenerierung, (iii) bei Inline XBRL-Transformationen
- bei der Ausführung von XBRL-Formulas: (i) das Nachverfolgungslimit für Assertions und (ii) ob nur nicht erfüllte Assertions verfolgt werden sollen oder ob sowohl erfüllte als auch nicht erfüllte Assertions verfolgt werden sollen.
- die Label-Einstellungen für die Generierung von XBRL-Tabellen
- dass leere Zeilen und Spalten bei der Generierung von XBRL-Tabellen entfernt werden.



Anmerkung: Das Dialogfeld "XBRL-Verarbeitungsoptionen" kann auch über die Dialogfelder XBRL Formula-Ausführung[1546] und XBRL-Tabellengenerierung[1549] aufgerufen werden.

Anmerkung zur Entfernung von Duplikaten

Es werden nur die folgenden Duplikatarten ignoriert: (i) vollständige Duplikate und (ii) konsistente Duplikate. Das Fact (aus den Duplikaten), das ausgewählt (und nicht ignoriert) wird, ist das genaueste. So wird z.B. aus

konsistenten Duplikaten eines numerischen Fact das Fact mit der höchsten numerischen Genauigkeit ausgewählt.

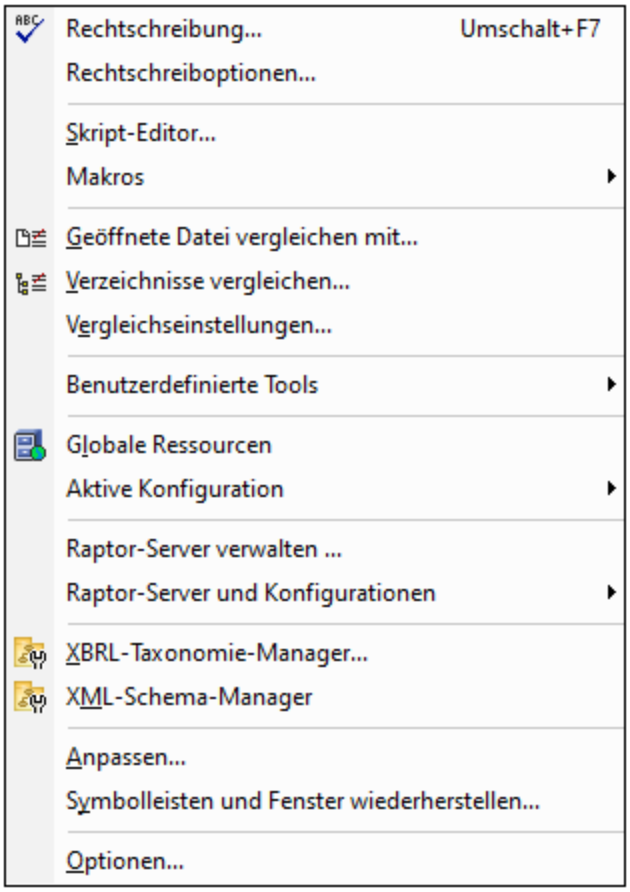
Nähere Informationen zu Duplikaten finden Sie in der folgenden Spezifikation: Handling Duplicate Facts in XBRL and Inline XBRL 1.0.

© 2016-2022 Altova GmbH

28.17 Menü "Extras"

Im Menü "Extras" können Sie:

- Die Rechtschreibung[1558] Ihrer XML-Dokumente überprüfen
- in die Skripting-Umgebung[1662] von XMLSpy wechseln. Sie können Ihre eigenen Formulare, Makros und Event Handler erstellen, bearbeiten und speichern.
- Derzeit zugewiesene Makros ansehen[1567]
- zwei beliebige Dateien vergleichen und auf Unterschiede überprüfen
- zwei beliebige Ordner vergleichen und auf Unterschiede überprüfen
- benutzerdefinierte Befehle aufrufen, die auf externe Applikationen zugreifen. Diese Befehle können im Dialogfeld "Anpassen" im Abschnitt "Extras"[1588] erstellt werden..
- globale Ressourcen definieren[1578]
- die aktive Konfiguration für globale Ressourcen[1579] in XMLSpy wechseln
- RaptorXML-Server[1580] für die XML- und XBRL-Validierung hinzufügen und die RaptorXML-Validierungsoptionen konfigurieren[1581]
- Eine Raptor-Server-Konfiguration als aktive Konfiguration auswählen[1583]
- Ihre XBRL-Taxonomiepakete über eine eigene Applikation, den Taxonomie-Manager[1583] verwalten
- Ihre Version von XMLSpy anpassen[1584]: eigene Menüleisten, Tastaturkürzel, Menüs und Makros erstellen
- Globale XMLSpy-Einstellungen[1602] definieren



28.17.1 Rechtschreibung

Die Rechtschreibprüfung von XMLSpy mit integrierten Wörterbüchern (*siehe Anmerkung unten*) kann in der Text-Ansicht, der Grid-Ansicht und der Authentic-Ansicht verwendet werden. Wenn Sie die Rechtschreibung eines Dokuments, das Sie in einer anderen Ansicht bearbeitet haben, überprüfen möchten, können Sie in die Text- oder die Grid-Ansicht wechseln und die Rechtschreibprüfung ausführen. Wenn Sie z.B. ein XML-Schema-Dokument in der Schema-Ansicht bearbeitet haben, wechseln Sie in die Textansicht oder die Grid-Ansicht um die Rechtschreibprüfung durchzuführen.

Anmerkung: Die in die Altova-Software integrierten Wörterbücher sind nicht Ausdruck einer Präferenz für bestimmte Sprachen durch Altova. Diese Auswahl basiert lediglich auf der Verfügbarkeit von Wörterbüchern, die mit kommerzieller Software im Rahmen von Lizenzen wie z.B. MPL-, LGPL- oder BSD-Lizenzen weitergegeben werden dürfen. Es gibt noch viele weitere Open Source-Wörterbücher, doch werden diese mit größeren Lizenzeinschränkungen wie z.B. im Rahmen von GPL Lizenzen weitergegeben. Viele dieser Wörterbücher stehen als Teil eines separaten Installationsprogramms unter http://www.altova.com/dictionaries zur Verfügung. Sie sollten die gewünschten Wörterbücher auf Basis der dazugehörigen Lizenz und ihrer Eignung für Ihre Zwecke auswählen.

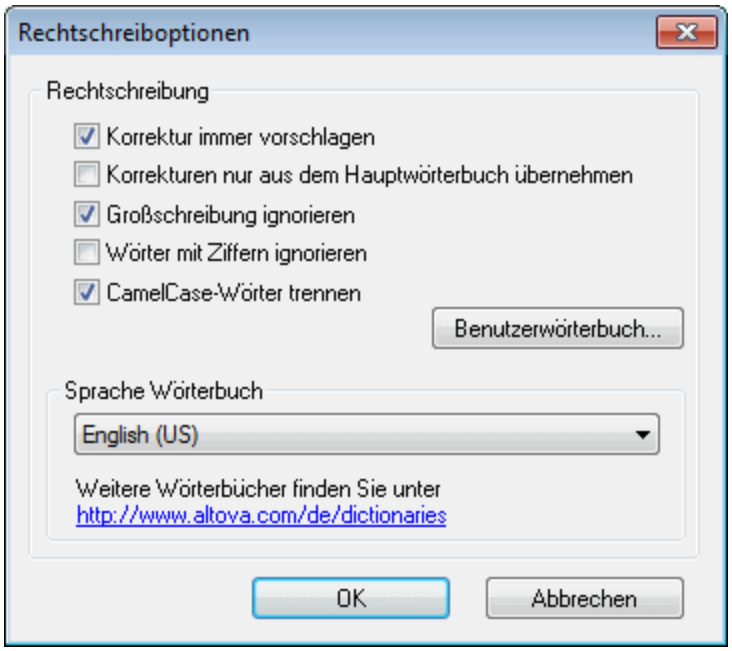
In diesem Abschnitt finden Sie eine Anleitung zur Verwendung der Rechtschreibprüfung. Der Abschnitt ist in die folgenden Unterabschnitte gegliedert:

- Auswahl der Sprache für die Rechtschreibprüfung 1558
- Definieren des Rahmens der Rechtschreibprüfung 1559
- Ausführen der Rechtschreibprüfung 1560

Auswahl der Sprache für die Rechtschreibprüfung

Die Sprache für die Rechtschreibprüfung wird folgendermaßen festgelegt:

1. Wählen Sie den Menübefehl **Extras | Rechtschreiboptionen**.
2. Wählen Sie im nun angezeigten Dialogfeld "Rechtschreiboptionen" (*Abbildung unten*) eines der installierten Wörterbücher aus der Dropdown-Liste des Auswahlfelds "Sprache Wörterbuch" aus.

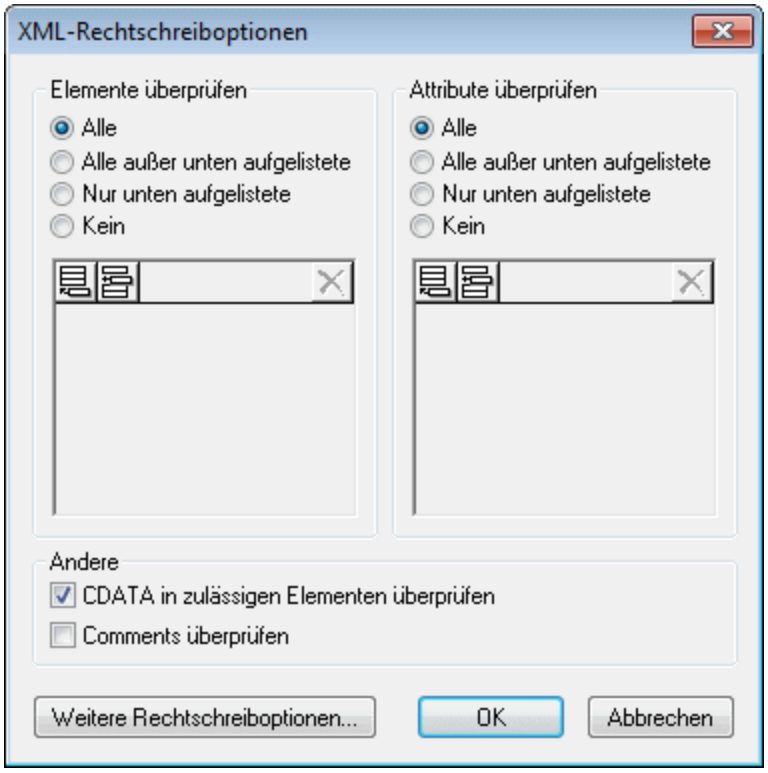


3. Klicken Sie abschließend auf **OK**.

Die ausgewählte Wörterbuchsprache wird von der Rechtschreibprüfung verwendet. Wenn die gewünschte Sprache noch nicht installiert ist, können Sie zusätzliche Wörterbücher herunterladen. Eine Anleitung dazu finden Sie im Abschnitt Hinzufügen von Wörterbüchern zur Rechtschreibprüfung[1564].

Definieren des Rahmens der Rechtschreibprüfung

Bei Ausführung der Rechtschreibprüfung in der Textansicht oder der Grid-Ansicht kann der Rahmen für die Rechtschreibprüfung unmittelbar vor dem Start der Rechtschreibprüfung definiert werden. Wählen Sie dazu den Befehl **Extras | Rechtschreiboptionen** und definieren Sie den gewünschten Rahmen im nun angezeigten Dialogfeld XML-Rechtschreiboptionen[1562] (*siehe Abbildung unten*).

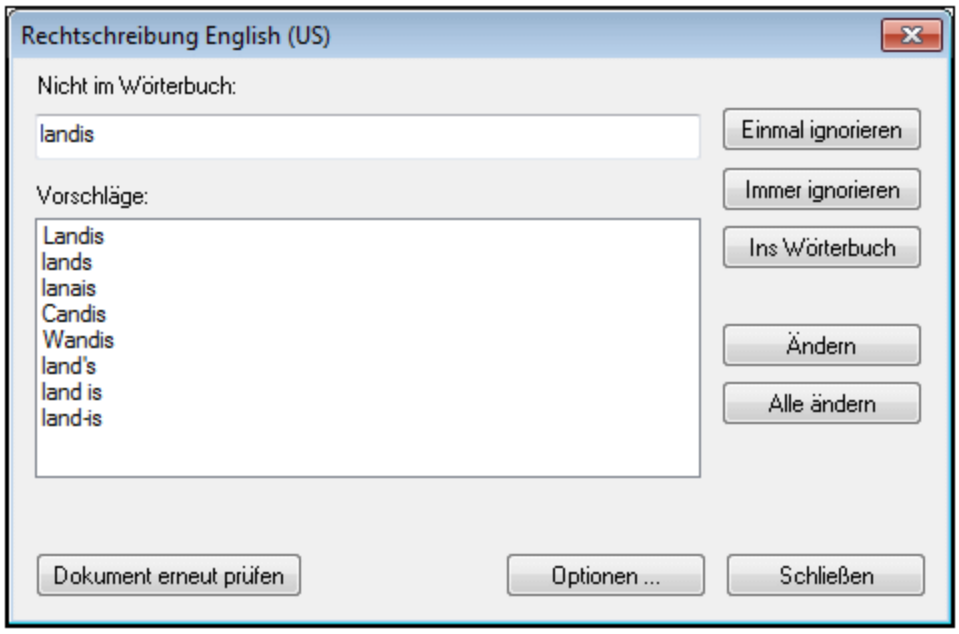


So können Sie zum Beispiel auswählen, welche Elemente und Attribute überprüft werden sollen und ob CDATA-Abschnitte und Kommentare bei der Rechtschreibprüfung berücksichtigt werden sollen. Informationen dazu finden Sie auch in der Beschreibung zum Befehl Rechtschreiboptionen[1562].

Ausführen der Rechtschreibprüfung

Bei Klicken auf den Befehl **Extras | Rechtschreibung (Umschalt+F7)** wird die Rechtschreibung des gerade aktiven XML-Dokuments automatisch gemäß dem definierten Überprüfungsrahmen[1559] überprüft. Wenn ein unbekanntes Wort gefunden wird, erscheint das Dialogfeld *Rechtschreibung: Nicht im Wörterbuch* (*Abbildung unten*). Andernfalls wird das Dokument weiter bis zum Ende überprüft.

© 2016-2022 Altova GmbH



Die verschiedenen Bereiche des Dialogfelds *Rechtschreibung: Nicht im Wörterbuch* und die verfügbaren Optionen werden weiter unten beschrieben:

Nicht im Wörterbuch
Dieses Textfeld enthält das Wort, das weder im ausgewählten Wörterbuch noch im Benutzerwörterbuch gefunden wurde. Es stehen die folgenden Optionen zur Verfügung:

- Sie können das Wort im Textfeld manuell bearbeiten oder einen Vorschlag aus dem Bereich *Vorschläge* auswählen. Klicken Sie anschließend auf **Ändern**, um das Wort im XML-Dokument durch das bearbeitete Wort zu ersetzen. (Durch Doppelklick auf ein vorgeschlagenes Wort wird dieses direkt ins XML-Dokument eingefügt). Ein Wort, das im Textfeld *Nicht im Wörterbuch* angezeigt wird, wird im XML-Dokument markiert anzeigt, sodass Sie das Wort auch direkt im Dokument bearbeiten können. Bei Auswahl der Schaltfläche **Alle ändern** werden alle Instanzen des Worts im XML-Dokument durch das geänderte Wort ersetzt.
- Sie können auch keine Änderung vornehmen und die Warnung der Rechtschreibprüfung ignorieren - und zwar entweder nur für die aktuelle Instanz des Worts oder für alle Instanzen.
- Sie können das Wort zum Benutzerwörterbuch hinzufügen, sodass das Wort in Zukunft bei jeder Rechtschreibprüfung als korrekt erkannt wird.

Vorschläge
Dieses Listenfeld zeigt Wörter an, die Ähnlichkeiten mit dem unbekannten Wort haben (die Wörter stammen aus dem Wörterbuch und Benutzerwörterbüchern). Durch Doppelklicken auf einen Eintrag wird dieser anstelle des unbekannten Wortes eingefügt und die Rechtschreibprüfung wird fortgesetzt.

Einmal ignorieren
Mit diesem Befehl können Sie das unbekannte Wort bei seinem ersten Auftreten ignorieren und die Rechtschreibprüfung fortsetzen. Sollte das Wort ein weiteres Mal im Dokument vorkommen, wird die Rechtschreibprüfung es dann wieder als unbekannt markieren.

Immer ignorieren
Mit diesem Befehl werden alle Instanzen des unbekannten Worts im gesamten Dokument ignoriert.

Ins Wörterbuch
Mit diesem Befehl können Sie das unbekannte Wort in das **Benutzerwörterbuch** einfügen. Das Benutzerwörterbuch kann über das Dialogfeld Rechtschreiboptionen[1566] aufgerufen und dort bearbeitet werden..

Ändern
Mit diesem Befehl ersetzen Sie das im XML-Dokument markierte Wort durch das bearbeitete Wort aus dem Textfeld *Nicht im Wörterbuch*.

Alle ändern
Mit diesem Befehl ersetzen Sie das Wort an allen Stellen im XML-Dokument, an denen es vorkommt, durch das bearbeitete Wort aus dem Textfeld *Nicht im Wörterbuch*.

Dokument erneut prüfen
Damit wird das Dokument von Anfang an neu überprüft.

Optionen
Über diese Schaltfläche öffnen Sie ein Dialogfeld in der aktuellen Ansicht:

- Bei Aufruf über die Authentic-Ansicht wird das Dialogfeld Rechtschreiboptionen[1562] geöffnet.
- In der Text oder Grid-Ansicht wird das Dialogfeld XML-Rechtschreiboptionen[1562] geöffnet.

Nähere Informationen dazu finden Sie auch in der Beschreibung zum Befehl Rechtschreiboptionen[1562].

Schließen
Mit diesem Befehl schließen Sie das Dialogfeld "Rechtschreibung".

28.17.2 Rechtschreiboptionen

Mit dem Befehl **Extras | Rechtschreiboptionen** öffnen Sie das Dialogfeld **Rechtschreiboptionen**[1563]. Je nachdem, welche Ansicht aktiv ist, wird entweder direkt das Dialogfeld **Rechtschreiboptionen**[1563] (Schema-Ansicht, WSDL-Ansicht, XBRL-Ansicht, Authentic-Ansicht, Browser-Ansicht) oder das Dialogfeld **XML-Rechtschreiboptionen**[1562] (Textansicht, Grid-Ansicht, usw.) geöffnet. Das Dialogfeld "XML-Rechtschreiboptionen" hat eine Schaltfläche **Rechtschreiboptionen**, um das Dialogfeld "Rechtschreiboptionen" aufzurufen.

Die verschiedenen Einstellungen, die in diesen beiden Dialogfeldern zur Verfügung stehen, werden in den Unterabschnitten dieses Abschnitts beschrieben:

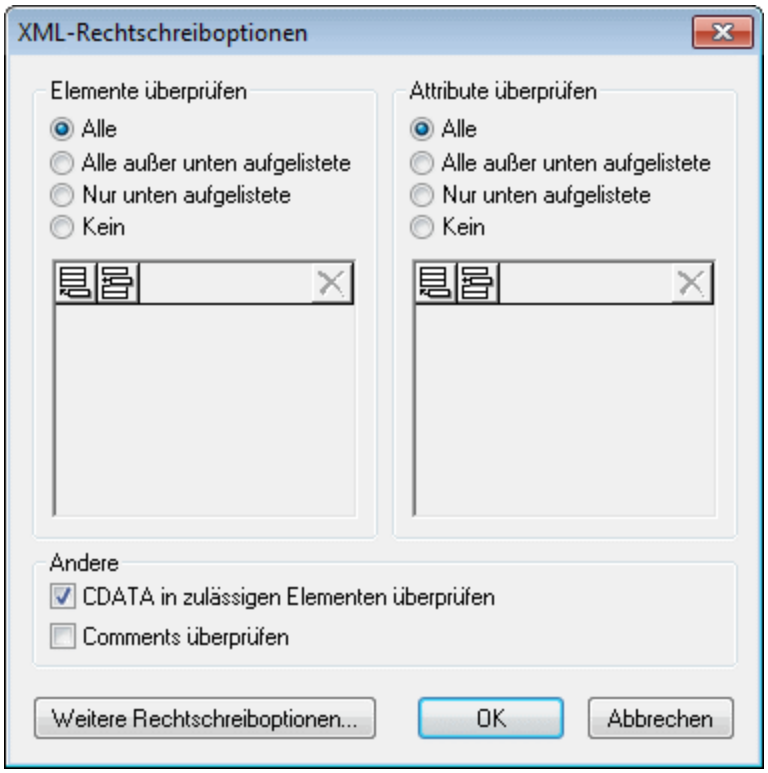
- Rechtschreibkontext[1562]
- Rechtschreiboptionen[1563]
- Hinzufügen von Wörterbüchern für die Rechtschreibprüfung[1564]
- Arbeiten mit dem Benutzerwörterbuch[1566]

Dialogfeld "XML-Rechtschreiboptionen"

Wenn Sie in der Text-Ansicht oder der Grid-Ansicht den Befehl **Extras | Rechtschreiboptionen** auswählen, wird das Dialogfeld "XML-Rechtschreiboptionen" (*Abbildung unten*) aufgerufen. Hier können Sie auswählen, in welchem Bereich die Rechtschreibprüfung durchgeführt werden soll. So können Sie zum Beispiel auswählen,

© 2016-2022 Altova GmbH

welche Elemente und Attribute überprüft werden sollen und ob CDATA-Abschnitte und Kommentare bei der Rechtschreibprüfung berücksichtigt werden sollen.

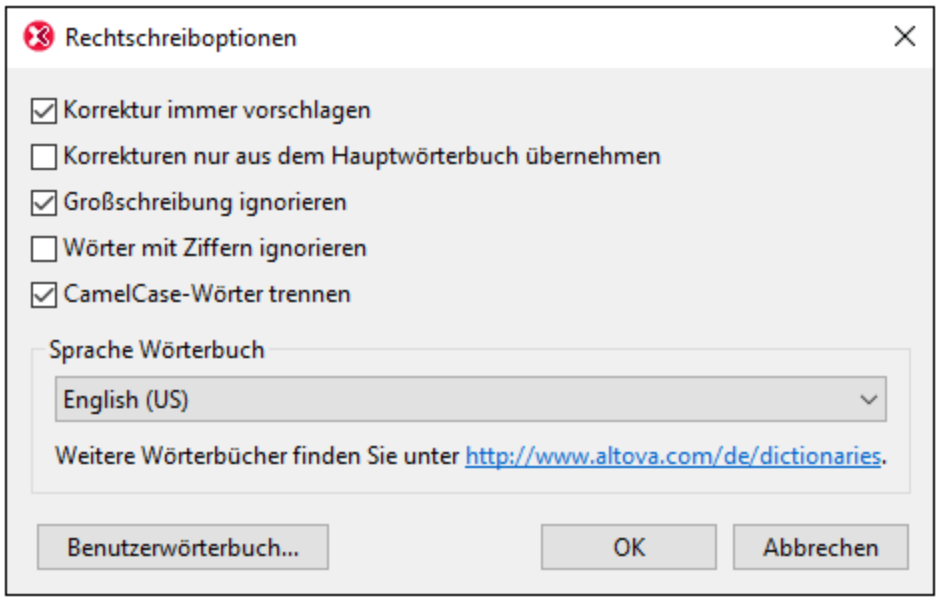


Sie können eine Liste von Elementen und/oder Attributen zusammenstellen, deren Rechtschreibung überprüft werden soll bzw. die bei der Rechtschreibprüfung nicht berücksichtigt werden sollen. Alternativ dazu können Sie die Rechtschreibprüfung an allen Elementen und/oder Attributen oder an keinem Element oder Attribut durchführen.

Wenn Sie am unteren Rand des Dialogfelds auf die Schaltfläche **Weitere Rechtschreiboptionen** klicken, wird das Dialogfeld "Rechtschreiboptionen" geöffnet.

Rechtschreiboptionen

Im Dialogfeld "Rechtschreiboptionen" können Sie globale Optionen für die Rechtschreibprüfung definieren.



Korrektur immer vorschlagen:
Wenn Sie dieses Kontrollkästchen aktivieren, werden die Vorschläge (sowohl aus dem Wörterbuch als auch aus dem Benutzerwörterbuch) in der Vorschlagsliste angezeigt. Wenn Sie diese Option deaktivieren, werden keinen Vorschläge angezeigt.

Korrekturen nur aus dem Hauptwörterbuch übernehmen:
Bei Aktivierung dieser Option werden nur Vorschläge angezeigt, die im Hauptwörterbuch zur Verfügung stehen. Das Benutzerwörterbuch wird nicht durchsucht. Die Schaltfläche **Benutzerwörterbuch** ist deaktiviert, sodass das Benutzerwörterbuch nicht bearbeitet werden kann.

Großschreibung ignorieren:
Großgeschriebene Wörter werden in der Rechtschreibprüfung ignoriert.

Wörter mit Ziffern ignorieren:
Wörter, die Ziffern enthalten, werden in der Rechtschreibprüfung ignoriert.

CamelCase-Wörter trennen
CamelCase-Wörter sind Wörter, die einen Großbuchstaben innerhalb des Worts haben. So ist z.B. im Wort "CamelCase" das "C" von "Case" groß geschrieben, d.h. es handelt sich hierbei um ein CamelCase-Wort. Da man derartige Wörte selten in einem Wörterbuch findet, würden Sie von der Rechtschreibprüfung als Fehler markiert. Um dies zu vermeiden, werden CamelCase-Wörter mit dieser Option in ihre beiden Komponenten aufgeteilt, sodass die Komponenten einzeln überprüft werden können. Diese Option ist standardmäßig aktiviert.

Sprache Wörterbuch
Wählen Sie im Dropdown-Listenfeld eine Sprache, in der die Rechtschreibprüfung erfolgen soll. Die Standardeinstellung ist US-Englisch. Wörterbücher in weiteren Sprachen können kostenlos von der Altova Website heruntergeladen werden.

Hinzufügen von Wörterbüchern für die Rechtschreibprüfung

Für jede Wörterbuchsprache stehen zwei Hunspell-Wörterbuchdateien zur Verfügung, die miteinander verwendet werden: eine `.aff` Datei und eine `.dic` Datei. Alle Wörterbücher werden unter dem folgenden Pfad

 © 2016-2022 Altova GmbH

im Ordner `Lexicons` installiert: `C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons`.

Im Ordner `Lexicons` werden unterschiedliche Sprachwörterbücher in jeweils anderen Ordnern gespeichert: `<Sprachenname>\<Wörterbuchdateien>`. So werden z.B. Dateien für die beiden englischsprachigen Wörterbücher (`English (British)` und `English (US)`) folgendermaßen gespeichert:

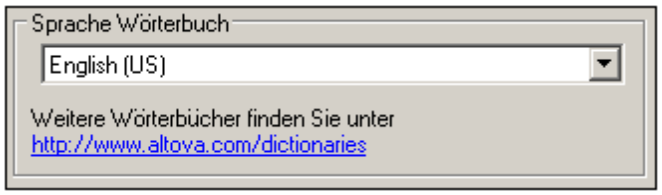
```
C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons\English (British)
\en_GB.aff
C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons\English (British)
\en_GB.dic
C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons\English (US)\en_US.aff
C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons\English (US)\en_US.dic
```

Die Sprachwörterbücher werden im Dialogfeld "Rechtschreiboptionen" in der Dropdown-Liste der Auswahlliste *Sprache Wörterbuch* aufgelistet. Diese Wörterbücher sind diejenigen, die im Ordner `Lexicons` zur Verfügung stehen, und haben dieselben Namen wie die Unterordner für die jeweilige Sprache im Ordner `Lexicons`. So würden z.B. im Fall der oben angegebenen englischsprachigen Wörterbücher die Wörterbücher in der Auswahlliste "Sprache Wörterbuch" als *English (British)* und *English (US)* angezeigt werden.

Alle installierten Wörterbücher stehen für alle Benutzer auf dem Rechner und alle Altova-Produktversionen (ob 32-Bit oder 64-Bit) zur Verfügung

Sie können Wörterbücher für die Rechtschreibprüfung auf zwei Arten hinzufügen, wobei die Dateien in keinem Fall im System registriert werden müssen:

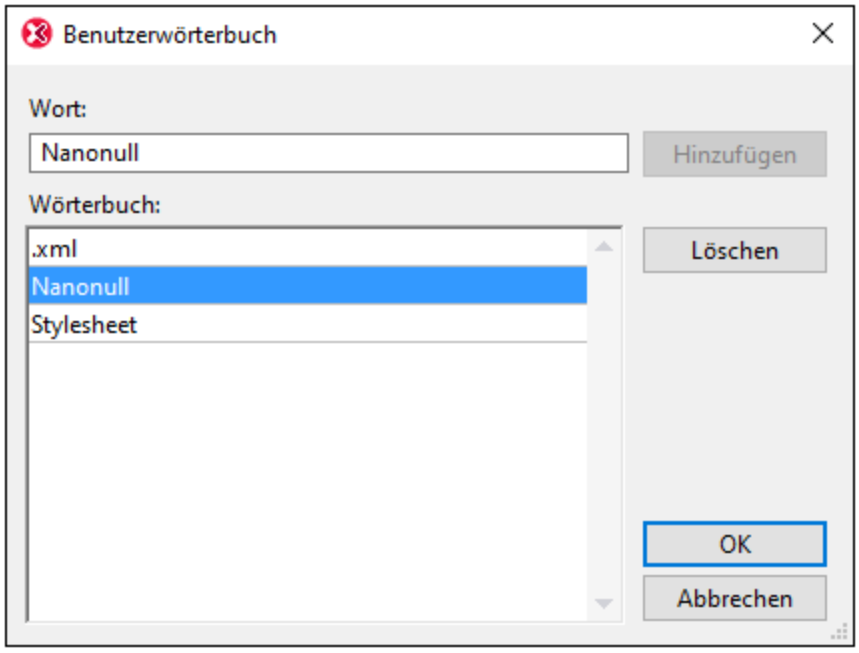
- Durch Hinzufügen von Hunspell-Wörterbüchern in einem neuen Unterordner des Ordners `Lexicons`. Hunspell-Wörterbücher können z.B. von https://wiki.services.openoffice.org/wiki/Dictionaries oder http://extensions.services.openoffice.org/en/dictionaries heruntergeladen werden. (Beachten Sie bitte, dass in OpenOffice das Zip-Format `OXT` verwendet wird. Ändern Sie also die Erweiterung in `.zip` und entpacken Sie die `.aff`- und `.dic`-Datei in die Sprachordner im Ordner `Lexicons`. Beachten Sie außerdem, dass Hunspell-Wörterbücher auf Myspell-Wörterbüchern basieren. Daher können auch Myspell-Wörterbücher verwendet werden.)
- Durch Verwendung des Altova-Wörterbuchinstallationsprogramms, das ein Paket mit mehreren Sprachwörterbüchern standardmäßig im richtigen Ordner auf Ihrem Rechner installiert. Das Installationsprogramm kann durch Klicken auf den Link im Bereich "Sprache Wörterbuch" des Dialogfelds "Rechtschreibung Optionen" (*siehe Abbildung unten*) heruntergeladen werden. Für die Installation benötigen Sie Administratorrechte, da die Installation sonst nicht durchgeführt werden kann.



Anmerkung: Es bleibt Ihnen überlassen, ob Sie mit den für das jeweilige Wörterbuch geltenden Lizenzbedingungen einverstanden sind und ob das Wörterbuch sich für die Verwendung mit der auf Ihrem Rechner installierten Software eignet.

Arbeiten mit dem Benutzerwörterbuch

Jeder Benutzer hat ein Benutzerwörterbuch, in dem vom Benutzer genehmigte Wörter gespeichert werden können. Bei der Rechtschreibprüfung wird die Rechtschreibung anhand einer Wörterliste bestehend aus den Wörtern im Sprachwörterbuch und denen im Benutzerwörterbuch durchgeführt. Sie können Wörter über das Dialogfeld "Benutzerwörterbuch" (*Abbildung unten*) zum Benutzerwörterbuch hinzufügen bzw. diese daraus löschen. Dieses Dialogfeld wird durch Klick auf die Schaltfläche "Benutzerwörterbuch" im Dialogfeld "Rechtschreiboptionen" aufgerufen (*siehe zweite Abbildung in diesem Abschnitt*).



Um ein Wort zum Benutzerwörterbuch hinzuzufügen, geben Sie das Wort in das Textfeld "Wort" ein und klicken Sie auf **Hinzufügen**. Daraufhin wird das Wort zur alphabetischen Liste im Fenster "Wörterbuch" hinzugefügt. Um ein Wort aus dem Wörterbuch zu löschen, wählen Sie das Wort im Fenster "Wörterbuch" aus und klicken Sie auf **Löschen**. Daraufhin wird das Wort aus dem Fenster "Wörterbuch" gelöscht. Wenn Sie mit der Bearbeitung des Benutzerwörterbuchs fertig sind, klicken Sie auf **OK**, damit die Änderungen im Benutzerwörterbuch gespeichert werden.

Wörter können auch während einer Rechtschreibprüfung zum Benutzerwörterbuch hinzugefügt werden. Wenn bei der Rechtschreibprüfung ein unbekanntes Wort gefunden wird, wird das Dialogfeld Rechtschreibprüfung[1558] aufgerufen und Sie werden aufgefordert zwischen verschiedenen Aktionen zu wählen. Wenn Sie auf die Schaltfläche **Ins Wörterbuch** klicken, wird das unbekannte Wort zum Benutzerwörterbuch hinzugefügt.

Das Benutzerwörterbuch befindet sich im folgenden Ordner: `C:\Benutzer\<Benutzer>\Dokumente\Altova\SpellChecker\Lexicons\user.dic`

28.17.3 Skript-Editor

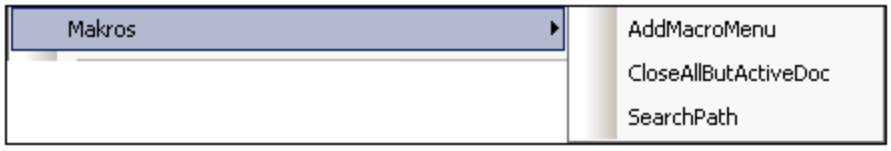
Mit dem Befehl "Skript-Editor" wird das Fenster "Skript-Editor" geöffnet. Eine Beschreibung zur Verwendung des Skript-Editors finden Sie in dieser Dokumentation im Abschnitt Skript-Editor[1662].

© 2016-2022 Altova GmbH

Anmerkung: Zur Ausführung des Skript-Editors muss die .NET Framework Version 2.0 oder höher auf Ihrem Rechner installiert sein.

28.17.4 Makros

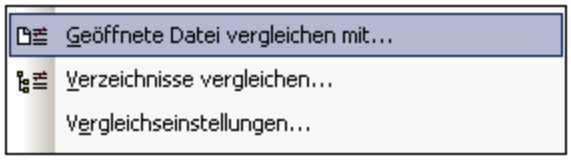
Wenn Sie die Maus über den Befehl **Makros** positionieren, wird ein Untermenü mit den Makros angezeigt, die im Skripting-Projekt, das in XMLSpy gerade aktiv ist, definiert sind (*Abbildung unten*).



Durch Klicken auf ein Makro im Untermenü (*Siehe Abbildung oben*) wird das Makro ausgeführt.

28.17.5 Vergleiche

XMLSpy bietet eine Vergleichs- (oder Differencing)-Funktionalität, mit der XML- und Textdateien sowie Ordner auf Unterschiede überprüft werden können.



Das Menü Extras enthält die folgenden Menüpunkte, mit Hilfe derer Sie Vergleiche an Dateien und Ordnern durchführen können:

- Geöffnete Datei vergleichen mit 1567
- Verzeichnisse vergleichen 1572
- Vergleichseinstellungen 1575

Diese Befehle werden in den folgenden Unterabschnitten näher beschrieben.

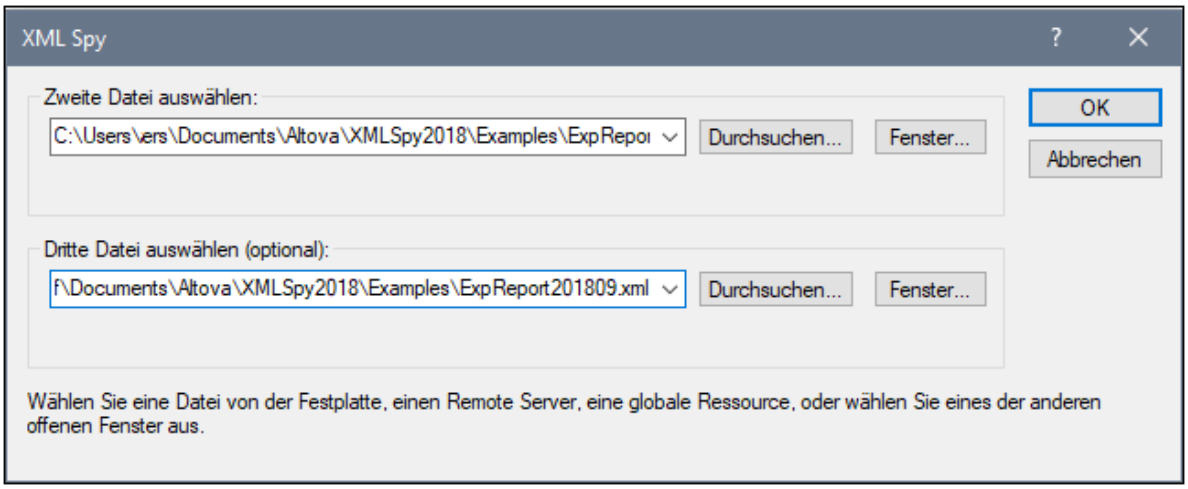
28.17.5.1 Geöffnete Datei vergleichen mit

Mit diesem Befehl können Sie eine geöffnete Datei mit einer anderen Datei vergleichen. Beim Vergleich werden beide Dateien im Hauptfenster senkrecht nebeneinander angezeigt, wobei die Unterschiede in den beiden Dateien markiert sind. Wenn ein Unterschied zwischen zwei Dateien besteht, wird der Unterschied grün markiert. Wenn sich der Inhalt in drei Dateien unterscheidet (*Funktion nur in der Enterprise Edition verfügbar*), so wird dies als Konflikt bezeichnet und der Konflikt wird pink markiert. Dateien können als XML-Dokumente (unter Berücksichtigung der Struktur und Semantik der Tags) oder als Textdokumente verglichen werden.

Anmerkung: Drei-Weg-Vergleiche stehen **nur in der Enterprise Edition** zur Verfügung.

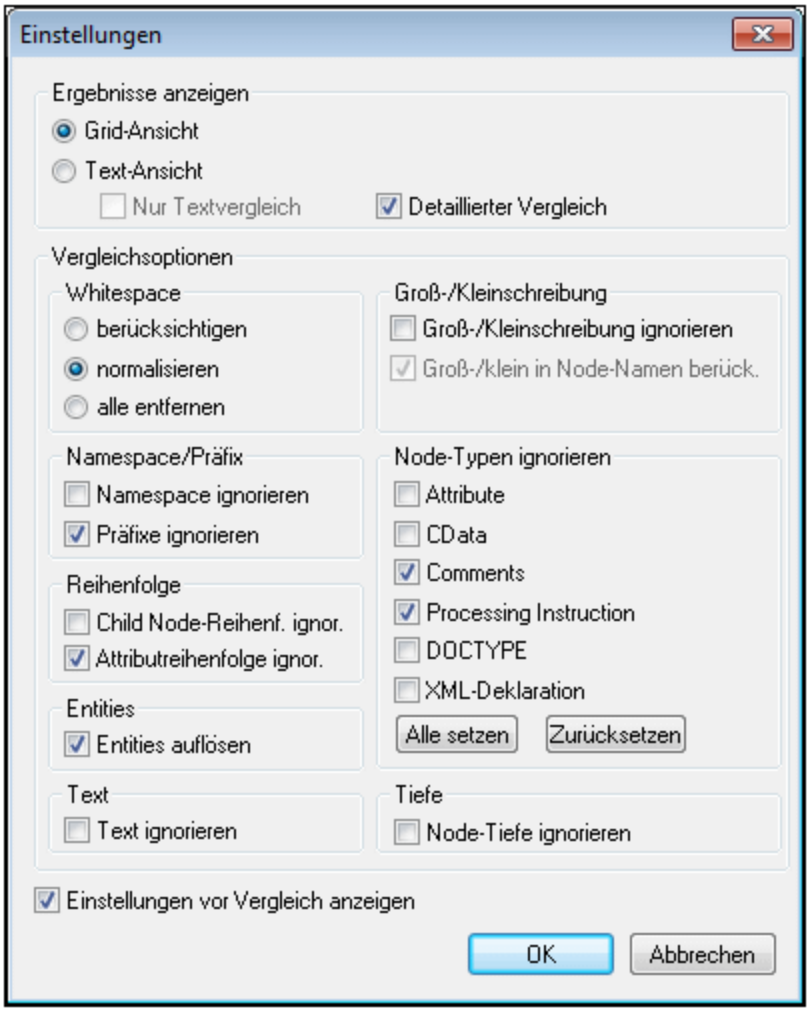
So vergleichen Sie die aktive Datei (auf der Benutzeroberfläche) mit einer anderen Datei:

1. Wenn die zu vergleichende Datei im Hauptfenster geöffnet ist (es kann immer nur jeweils eine Datei aktiv sein), klicken Sie auf **Extras | Geöffnete Datei vergleichen mit...**... . Daraufhin wird ein Durchsuchen-Dialogfeld angezeigt (Abbildung unten), in dem Sie zu einer oder zwei Dateien, die mit der aktiven Datei verglichen werden sollen, navigieren können.



2. Klicken Sie auf die Schaltfläche **Durchsuchen**[1280], um eine Datei über den Datei-Explorer, eine globale Ressource oder eine URL auszuwählen. Klicken Sie auf **Fenster**, um eine Datei auszuwählen, die in einem der Fenster von XMLSpy geöffnet ist.
3. Bestätigen Sie mit **OK**. Daraufhin wird das Dialogfeld "Einstellungen" (*Abbildung unten*) geöffnet. Diese Einstellungen sind im Kapitel Vergleichsoptionen[1575] beschrieben. Wenn dieses Dialogfeld nicht bei jedem Start einer Vergleichssitzung angezeigt werden soll, deaktivieren Sie das Kontrollkästchen *Einstellungen vor Vergleich anzeigen* (am unteren Rand des Dialogfelds).

© 2016-2022 Altova GmbH

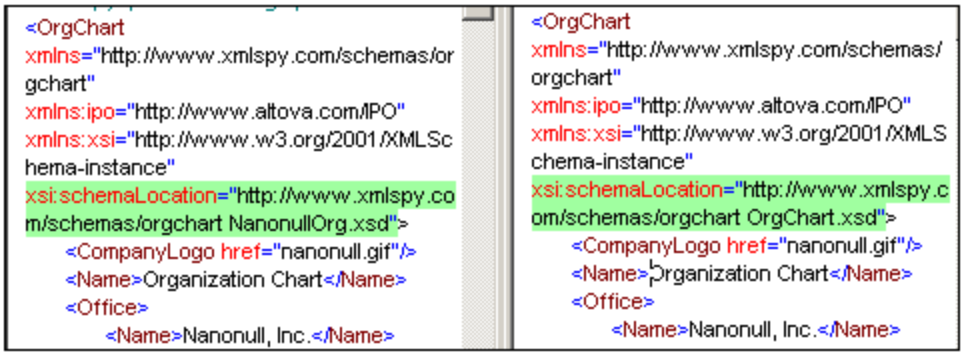


4. Wählen Sie die gewünschten Einstellungen aus und klicken Sie auf **OK**. Es geschehen nun zwei Dinge: (a) Die Dateien werden nun im Hauptfenster nebeneinander in separaten Bereichen angezeigt. (b) Das Steuerungsfenster "Dateien vergleichen" wird angezeigt. (*Abbildungen im nächsten Abschnitt, Dateivergleiche*[1569]).
5. Mit Hilfe der Schaltflächen im Fenster "Dateien vergleichen" können Sie zwischen den Unterschieden navigieren und diese zusammenführen. (*Eine Anleitung dazu finden Sie im nächsten Abschnitt Dateivergleiche*[1569]).
5. Nach Abschluss der Überprüfung und/oder des Zusammenführens von Unterschieden, klicken Sie auf "**Fertig**".

Dateivergleiche und Zusammenführen von Unterschieden

Es gibt zwei Arten von Vergleichen:

- *Zwei-Weg-Vergleich:* Es werden zwei Dateien verglichen. Die Datei, die (bei Start des Vergleichs; kurz "aktive Datei") aktiv war, wird im linken Bereich angezeigt und mit einer zweiten Datei (*siehe Abbildung unten*) verglichen.



- *Drei-Weg-Vergleich (Nur Enterprise Edition):* Es werden drei Dateien verglichen. Die aktive Datei befindet sich im linken Bereich. Die erste der beiden für den Vergleich mit der aktiven Datei ausgewählten Dateien wird im mittleren Bereich angezeigt. Die mittlere Datei wird als die Basisdatei bezeichnet. Sie können einen Unterschied zwischen der linken Datei und der Basisdatei oder zwischen der rechten Datei und der Basisdatei zusammenführen.

Anmerkung: Sie sollten Bereiche im Fenster nicht verschieben, da sonst die "Kopieren nach" Richtungen der Schaltflächen im Steuerungsfenster "Dateien vergleichen" eventuell nicht mehr stimmen.

Unterschiede und Konflikte

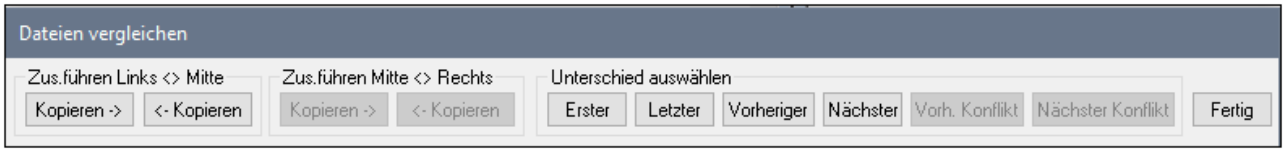
Unterschiede und Konflikte werden in unterschiedlichen Farben markiert (*siehe* Markierungsfarben[1571] *unten*). Die Navigation darin erfolgt separat.

- *Unterschiede:* Wenn der Inhalt an den entsprechenden Stellen in den beiden Dateien verschieden ist. Wird grün markiert.
- *Konflikte:* Wenn der Inhalt an den entsprechenden Stellen in drei Dateien verschieden ist. Wird pink markiert.

Anmerkung: In Zwei-Weg-Dateien werden nur Unterschiede, keine Konflikte angezeigt.

Navigieren zwischen Unterschieden und Konflikten

Mit Hilfe der Schaltflächen im Bereich "Unterschied auswählen" des Steuerungsfensters "Dateien vergleichen" können Sie durch das Dokument navigieren (*Abbildungen unten: Zwei-Weg-Vergleich, gefolgt von Drei-Weg-Vergleich)*. Der Bereich enthält für den Zwei-Weg-Vergleich vier und für den Drei-Weg-Vergleich sechs Schaltflächen (*siehe Abbildungen*).



Das Steuerungsfenster "Dateien vergleichen" enthält die folgenden Schaltflächen:

- *Erster:* Geht zum ersten der Unterschiede und Konflikte im Dokument.
- *Letzter:* Geht zum letzten der Unterschiede und Konflikte im Dokument.
- *Vorheriger:* Geht vom aktuell aktiven Unterschied oder Konflikt zum vorherigen Unterschied oder Konflikt.

 © 2016-2022 Altova GmbH

- *Nächster:* Geht vom aktuell aktiven Unterschied oder Konflikt zum nächsten Unterschied oder Konflikt.
- *Vorheriger Konflikt:* Geht vom aktuell aktiven Unterschied oder Konflikt zum vorherigen Konflikt.
- *Nächster Konflikt:* Geht vom aktuell aktiven Unterschied oder Konflikt zum nächsten Konflikt.

Zusammenführen von Unterschieden und Konflikten
Mit Hilfe der Schaltflächen im Bereich *Zusammenführen* können Sie den markierten Inhalt von einem Bereich in einen anderen kopieren. Verwenden Sie dazu die entsprechende Schaltfläche eines Bereichspaars für die Kopierrichtung. Damit eine Zusammenführung erfolgen kann, müssen die folgenden Vergleichsoptionen[1575] definiert sein:

- *Detaillierter Vergleich* muss aktiviert und
- *Node-Tiefe ignorieren* muss deaktiviert sein.

Nachdem Sie die Unterschiede fertig überprüft und/oder zusammengeführt haben, klicken Sie auf **Fertig**.

Anmerkung: Wenn Sie eine Zusammenführung rückgängig machen möchten, stoppen Sie die Vergleichssitzung, wählen Sie die Datei, in der die Änderung rückgängig gemacht werden soll, aus und wählen Sie dem Befehl **Bearbeiten | Rückgängig** oder drücken Sie **Strg+Z**.

Anmerkung: Während einer aktiven Vergleichssitzung ist keine Bearbeitung oder Änderung der Ansichten gestattet. Bei jedem Versuch die Datei zu bearbeiten oder die Ansicht einer Datei zu wechseln wird eine Warnmeldung angezeigt, dass die Vergleichssitzung beendet wird.

Markierungsfarben

Wenn die entsprechenden Zeilen in zwei Dateien unterschiedlich sind, werden sie in beiden Dateien grün markiert (Unterschiede). Wenn Zeilen in drei Dateien unterschiedlich sind, werden Sie in allen drei Dateien pink markiert. Wenn ein Unterschied ausgewählt ist, wird dies in beiden Dateien in einem dunkleren Grün gekennzeichnet. Wenn ein Konflikt ausgewählt ist, wird dieser in beiden Dateien in einem dunkleren Pink gekennzeichnet.

In der Tabelle unten sehen Sie die Farben, die in Vergleichssitzungen verwendet werden.

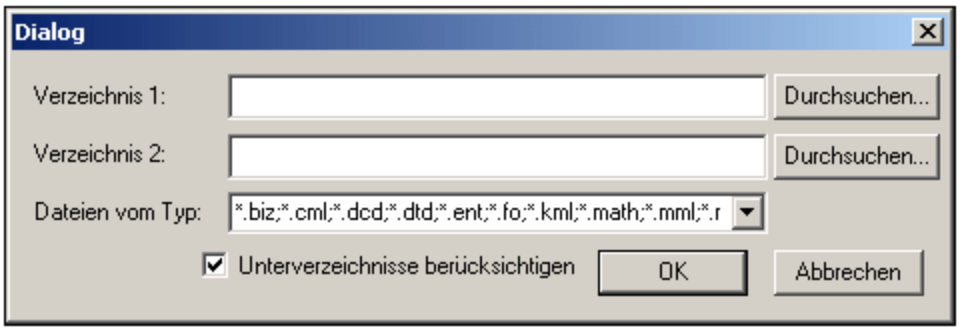
	Unterschied
	Aktueller Unterschied
	Zusammengeführter Unterschied
	Aktueller zusammengeführter Unterschied
	Konflikt
	Aktueller Konflikt

28.17.5.2 Verzeichnisse vergleichen

Mit dem Befehl **Verzeichnisse vergleichen** können Sie zwei Verzeichnisse einschließlich oder ausschließlich ihrer Unterverzeichnisse vergleichen. Der Vergleich von Verzeichnissen dient dazu, festzustellen, welche Dateien fehlen und ob Dateien desselben Namens Unterschiede aufweisen oder nicht.

So vergleichen Sie zwei Verzeichnisse:

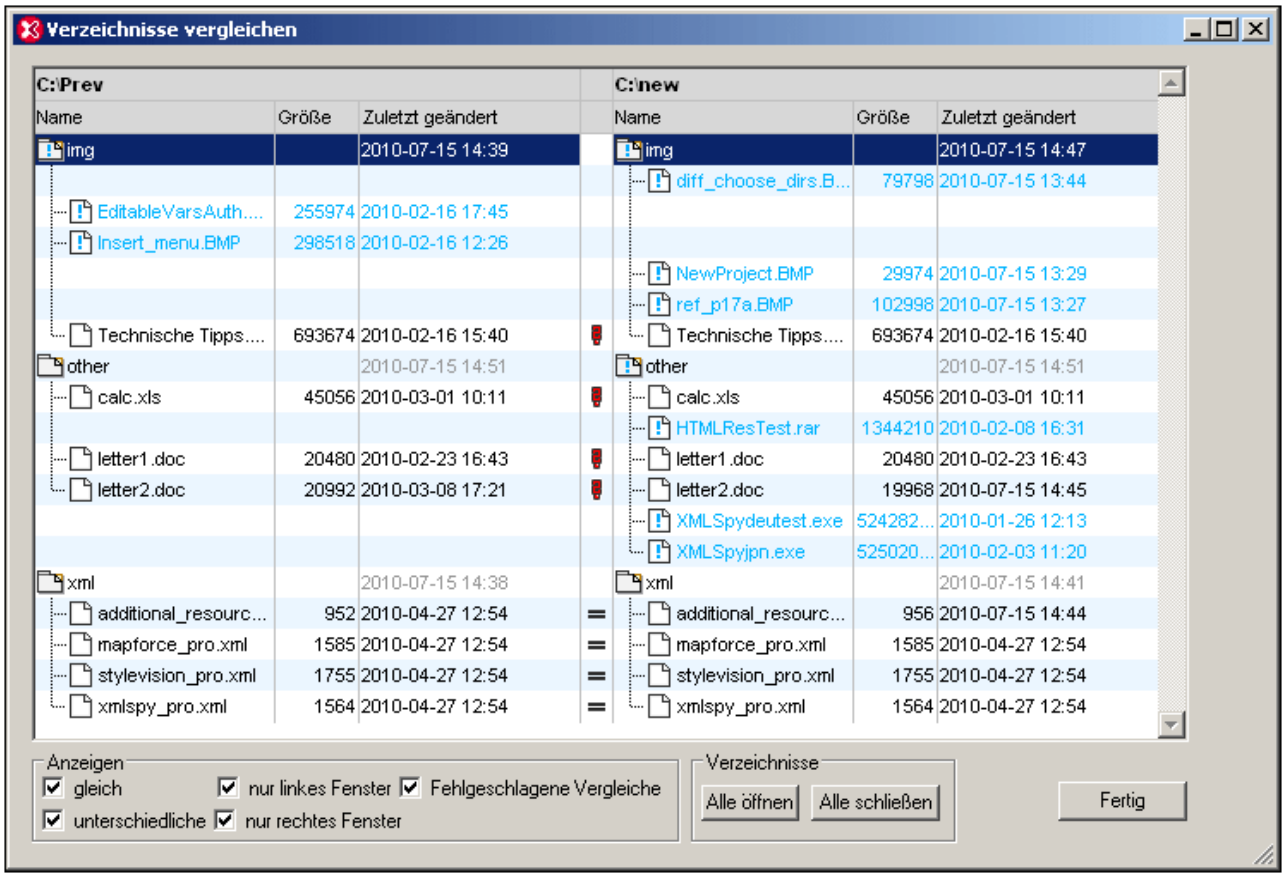
1. Klicken Sie auf **Extras | Verzeichnisse vergleichen**. Daraufhin wird das folgende Dialogfeld angezeigt.



2. Suchen Sie die zu vergleichenden Verzeichnisse und aktivieren Sie das Kontrollkästchen **Unterverzeichnisse berücksichtigen**, wenn Unterverzeichnisse ebenfalls einbezogen werden sollen.
3. Wählen Sie die zu vergleichenden Dateitypen aus. Das Dropdown-Menü enthält drei Optionen: (i) XML-Dateitypen; (ii) In XMLSpy definierte Dateitypen[1606]; und (iii) alle Dateitypen. ZIP-Archive werden als Verzeichnisse behandelt. Verzeichnisse und Dateien in einem ZIP-Archiv werden im Ergebnisfenster angezeigt. Damit ZIP-Archive als Verzeichnisse gelesen werden, muss die erforderliche Erweiterung in das Eingabefeld "Dateien vom Typ" inkludiert werden.
4. Klicken Sie auf OK. Das Dialogfeld "Einstellungen" wird angezeigt (Beschreibung siehe Vergleichseinstellungen[1575]).
5. Wählen Sie die gewünschten Einstellungen für den Dateivergleich aus.
6. Klicken Sie auf OK. Daraufhin erscheint ein Dialogfeld, in dem Sie den Verlauf des Vergleichs beobachten können.

Das Ergebnis des Vergleichs wird in einem Fenster angezeigt und sieht etwa folgendermaßen aus:

© 2016-2022 Altova GmbH



Verzeichnissymbole

Alle Verzeichnisnamen werden schwarz angezeigt.

Das Verzeichnis ist reduziert und sein Inhalt wird nicht angezeigt.

Das Verzeichnis ist erweitert. Dies wird durch die umgeklappte Ecke gekennzeichnet. Der Inhalt des Verzeichnisses wird angezeigt.

Das Verzeichnis enthält Dateien, die entweder alle nicht verglichen werden konnten oder sich nicht von der gleichnamigen Datei im anderen Verzeichnis unterscheiden.

Das Verzeichnis enthält eine oder mehrere Dateien, die im Verzeichnis, mit dem es verglichen wird, nicht vorhanden sind.

Das Verzeichnis enthält eine oder mehrere Dateien, die im damit verglichenen Verzeichnis nicht vorhanden sind und eine oder mehr Dateien, die sich von der gleichnamigen Datei im anderen Verzeichnis unterscheiden.

Bitte beachten Sie: Wenn ein Verzeichnis sowohl eine Datei enthält, die im verglichenen Verzeichnis nicht existiert, als auch eine Datei, die sich von der gleichnamigen Datei im verglichenen Verzeichnis unterscheidet, wird für das Verzeichnis das Symbol angezeigt.

Dateisymbole

Je nach Vergleichsstatus werden die Dateinamen in unterschiedlichen Farben angezeigt. Zur Kennzeichnung werden die folgenden Farben verwendet:



Diese Datei kann nicht (mit der gleichnamigen Datei im anderen Verzeichnis) verglichen werden. In der mittleren Spalte erscheint ein Fragezeichen. Der Dateiname wird schwarz angezeigt.



Diese Datei unterscheidet sich nicht von der gleichnamigen Datei im anderen Verzeichnis. In der mittleren Spalte wird ein Ist gleich-Zeichen angezeigt. Der Dateiname erscheint schwarz.



Diese Datei existiert im verglichenen Verzeichnis nicht. Die mittlere Spalte ist leer. Der Dateiname wird blau angezeigt.



Diese Datei weist Unterschiede zur gleichnamigen Datei im verglichenen Verzeichnis auf und die letzte Änderung dieser Datei ist jüngeren Datums als die letzte Änderung der verglichenen Datei. Die neuere Datei wird hellrot angezeigt und im Symbol wird die hellrote Datei auf der Seite der neueren Datei angezeigt.

Anzeigeoptionen

- Wählen Sie aus, welche Dateien angezeigt werden sollen. Aktivieren bzw. deaktivieren Sie dazu im Anzeigebereich am unteren Rand des Ergebnisfensters die entsprechenden Optionen.
- Öffnen oder schließen Sie alle Unterverzeichnisse durch Klicken auf die entsprechende Schaltfläche im Bereich "Verzeichnisse".
- Erweitern bzw. reduzieren Sie Unterverzeichnisse durch Doppelklicken auf das entsprechende Ordnersymbol.
- Die Spalten "Größe" und "Zuletzt geändert" können ein und ausgeblendet werden, indem Sie mit der rechten Maustaste auf eine Dateikopfzeile klicken und auf "Größe" und "Zuletzt geändert" klicken.



- Durch Ziehen der Spalten können Sie die Spaltenbreite ändern.
- Sie können das Ergebnisfenster maximieren, minimieren oder seine Größe anpassen.

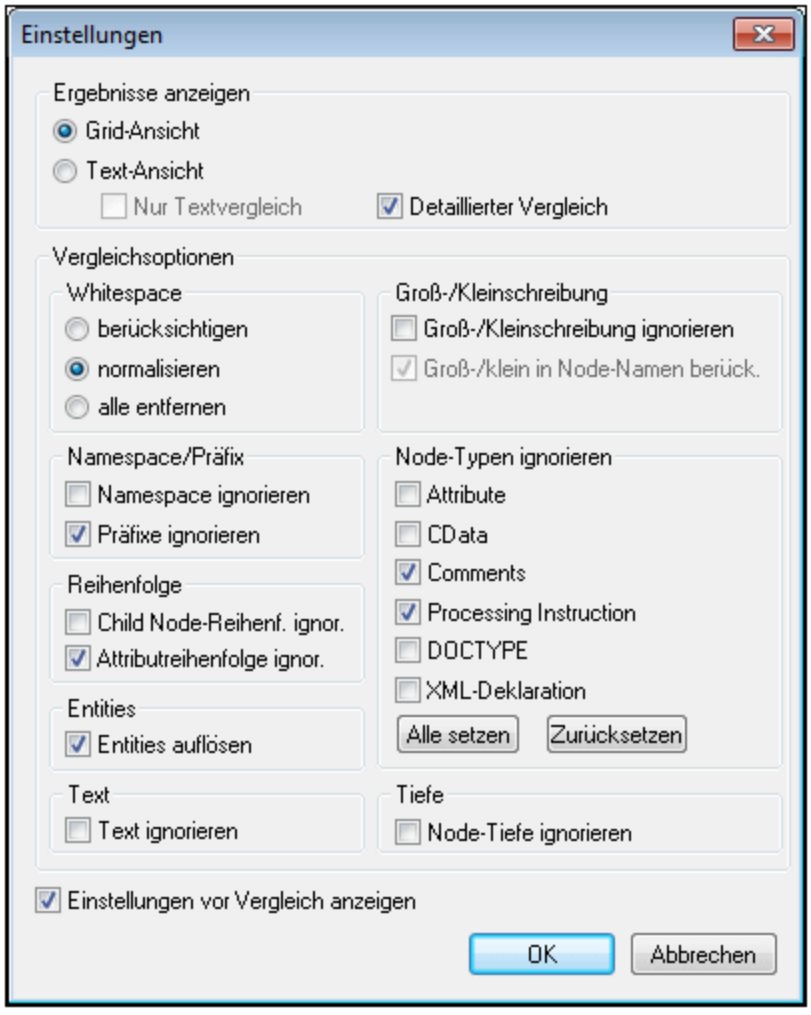
© 2016-2022 Altova GmbH

Vergleichen und Zusammenführen von Dateien

Wenn Sie auf eine Zeile doppelklicken, werden beide in dieser Zeile befindlichen Dateien im Hauptfenster geöffnet, und an den beiden Dateien wird ein Dateivergleich durchgeführt. Sie können anschließend genau wie in einer normalen Vergleichssitzung fortfahren (siehe "Geöffnete Datei vergleichen mit 1567").

28.17.5.3 Vergleichsoptionen

Wenn Sie auf **Vergleichsoptionen** klicken, wird das unten gezeigte Dialogfeld "Einstellungen" angezeigt, in dem Sie die Einstellungen für Ihre Vergleichssitzungen festlegen. Die Einstellungen, die beim Start einer Vergleichssitzung aktiv sind, sind diejenigen, die auf diese Sitzung angewendet werden.



Ergebnisse anzeigen

Wählt die Ansicht aus, in der die Ergebnisse angezeigt werden sollen. Sie haben die Wahl zwischen folgenden Möglichkeiten:

- Grid-Ansicht (XML-Vergleich)
- Textansicht; die Option "Nur Textvergleich" ist deaktiviert (XML-Vergleich)
- Textansicht; die Option "Nur Textvergleich" ist aktiviert (Textvergleich)

Wenn Sie eine Ansicht auswählen, die den XML-Vergleich unterstützt, werden die Dokumente als XML-Dokumente behandelt und die XML-Vergleichsoptionen werden aktiviert. Wird Textvergleich ausgewählt, werden nur diejenigen Optionen aktiviert, die den Textvergleich betreffen (Whitespaces und Groß- und Kleinschreibung); alle anderen Vergleichsoptionen sind deaktiviert.

Anmerkung: Sie können Dateien sowohl in der Grid-Ansicht als auch in der Textansicht und sowohl im XML- als auch im Textvergleichsmodus zusammenführen. Wenn Sie eine Zusammenführung rückgängig machen möchten, stoppen sie den Vergleich, wählen Sie die Datei aus, in der die Änderung rückgängig gemacht werden soll und klicken Sie auf **Bearbeiten | Rückgängig** oder drücken Sie **Strg + Z**.

Detaillierter Vergleich

Ist diese Option deaktiviert, werden Unterschiede zwischen unmittelbaren Geschwisterelementen als ein einziger Unterschied dargestellt und die Zusammenführung wird deaktiviert. Ist die Option aktiv, werden Unterschiede zwischen unmittelbaren Geschwisterelementen als separate Unterschiede markiert und die Zusammenführung wird aktiviert.

Anmerkung: Um die Zusammenführung von Dateien zu aktivieren, muss die Option "Detaillierter Vergleich" aktiviert sein.

Whitespaces

Bei Whitespaces handelt es sich um Leerzeichen, Tabulatorzeichen, Wagenrücklauf und Zeilenvorschub. Wenn Whitespaces normalisiert werden, so werden aufeinander folgende Whitespaces zu einem einzigen Whitespace-Zeichen reduziert; beachten Sie jedoch, dass dass vorangestellte und nachgestellte Leerzeichen in Attributwerten bei einer Normalisierung der Leerzeichen gemäß der XML-Spezifikation komplett entfernt werden. Die Optionen in diesem Bereich dienen (i) zum Vergleichen von Dateien, bei denen die Whitespaces identisch sind, (ii) zum Vergleich von Dateien bei denen die Whitespaces normalisiert wurden und (iii) zum Vergleich von Dateien, bei denen alle Whitespaces entfernt wurden. Die Option "Whitespaces" steht sowohl für XML- als auch für Textvergleiche zur Verfügung.

Groß-/Kleinschreibung

Ist das Kontrollkästchen "Groß-/Kleinschreibung ignorieren" aktiviert, haben Sie die Option, die Groß- und Kleinschreibung in Node-Namen zu ignorieren oder zu berücksichtigen (nur bei XML-Vergleichen). Die Option "Groß-/Kleinschreibung" steht sowohl für XML- als auch für Textvergleiche zur Verfügung.

Namespace/Präfix

Diese Optionen dienen zum Ignorieren von Namespaces und Präfixen beim Suchen nach Unterschieden.

© 2016-2022 Altova GmbH

Reihenfolge

Wenn die Option "Child-Node Reihenfolge ignorieren" ausgewählt ist, spielt die Position der Child Nodes relativ zueinander keine Rolle. Der Vergleich wird an der Gesamtheit aller Child Nodes durchgeführt und wenn der einzige Unterschied zwischen einem Child Node in dem einen Dokument und einem Child Node in dem damit verglichenen Dokument die relative Position im Nodeset ist, so wird dieser Unterschied ignoriert. Jeder Child-Element-Node wird anhand seines Namens, seiner Attribute und seiner Position identifiziert. Haben mehrere Nodes im Geschwister-Nodeset denselben Namen, so werden diese Nodes anhand ihrer Position identifiziert, selbst wenn die Option "Child-Node Reihenfolge ignorieren" ausgewählt ist. Diese Option wird auf jede Ebene separat angewendet. Ist die Option **Child-Node Reihenfolge ignorieren** deaktiviert, so werden die Unterschiede in der Reihenfolge der Nodes als Unterschiede markiert.

Des Weiteren steht eine Option zum Ignorieren der Reihenfolge der Attribute zur Verfügung. Sie wird auf die Reihenfolge der Attribute eines einzelnen Elements angewendet.

Entities

Ist die Option "Entities auflösen" ausgewählt, werden alle Entities im Dokument aufgelöst. Andernfalls werden die Dateien mit den Entities in ihrem gegenwärtigen Zustand verglichen.

Text

Wenn Sie die Option "Text ignorieren" auswählen, werden die Unterschiede in einander entsprechenden Text-Nodes nicht berücksichtigt.

Node-Typen ignorieren

Aktivieren Sie das Kontrollkästchen für die Node-Typen, die beim Dateivergleich nicht berücksichtigt werden sollen. Zur Wahl stehen die Node-Typen Attribute, CDATA, Comments, Processing Instructions, DOCTYPE statements und XML declaration.

Tiefe

Ist die Option "Node-Tiefe ignorieren" ausgewählt, so wird die Tiefe eines Elements (d.h. weitere Ebenen mit Nachfahren dieses Elements) relativ zur Tiefe des entsprechenden Elements in der verglichenen Datei ignoriert. Um die Zusammenführung zu ermöglichen, muss diese Option deaktiviert werden.

Einstellungen vor Vergleich anzeigen

Wenn Sie diese Option aktivieren, wird das Dialogfeld "Einstellungen" (dieses Dialogfeld) vor jedem Datei- oder Verzeichnisvergleich (mit den Befehlen **Geöffnete Datei vergleichen mit** und **Verzeichnisse vergleichen**) angezeigt. Auf diese Art können Sie die Einstellungen vor jedem Vergleich überprüfen und ändern.

Ist dieses Kontrollkästchen deaktiviert, wird der Vergleich direkt gestartet, wenn Sie den entsprechenden Befehl wählen.

28.17.6 Benutzerdefinierte Tools

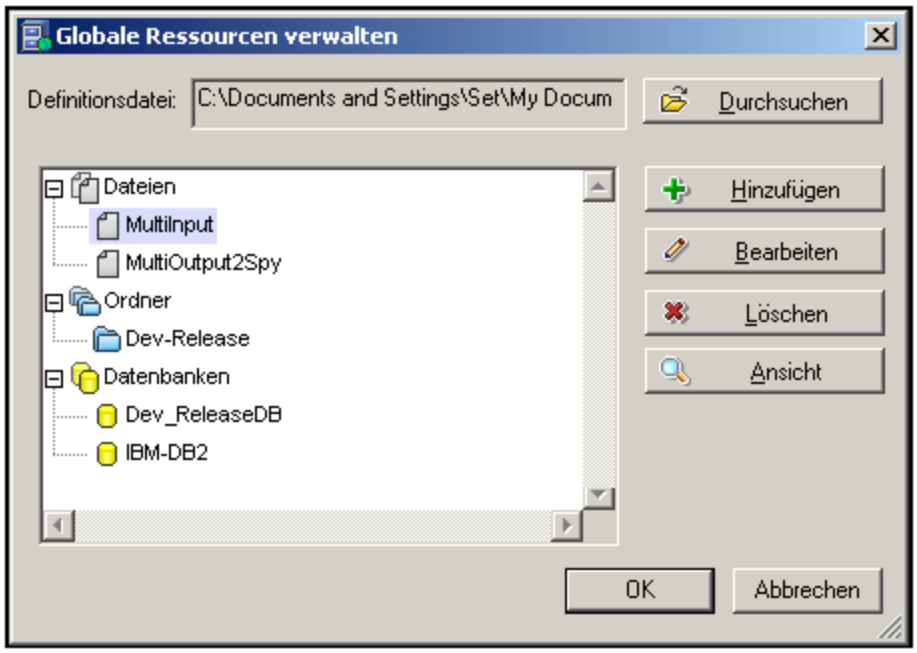
Wenn Sie den Mauszeiger über den Befehl **Benutzerdefinierte Tools** platzieren, wird ein Untermenü mit benutzerdefinierten Befehlen angezeigt, die auf externe Applikationen zugreifen.Diese Befehle können im Dialogfeld "Anpassen" auf dem Register "Extras" 1588 erstellt werden. Wenn Sie auf einen dieser benutzerdefinierten Befehle klicken, wird die mit diesem Befehl verknüpfte Aktion ausgeführt.

Mit dem Befehl **Benutzerdefinierte Tools | Anpassen** wird das Register "Extras" im Dialogfeld "Anpassen" 1588 geöffnet (in dem Sie den benutzerdefinierten Befehl, der im Menü des Befehls **Benutzerdefinierte Tools** angezeigt wird, erstellen können.)

28.17.7 Globale Ressoucen

MIt dem Befehl **Globale Ressourcen** wird das Dialogfeld "Globale Ressourcen" aufgerufen (*Abbildung unten*), in dem Sie Folgendes tun können:

- Angabe der XML-Datei für globale Ressourcen.
- Hinzufügen von globalen Ressourcen (oder Aliassen) vom Typ Datei, Ordner und Datenbank
- Angabe verschiedener Konfigurationen für die einzelnen globalen Ressourcen (Aliasse). Jede Konfiguration wird einer bestimmten Ressource zugeordnet.



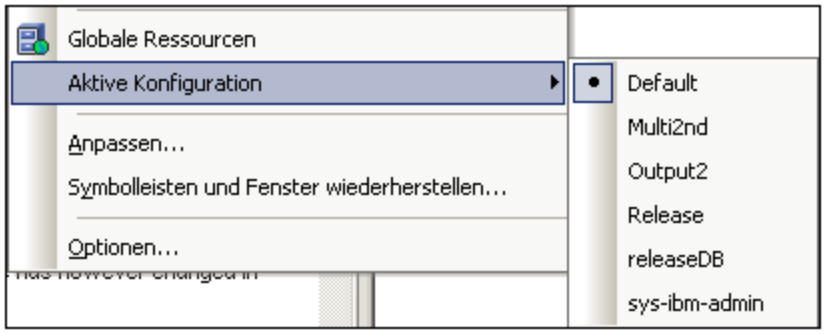
Eine Beschreibung, wie man globale Ressoucen definiert, finden Sie im Abschnitt Definieren globaler Ressourcen 1061.

Anmerkung: Das Dialogfeld "Globale Ressourcen" kann auch über die Symbolleiste "Globale Ressourcen" 1586 aufgerufen werden (**Ansicht | Anpassen |Symbolleisten | Globale Ressourcen**)

© 2016-2022 Altova GmbH

28.17.8 Aktive Konfiguration

Wenn Sie die Maus über den Menübefehl **Aktive Konfiguration** platzieren, wird ein Untermenü ausgerollt, das alle in der aktiven XML-Datei für globale Ressourcen[1578] definierten Konfigurationen enthält. (*Abbildung unten*).

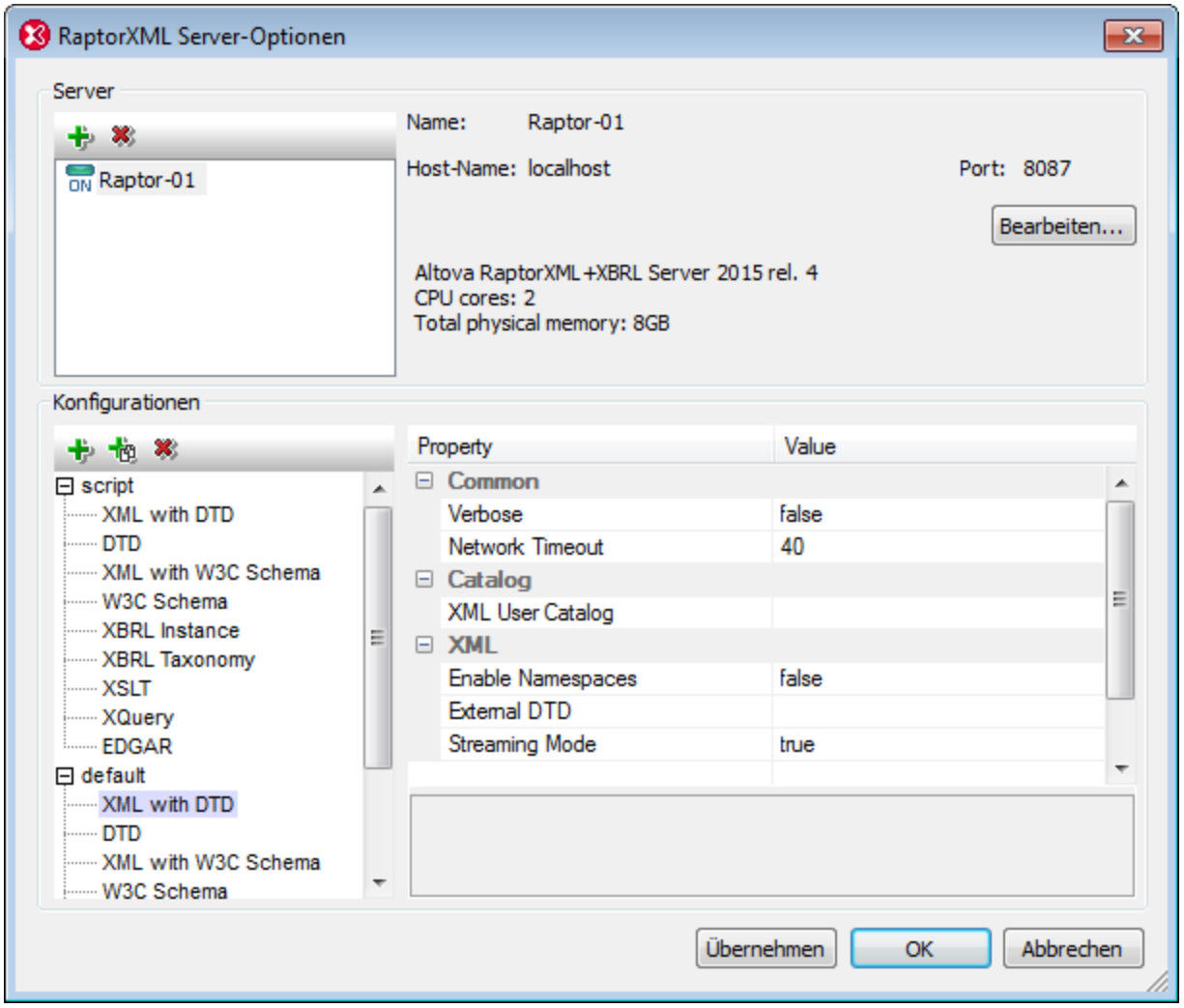


Die gerade aktive Konfiguration ist durch einen Punkt gekennzeichnet. In der Abbildung oben ist die gerade aktive Konfiguration `Default`. Um die aktive Konfiguration zu wechseln, wählen Sie die Konfiguration aus, die aktiv gemacht werden soll.

Anmerkung: Die aktive Konfiguration kann auch über die Symbolleiste "Globale Ressourcen"[1586] ausgewählt werden (**Ansicht | Anpassen | Symbolleisten | Globale Ressourcen**).

28.17.9 Raptor-Server verwalten

Mit dem Befehl **Raptor-Server verwalten** können Sie mehrere Raptor-Server zum Pool der verfügbaren Raptor-Server hinzufügen und anschließend mehrere Konfigurationen für die einzelnen Server definieren. Eine Übersicht zur Verwendung von RaptorXML Server für die Validierung von XML- und XBRL-Dokumenten finden Sie im Kapitel Validieren mit RaptorXML Server[1088].



Hinzufügen eines Raptor-Servers

Klicken Sie im Bereich *Server* des Dialogfelds (*Abbildung unten*) auf das Symbol **Server hinzufügen** und geben Sie anschließend den Namen, anhand dessen der Raptor-Server identifiziert werden soll, den Netzwerk-Namen des Rechners, auf dem Raptor installiert ist (Host-Name) und den Port des Raptor-Servers ein. Klicken Sie auf **OK**, um die Einstellungen zu speichern.

© 2016-2022 Altova GmbH



- *Name:* Hier können Sie einen beliebigen String eingeben. Anhand des Namens wird in XMLSpy ein bestimmter RaptorXML-Server identifiziert.
- *Host-Name:* Der Name oder die IP-Adresse des Netzwerk-Rechners, auf dem der Raptor-Server installiert ist. Die Vearbeitung erfolgt schneller, wenn anstelle eines Host-Namens eine IP-Adresse verwendet wird. Die IP-Adresse für `localhost` (Ihren lokalen Rechner) ist **`127.0.0.1`**.
- *Port:* Der Port, über den der Zugriff auf den Raptor-Server erfolgt. Dieser Port wird in der Konfigurationsdatei von Raptor (`server_config.xml`) definiert. Der Port muss fix und bekannt sein, damit Requests korrekt an den Dienst adressiert werden können. Nähere Informationen zur Raptor-Konfigurationsdatei finden Sie in den Benutzerhandbüchern *RaptorXML Server* und *RaptorXML+XBRL Server*.

Klicken Sie nach Eingabe der Server-Informationen auf **OK**. Der Server-Name, den Sie eingegeben haben, wird (im linken Bereich) in der Liste der Server angezeigt. Neben dem Namen des Servers wird ein grünes Symbol angezeigt. Damit wird gekennzeichnet, dass der Raptor-Server gestartet wurde und ausgeführt wird. Die Details zum Server werden im Bereich angezeigt (*siehe Abbildung oben*). Ein rotes Symbol bedeutet, dass der Server offline ist. Wenn der Server nicht gefunden wurde, wird eine Fehlermeldung angezeigt.

Anmerkung: Der Raptor-Server muss ausgeführt werden, wenn der Server hinzugefügt wird, damit XMLSpy Informationen über den Server abrufen und speichern kann. Wenn der Server, nachdem er hinzugefügt wurde, offline ist oder nicht gefunden wird, so wird dies durch ein rotes Symbol bzw. eine Fehlermeldung angezeigt.

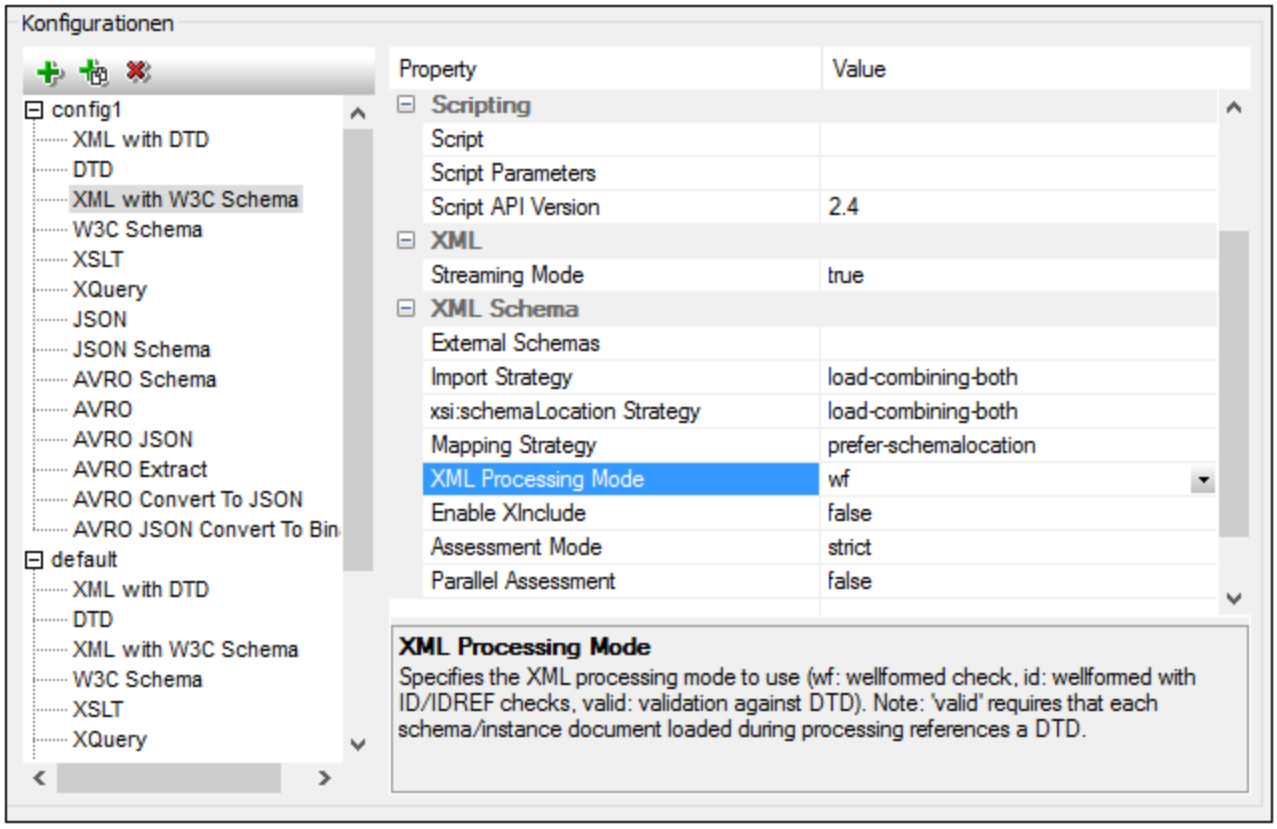
Um den Namen, Host-Namen oder Port eines Servers zu bearbeiten, klicken Sie auf die Schaltfläche **Bearbeiten** und bearbeiten Sie die entsprechenden Informationen im daraufhin angezeigten Dialogfeld nach Bedarf. Um einen Server aus dem Pool zu entfernen, wählen Sie den Server aus und klicken Sie auf das Symbol **Ausgewählten Server entfernen**.

Server-Konfigurationen

Eine Konfiguration ist eine Gruppe von RaptorXML-Validierungsoptionen. Wenn ein Server hinzugefügt wird, erhält er eine Konfiguration namens `default`. Dabei handelt es sich um eine Gruppe von RaptorXML-Optionen, für die als Optionen die Standardeinstellungen definiert sind. Sie können neue Konfigurationen hinzufügen, die andere Optionswerte enthalten. Nachdem Sie mehrere Server-Konfigurationen definiert haben, können Sie eine dieser Konfigurationen als aktive Konfiguration auswählen. Dies ist die Konfiguration, die bei Aktivierung des Befehls **Am Server validieren** verwendet wird.

Der Bereich *Konfigurationen* besteht aus zwei Teilen: (i) einem linken Bereich, in dem die Konfigurationen jeweils mit einer Liste von validierbaren Dokumenttypen aufgelistet werden; (ii) einem rechte Bereich, in dem die Validierunsoptionen des im linken Bereich ausgewählten Dokumenttyps angezeigt werden; am unteren

Rand des rechten Bereichs sehen Sie eine Beschreibung der ausgewählten Option (*siehe Abbildung unten*).



Hinzufügen einer Konfiguration

Klicken Sie im Bereich *Konfigurationen* des Dialogfelds "RaptorXML Server-Optionen" auf **Konfiguration hinzufügen** (*Abbildung oben*). Daraufhin wird eine neue Konfiguration mit den Standardoptionswerten hinzugefügt. Sie können auch auf **Ausgewählte Konfiguration kopieren** klicken, um eine neue Konfiguration zu erstellen. Auf diese Art erstellen Sie eine neue Konfiguration mit den Optionswerten der kopierten Konfiguration. Neue Konfigurationen werden mit Standardnamen vom Typ `config<X>` erstellt. Sie können den Namen einer Konfiguration bearbeiten, indem Sie auf den Namen doppelklicken und einen neuen Namen eingeben. Sie können jeden der Optionswerte der Konfiguration bearbeiten.

Bearbeiten der Optionswerte der Konfiguration

Wählen Sie zuerst den Dokumenttyp im linken Bereich aus. Daraufhin werden die Validierungsoptionen des ausgewählten Dokumenttyps im rechten Bereich angezeigt. Um den Wert einer Option zu bearbeiten, wählen Sie (abhängig vom Typ des Optionswerts) eine der folgenden Methoden:

- Wenn es sich beim Wert um einen aus einer Gruppe vordefinierter Werte handeln kann, wählen Sie den gewünschten Wert aus der Auswahlliste der Wertespalte dieser Option aus.
- Wenn es für den Wert keine Einschränkungen gibt, klicken Sie in das Wertefeld der Option und geben Sie den gewünschten Wert ein.
- Wenn es sich beim Wert um einen Dateipfad handelt, haben Sie zusätzlich zur Eingabe des Werts auch die Möglichkeit, über die Schaltfläche **Durchsuchen** in der Wertespalte der Option zur gewünschten Datei zu navigieren.

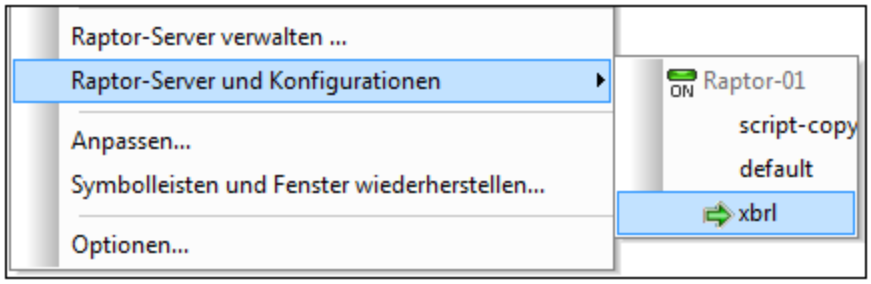
© 2016-2022 Altova GmbH

Wenn Sie eine Option auswählen, wird die Beschreibung dazu in einem Feld im rechten unteren Bereich angezeigt. Ausführlichere Beschreibungen zu den einzelnen Optionen finden Sie in den Benutzerhandbüchern zu RaptorXML Server und RaptorXML(+XBRL) Server in den Kapiteln zu den Befehlszeilenoptionen.

Entfernen einer Konfiguration
Wählen Sie die zu entfernende Konfiguration im linken Bereich aus und klicken Sie auf **Ausgewählte Konfiguration entfernen**.

28.17.10 Raptor-Server und Konfigurationen

Wenn Sie mehrere Konfigurationen auf mehreren Servern definiert haben, können Sie einen Server und eine seiner Konfigurationen als aktive Konfiguration auswählen. Die aktive Konfiguration wird in der Folge für Validierungen verwendet. Wenn Sie den Cursor über den Befehl **Extras | Raptor-Server und Konfigurationen** (*siehe Abbildung unten*) platzieren, wird ein Untermenü mit allen hinzugefügten Servern und deren Konfiguration angezeigt. Wählen Sie die Konfiguration, die Sie zur aktiven Konfiguration machen wollen, aus. In der Abbildung unten wurde die Konfiguration `xbrl` des Servers mit dem Namen `Raptor-01` als aktive Konfiguration ausgewählt (gekennzeichnet durch einen grünen Pfeil).



Eine Übersicht über die Verwendung von Raptor von XMLSpy aus finden Sie im Abschnitt RaptorXML(+XBRL) Server[1088].

28.17.11 XBRL-Taxonomie-Manager

Mit diesem Befehl wird das Dialogfeld Taxonomie-Manager aufgerufen, in dem Sie Ihre XBRL-Taxonomiepakete verwalten können.

Um eine XBRL-Taxonomie zu installieren, aktivieren Sie das Kontrollkästchen neben der gewünschten Taxonomie und klicken Sie auf **Anwenden**. Des Weiteren können Sie Taxonomien deinstallieren, aktualisieren, überprüfen, ob neue Taxonomien hinzugekommen sind, und alle Ihre Taxonomien zentral verwalten.

Nähere Informationen dazu finden Sie unter Taxonomie-Manager[831].

28.17.12 XML-Schema-Manager

Mit diesem Befehl wird das Dialogfeld Schema-Manager aufgerufen, in dem Sie Ihre XML-Schema-Pakete verwalten können.

Um ein XML-Schema zu installieren, aktivieren Sie das Kontrollkästchen neben dem gewünschten Schema und klicken Sie auf **Anwenden**. Des Weiteren können Sie Schemas deinstallieren, aktualisieren, überprüfen, ob neue Schemas hinzugekommen sind, und alle Ihre Schemas zentral verwalten.

Nähere Informationen dazu finden Sie unter Schema-Manager[433].

28.17.13 Anpassen

Mit diesem Befehl können Sie die Applikationsmenüs und Symbolleisten an Ihre eigenen Bedürfnisse anpassen. Bei Aufruf des Befehls, wird das Dialogfeld "Anpassen" angezeigt, das die folgenden Register enthält:

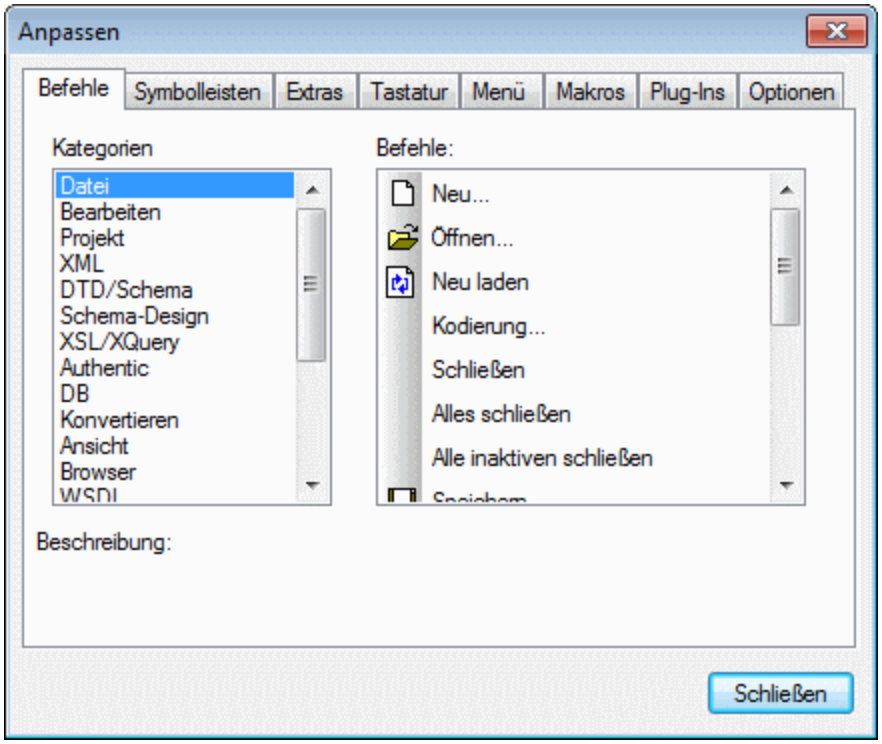
- Befehle[1584]: Alle Applikations- und Makrobefehle können von diesem Register in Menüleisten, Menüs und Symbolleisten gezogen werden.
- Symbolleisten[1586]: Die Symbolleisten können einzeln aktiviert, deaktiviert und zurückgesetzt werden.
- Extras[1557]: Befehle, mit denen externe Programme von der Benutzeroberfläche aus geöffnet werden, können zur Benutzeroberfläche hinzugefügt werden.
- Tastatur[1589]: Für einzelne Applikationen und Makrobefehle können Tastaturkürzel erstellt werden.
- Menü[1593]: Auf diesem Register können Menüleisten und Kontextmenüs, die angepasst werden sollen, ausgewählt und aktiviert werden. Dieses Register wird gemeinsam mit dem Register "Befehle" verwendet.
- Makros[1595]: Makros können hier mit neuen Befehlen verknüpft werden.
- Plug-Ins[1597]: Plug-Ins können aktiviert und in die Benutzeroberfläche integriert werden.
- Optionen[1602]: Auf diesem Register werden Anzeigeoptionen für Symbolleisten eingestellt.

In diesem Abschnitt wird auch das Kontextmenü[1598] beschrieben, das bei geöffneten Dialogfeld "Anpassen" bei Rechtsklick auf Elemente der Menüleiste, des Menüs oder der Symbolleiste angezeigt wird.

28.17.13.1 Befehle

Auf dem Register Befehle können Sie Ihre Menüs oder Symbolleisten anpassen. Sie können ganz nach Wunsch Applikationsbefehle zu Ihren Menüs und Symbolleisten hinzufügen. Beachten Sie, dass Sie selbst keine neuen Applikationsbefehle oder Menüs erstellen können.

© 2016-2022 Altova GmbH



So fügen Sie einen Befehl zu einer Symbolleiste oder einem Menü hinzu:

1. Öffnen Sie das Dialogfeld über die Menübefehle **Extras | Anpassen**. Daraufhin wird das Dialogfeld "Anpassen" angezeigt.
2. Wählen Sie die Befehlskategorie *Alle Befehle* im Kategorie-Listenfeld. Die verfügbaren Befehle werden im entsprechenden Listenfeld angezeigt.
3. Markieren Sie einen Befehl aus dem Listenfeld, und ziehen Sie ihn in ein existierendes Menü oder eine Symbolleiste.
4. Ein **I**-Zeichen wird angezeigt, wenn Sie den Cursor an eine Stelle bewegen, an die der ausgewählte Befehl gestellt werden kann.
5. Lassen Sie die Maustaste los, um den Befehl an der gewünschten Stelle einzufügen.

Beachten Sie die folgenden Punkte.

- An der Spitze des Mauszeigers wird beim Ziehen eines Befehls eine kleine Schaltfläche angezeigt. Damit wird angezeigt, dass der Befehl gerade mit der Maus gezogen wird
- Ein "**x**" unterhalb des Mauszeigers bedeutet, dass der Befehl nicht an die aktuelle Cursorposition gezogen werden kann.
- Sobald der Cursor an eine Stelle verschoben wird, an die der Befehl gezogen werden kann (eine Symbolleiste oder ein Menü), verschwindet das **x**. Anstelle des **x**-Zeichens erscheint ein **I**-Zeichen.
- Befehle können in Menüs oder Symbolleisten eingefügt werden. Wenn Sie eine eigene Symbolleiste erstellt[1586] haben, können Sie nun die gewünschten Befehle einfügen.
- Wenn Sie den Cursor über ein geschlossenes Menü platzieren, wird dieses Menü geöffnet, sodass Sie den Befehl an einer beliebigen Stelle in diesem Menü einfügen können.

Hinzufügen von Befehlen zu Kontextmenüs

Sie können Befehle auch zu Kontextmenüs hinzufügen, indem Sie die Befehle aus dem Listenfeld *Befehle* in das Kontextmenü ziehen. Gehen Sie dazu folgendermaßen vor:

1. Klicken Sie im Dialogfeld "Anpassen" auf das Register 1593 **"Menü"** 1593.
2. Wählen Sie im Bereich "Kontextmenü" ein Kontextmenü aus der Auswahlliste aus. Daraufhin wird das ausgewählte Kontextmenü angezeigt.
3. Wechseln Sie im Dialogfeld "Anpassen" zurück zum Register "Befehle".
4. Ziehen Sie den gewünschten Befehle aus der Liste *Befehle* an die gewünschte Stelle im Kontextmenü.

Löschen eines Befehls oder Menüs

Um einen Befehl aus einem Menü, einem Kontextmenü (nähere Informationen zum Aufrufen von Kontextmenüs siehe oben) oder einer Symbolleiste zu löschen oder um ein ganzes Menü zu löschen, gehen Sie folgendermaßen vor:

1. Öffnen Sie das Dialogfeld "Anpassen" (**Extras | Anpassen).**
2. Klicken Sie, während das Dialogfeld "Anpassen" geöffnet (und ein beliebiges Register ausgewählt) ist, mit der rechten Maustaste auf ein Menü oder einen Menübefehl und wählen Sie anschließend im Kontextmenü den Befehl **Löschen**. Alternativ dazu können Sie das Menü oder den Menübefehl auch mit der Maus ziehen, bis ein "x"-Symbol unterhalb des Mauszeigers erscheint. Wenn Sie die Maustaste nun loslassen, wird das Menü bzw. der Menübefehl gelöscht.

Um gelöschte Menübefehle wiederherzustellen, gehen Sie vor, wie in diesem Abschnitt beschrieben. Um ein gelöschtes Menü wiederherzustellen, gehen Sie zu **Extras | Anpassen | Menü** und klicken Sie im Bereich *Applikations-Menüs* auf die Schaltfläche **Zurücksetzen**. Alternativ dazu können Sie das Menü auch über **Extras | Anpassen | Symbolleisten**, durch Auswahl von "Menüleiste" und klicken auf die Schaltfläche "Zurücksetzen" zurücksetzen.

28.17.13.2 Symbolleisten

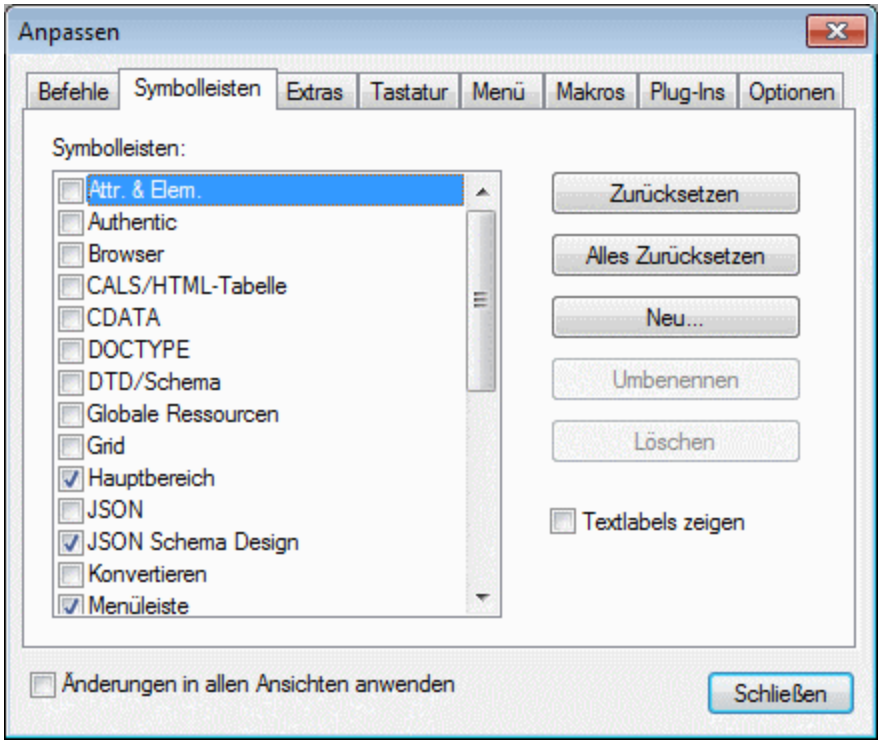
Auf dem Register **Symbolleisten** können Sie (i) spezifische Symbolleisten aktivieren oder deaktivieren, d.h. festlegen, welche davon auf der Benutzeroberfläche angezeigt werden sollen; (ii) festlegen, welche Schaltflächen in jeder Symbolleiste angezeigt werden sollen und (iii) Ihre eigenen Symbolleisten erstellen.

Die Symbolleisten enthalten Schaltflächen für die meistverwendeten Befehle. Zu jeder Schaltfläche werden in einem Tooltip und in der Statusleiste Informationen angezeigt, wenn Sie den Cursor über die Schaltfläche platzieren. Sie können die Symbolleisten aus Ihrer Standardposition an eine andere beliebige Stelle am Bildschirm verschieben, wo diese als frei schwebendes Fenster angezeigt werden.

Anmerkung: Um einen Befehl zu einer Symbolleiste hinzuzufügen, ziehen Sie den gewünschten Befehl aus dem Listenfeld *Befehle* des Registers **Befehle** 1584 in die Symbolleiste. Um einen Befehl aus einer Symbolleiste zu löschen, öffnen Sie das Dialogfeld "Anpassen" und ziehen Sie den Befehl mit der Maus aus der Symbolleiste hinaus (näherer Informationen siehe **Befehle** 1584).

Anmerkung: Symbolleisteneinstellungen, die in einer bestimmten Ansicht definiert wurden, stehen standardmäßig nur in der jeweiligen Ansicht zur Verfügung. Damit die Einstellungen auf alle Ansichten angewendet werden, aktivieren Sie das Kontrollkästchen am unteren Rand des Dialogfelds.

© 2016-2022 Altova GmbH



Es stehen die folgenden Funktionalitäten zur Verfügung:

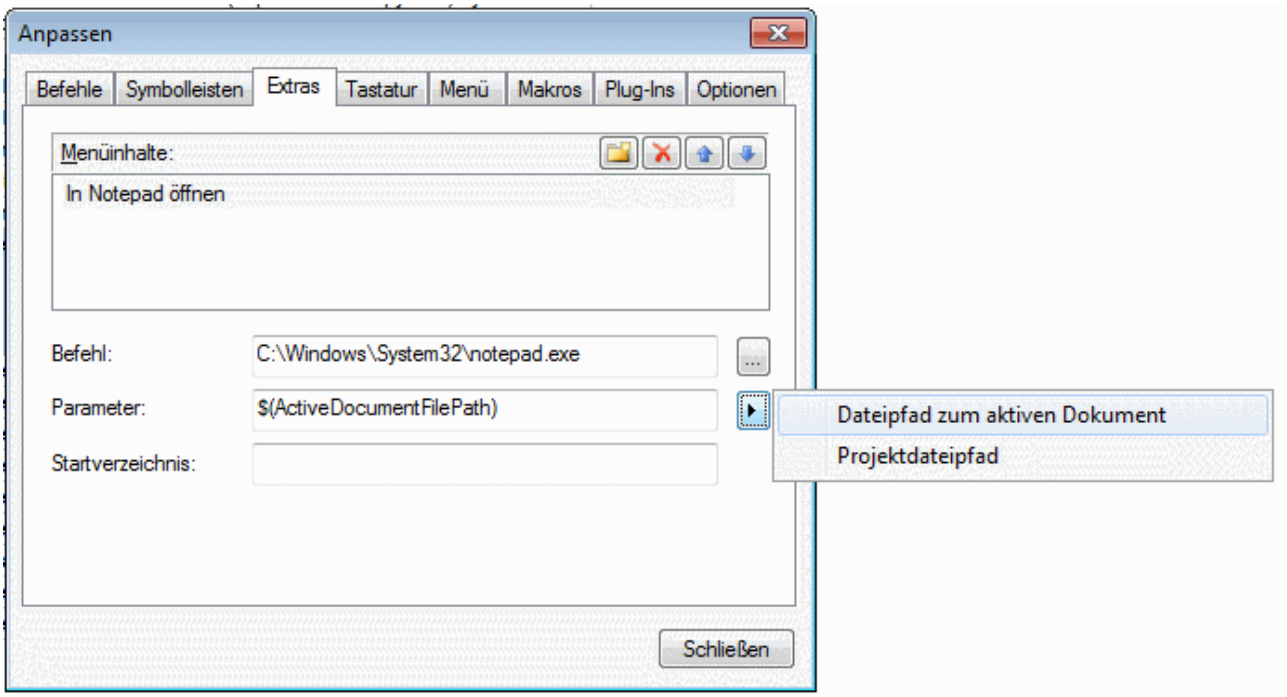
- *Um eine Symbolleiste zu aktivieren oder deaktivieren*: Klicken Sie im Listenfeld *Symbolleisten* auf das jeweilige Kontrollkästchen.
- *Um Änderungen auf alle Ansichten anzuwenden*: Aktivieren Sie das Kontrollkästchen am unteren Rand des Dialogfelds. Andernfalls werden die Änderungen nur auf die aktuelle Ansicht angewendet. Beachten Sie, dass nur Änderungen, die **nach** Aktivierung des Kontrollkästchens vorgenommen wurden, auf alle Ansichten angewendet werden.
- *Um eine neue Symbolleiste hinzuzufügen*: Klicken Sie auf die Schaltfläche **Neu** und geben Sie der neuen Symbolleiste einen Namen (Dialogfeld Symbolleistenname). Ziehen Sie Befehle aus dem Register **Befehle** 1584 in die neue Symbolleiste.
- *Um den Namen einer hinzugefügten Symbolleiste zu ändern:* Wählen Sie die hinzugefügte Symbolleiste aus, klicken Sie auf die Schaltfläche **Umbenennen** und bearbeiten Sie den Namen im Dialogfeld "Symbolleistenname".
- *Um die Menüleiste zurückzusetzen:* Klicken Sie im Bereich *Symbolleisten* auf den Menüleisteneintrag und klicken Sie auf die Schaltfläche **Zurücksetzen.** Daraufhin wird die Menüleiste auf die Standardeinstellung zurückgesetzt.
- *Um alle Symbolleisten- und Menübefehle zurückzusetzen:* Klicken Sie auf die Schaltfläche **Alles Zurücksetzen.** Daraufhin werden alle Symbolleisten und Menüs auf die Standardinstallationseinstellung zurückgesetzt.

- *Um eine Symbolleiste zu löschen:* Wählen Sie die gewünschte Symbolleiste im Bereich *Symbolleisten* aus und klicken Sie auf **Löschen**.

- *Um Textlabels von Befehlen in einer bestimmten Symbolleiste anzuzeigen:* Wählen Sie die gewünschte Symbolleiste aus und aktivieren Sie das Kontrollkästchen *Textlabels anzeigen*. Beachten Sie, dass die Textlabels für die einzelnen Symbolleisten separat aktiviert werden müssen.

28.17.13.3 Extras

Auf dem Register **Extras** können Sie Befehle definieren, mit denen Sie von XMLSpy aus externe Applikationen verwenden können. Diese Befehle werden zum Menü **Extras | Benutzerdefinierte Tools** hinzugefügt. So kann z.B. die aktive Datei im Hauptfenster von XMLSpy durch Auswahl eines Befehls, den Sie im Menü **Extras | Benutzerdefinierte Tools** erstellt haben, in einer externen Applikation wie z.B. Notepad geöffnet werden.



Um einen Befehl zur Verwendung einer externen Applikation zu definieren, gehen Sie folgendermaßen vor:

1. Klicken Sie im Bereich *Menüinhalte* (*siehe Abbildung oben*) in der Titelleiste des Bereichs auf die Schaltfläche "Neu" und geben Sie in der Zeile, die für den Eintrag erstellt wurde, den Namen des gewünschten Menübefehls ein. In der Abbildung oben wurde ein einziger Menübefehl eingegeben, nämlich **In Notepad öffnen**. Mit diesem Befehl soll das aktive Dokument in der externen Applikation Notepad geöffnet werden. Durch Klicken auf die Schaltfläche "Neu" können weitere Befehle zur Befehlsliste hinzugefügt werden. Mit Hilfe der Schaltflächen **Element hinauf** und **Element hinab** können Sie die Befehle in der Liste nach oben oder unten verschieben. Um einen Befehl zu löschen, wählen Sie ihn aus und klicken Sie auf die Schaltfläche **Löschen.**
2. Um eine externe Applikation mit einem Befehl zu verknüpfen, wählen Sie den Befehl im Bereich *Menüinhalte* aus. Geben Sie anschließend im Feld *Befehl* den Pfad zur ausführbaren Datei der

 © 2016-2022 Altova GmbH

externen Applikation ein oder wählen Sie ihn aus. In der Abbildung oben wurde im Feld *Befehl* der Pfad zur Applikation Notepad eingegeben.

3. Wenn Sie auf die Flyout-Schaltfläche des Felds *Parameter* klicken (*siehe Abbildung oben)*, werden die Aktionen angezeigt, die für die externe Applikation verfügbar sind. Diese Aktionen sind in der Liste unterhalb davon beschrieben. Wenn Sie eine Aktion auswählen, wird ein Code-String für die Aktion in das Feld *Parameter* eingetragen.
4. Wenn Sie ein aktuelles Arbeitsverzeichnis definieren möchten, geben Sie dieses in das Feld *Startverzeichnis* ein.
5. Klicken Sie auf **Schließen,** um den Vorgang abzuschließen.

Die von Ihnen erstellten Befehle werden im Menü **Extras | Benutzerdefinierte Tools** und im Kontextmenü von Dateien und Ordnern im Projektfenster im Untermenü **Benutzerdefinierte Tools** angezeigt.

Wenn Sie (im Menü **Extras | Benutzerdefinierte Tools**) auf den von Ihnen erstellten Befehl klicken, wird die mit dem Befehl verknüpfte Aktion ausgeführt. In dem in der Abbildung oben gezeigten Beispiel geschieht Folgendes: Das im Hauptfenster von XMLSpy aktive Dokument wird in Notepad geöffnet. Der Befehl für die externe Applikation steht auch im Projektfenster im Kontextmenü für die Dateien zur Verfügung (Klicken Sie mit der rechten Maustaste auf eine Datei im Projektfenster, um das Kontextmenü für die Datei anzuzeigen). Sie können über das Projektfenster auch mehrere Dateie öffnen (wenn dies die Applikation erlaubt). Wählen Sie dazu mehrere Dateien aus und klicken Sie anschließend im Kontextmenü auf den entsprechenden Befehl.

Parameter

Im Feld *Parameter* wird die Aktion definiert, die vom Befehl der externen Applikation ausgeführt werden soll. Die folgenden Parameter stehen zur Verfügung.

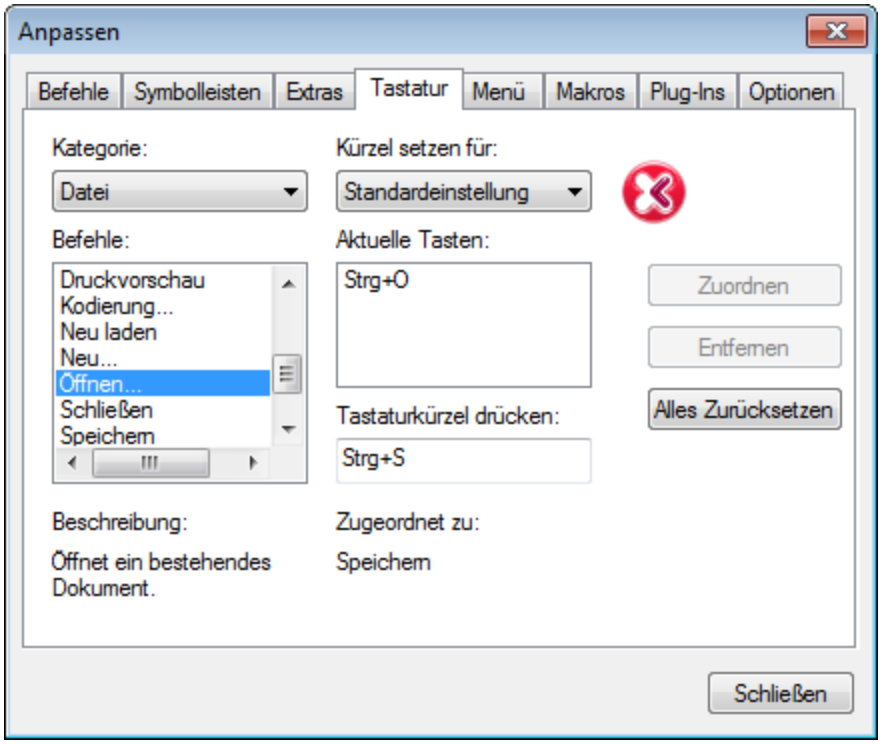
- *Dateipfad zum aktiven Dokument:* Mit dem Befehl im Menü **Extras | Benutzerdefinierte Tools** wird das in XMLSpy aktive Dokument in der externen Applikation geöffnet. Der Befehl im Kontextmenü einer Datei im Projektfenster öffnet die ausgewählte Datei in der externen Applikation.
- *Projektdateipfad:* Öffnet die XMLSpy Projektdatei (die `.spp` Datei) in der externen Applikation.

Startverzeichnis

Der Eintrag unter *Startverzeichnis* ist optional und ist ein Pfad, der als das aktuelle Verzeichnis verwendet wird.

28.17.13.4 Tastatur

Auf dem Register Tastatur können Sie neue Tastaturkürzel für jeden Applikationsbefehl definieren oder bestehende ändern.



Um einem Befehl ein neues Kürzel zuzuweisen oder ein bestehendes Tastaturkürzel zu ändern, gehen Sie folgendermaßen vor:

1. Wählen Sie in der Auswahlliste die Kategorie *Alle Befehle* aus. Beachten Sie: Wenn ein Makro als verknüpfter Befehl ausgewählt wurde[1595], können über die Auswahlliste auch Makros ausgewählt werden und es kann für das Makro ein Tastenkürzel definiert werden.
2. Wählen Sie im Listenfeld Befehle den Befehl aus, welchem Sie ein neues Kürzel zuweisen möchten, bzw. dessen Kürzel Sie ändern möchten.
3. Klicken Sie in das Textfeld Tastaturkürzel drücken und drücken Sie das Tastaturkürzel, das Sie diesem Befehl zuweisen möchten. Das Kürzel wird im Textfeld Tastaturkürzel drücken angezeigt. Wenn das Kürzel noch mit keinem Befehl verknüpft ist, ist die Schaltfläche **Zuweisen** aktiv, Wenn das Kürzel schon einem anderen Befehl zugeordnet war, wird dieser Befehl unterhalb des Textfelds angezeigt und die Schaltfläche **Zuweisen** ist deaktiviert. (Um den Eintrag im Feld Tastaturkürzel drücken zu löschen, drücken Sie eine der Tasten **Strg**, **Alt** oder **Umschalt**).
4. Klicken Sie auf die Schaltfläche **Zuweisen**, um das Tastaturkürzel zuzuweisen. Das Kürzel wird nun im Listenfeld Aktuelle Tasten angezeigt. Sie können einem einzigen Befehl mehrere Tastenkürzel zuweisen.
5. Klicken Sie auf die Schaltfläche **Schließen**, um die Einstellung zu bestätigen.

Löschen eines Tastaturkürzels

Ein Tastaturkürzel kann nicht mehreren Befehlen zugewiesen werden. Um ein Kürzel zu löschen, klicken Sie in das Listenfeld *Aktuelle Tasten* und anschließend auf die Schaltfläche **Entfernen.**

Kürzel setzen für:

Derzeit können Tastaturkürzel nur als Standard definiert werden. Es steht kein anderer Modus zur Verfügung.

© 2016-2022 Altova GmbH

Standard-Tastaturkürzel

In der unten stehenden Tabelle sind die Standard-Tastenkürzel häufig verwendeter Befehle aufgelistet. Eine Übersicht über alle Applikationsmenübefehle finden Sie in der Tastenzuordnung (**Hilfe | Tastaturzuordnungen**[1653]).

- *Funktionstasten-Tastaturkürzel (einschl. Validierung und Transformation)*

F1	Menü "Hilfe"
F1 + Alt	Letzte Datei öffnen
F3	Weitersuchen
F4 + STRG	Aktives Fenster schließen
F4 + Alt	XMLSpy schließen
F5	Aktualisieren
F6 + STRG	Offene Fenster der Reihe nach aufrufen
F7	Wohlgeformtheit prüfen
F8	Validieren
F10	XSL-Transformation
F10 + STRG	XSL:FO-Transformation

- *Datei- und Applikationsbefehle*

Alt + F1	Letzte Datei öffnen
STRG + O	Datei öffnen
STRG + N	Datei neu
STRG + P	Datei drucken
STRG + S	Datei speichern
STRG + F4	Aktives Fenster schließen
STRG + F6	Offene Fenster der Reihe nach aufrufen
STRG + TAB	Zwischen offenen Dokumenten wechseln
Alt + F4	XMLSpy schließen

- *Diverse Tastaturkürzel*

Pfeil nach oben/nach unten	Cursor oder Auswahlleiste verschieben
Esc	Bearbeitungen verwerfen oder Dialogfeld schließen
Eingabetaste	Auswahl bestätigen
Entf	Zeichen oder Auswahl löschen
Umschalt + Entf	Ausschneiden

- *Bearbeitungsbefehle*

STRG + A	Alles markieren

STRG + F	Suchen
STRG + G	Gehe zu Zeile/Zeichen
STRG + H	Ersetzen
STRG + V	Einfügen
STRG + X	Ausschneiden
STRG + Y	Wiederherstellen
STRG + Z	Rückgängig machen

Befehle in der Textansicht

STRG + E	Zwischen Start- und End-Tag hin- und herspringen
STRG + Umschalt + E	Element, das den Cursor enthält, auswählen
STRG + Alt + E	Gehe zum übergeordneten Element
STRG + "+"	Vergrößern
STRG + "–"	Verkleinern
STRG + 0	Zoom zurücksetzen
STRG + Mausrad vorwärts	Vergrößern
STRG + Mausrad rückwärts	Verkleinern

Befehle in der Grid-Ansicht

STRG + D	CDATA anhängen
STRG + E	Element anhängen
STRG + I	Attribut anhängen
STRG + M	Comment anhängen
STRG + T	Text anhängen
STRG + Shift + D	CDATA einfügen
STRG + Shift + E	Element einfügen
STRG + Shift + I	Attribut einfügen
STRG + Shift + M	Comment einfügen
STRG + Shift + T	Text einfügen
STRG + Alt + D	Child CDATA hinzufügen
STRG + Alt + E	Child Element hinzufügen
STRG + Alt + I	Child Attribut hinzufügen
STRG + Alt + M	Child Comment hinzufügen
STRG + Alt + T	Child Text hinzufügen

© 2016-2022 Altova GmbH

Befehle in der Schema-Ansicht

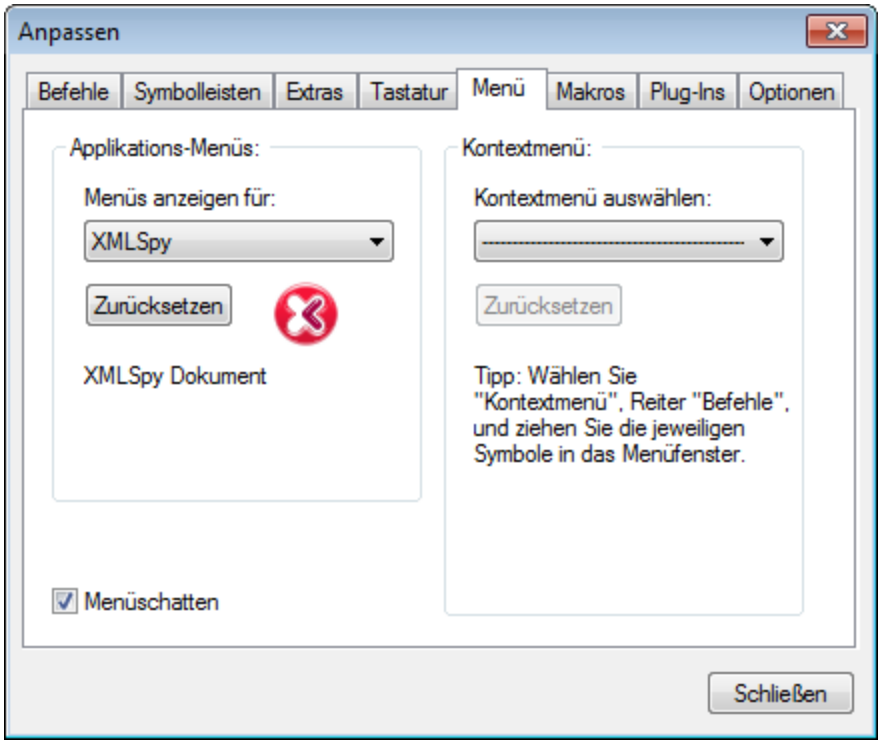
STRG + Doppelklick auf Element	Element-Definition anzeigen

- *Debugger-Befehle*

F9	Breakpoint einfügen/löschen
F9 + Umschalt	Tracepoint einfügen/löschen
F9 + STRG	Breakpoint aktivieren/deaktivieren
F9 + Umschalt + STRG	Tracepoint aktivieren/deaktivieren
F11	Einsteigen
F11 + Umschalt	Aussteigen
F11 + STRG	Überspringen
F11 + Alt	Debugger starten

28.17.13.5 Menü

Auf dem Register **Menü** können Sie die beiden Hauptmenüleisten (Standardmenüleiste und Applikationsmenüleiste) und die Kontextmenüs der Applikation anpassen.



Anpassen der Standardmenüleiste und der Applikationsmenüleiste

Die Standardmenüleiste ist die Menüleiste, die angezeigt wird, wenn kein Dokument im Hauptfenster geöffnet ist. Die Applikationsmenüleiste ist die Menüleiste, die angezeigt wird, wenn ein oder mehrere Dokumente im Hauptfenster geöffnet sind. Jede Menüleiste kann separat angepasst werden. Änderungen, die an einer der Leisten vorgenommen wurden, haben keine Auswirkung auf andere Menüleisten.

Um eine Menüleiste anzupassen, wählen Sie diese in der Auswahlliste *Mernüs anzeigen für* (*siehe Abbildung oben*) aus. Wechseln Sie anschließend zum Register "Befehle" des Dialogfelds "Anpassen" 1584 und ziehen Sie die Befehle aus dem Listenfeld "Befehle" in die Menüleiste oder in eines der Menüs.

Löschen von Befehlen aus Menüs und Zurücksetzen der Menüleisten

So **löschen** Sie ein ganzes Menü oder einen Befehl in einem Menü:

1. Wählen Sie im Bereich "Applikations-Menüs" entweder *Standardmenü* (damit werden die Menüs angezeigt, die verfügbar sind, wenn kein Dokument offen ist) oder *XMLSpy* (damit werden die Menüs angezeigt, die verfügbar sind, wenn ein oder mehrere Dokumente offen sind) aus.
2. Während das Dialogfeld "Anpassen" geöffnet ist, (i) wählen Sie das Menü, das Sie aus der Menüleiste der Applikation löschen möchten, aus oder (ii) wählen Sie den Befehl aus, den Sie aus einem dieser Menüs löschen möchten.
3. Ziehen Sie entweder das Menü aus der Menüleiste oder den Menübefehl aus dem Menü oder (ii) klicken Sie mit der rechten Maustaste auf das Menü oder den Menübefehl und wählen Sie den Befehl **Löschen**.

Sie können jede der beiden Menüleisten (die Standardmenüleiste und die Applikationsmenüleiste) in ihren Originalzustand (Zustand bei der Installation) **zurücksetzen**. Wählen Sie dazu das Menü in der Auswahlliste *Menüs anzeigen für* aus und klicken Sie anschließend unterhalb der Auswahlliste auf den Befehl **Zurücksetzen**.

Anpassen der Kontextmenüs der Applikation

Die Kontextmenüs sind die Menüs, die angezeigt werden, wenn Sie mit der rechten Maustaste auf bestimmte Objekte auf der Benutzeroberfläche der Applikation klicken. Jedes dieser Kontextmenüs kann folgendermaßen angepasst werden:

1. Wählen Sie das gewünschte Kontextmenü in der Auswahlliste *Kontextmenü auswählen* aus. Daraufhin wird das Kontextmenü angezeigt.
2. Wechseln Sie auf das Register Befehle des Dialogfelds "Anpassen" 1584.
3. Ziehen Sie den gewünschten Befehl aus dem Listenfeld *Befehle* in das Kontextmenü.
4. Wenn Sie einen Befehl aus dem Kontextmenü löschen möchten, klicken Sie mit der rechten Maustaste auf diesen Befehl im Kontextmenü und wählen Sie den Befehl **Löschen**. Alternativ dazu können Sie den zu löschenden Befehl auch mit der Maus aus dem Kontextmenü herausziehen.

Sie können jedes Kontextmenü in den Originalzustand zurücksetzen. Wählen Sie dazu das Kontextmenü in der Auswahlliste *Kontextmenüs anzeigen* aus und klicken Sie anschließend unterhalb der Auswahlliste auf den Befehl **Zurücksetzen**.

Menüschatten

Klicken Sie auf das Kontrollkästchen *Menüschatten*, wenn Menüs mit Schatten dargestellt werden sollen.

© 2016-2022 Altova GmbH

28.17.13.6 Makros

Über das Register **Makros** können Sie Applikationsbefehle für mit dem Skript-Editor von XMLSpy erstellte Makros erstellen. Diese Applikationsbefehle (mit denen die damit verknüpften Makros ausgeführt werden), können anschließend in Menüs und Symbolleisten zur Verfügung gestellt werden - entweder direkt über das Register "Makros" oder über den Mechanismus auf dem Register Befehle des Dialogfelds "Anpassen"[1584]. Als Applikationsbefehlen können diesen Makros über das Register "Tastatur" des Dialogfelds "Anpassen"[1589] auch Tastaturkürzel zugewiesen werden.

Funktionsweise von Makros in XMLSpy

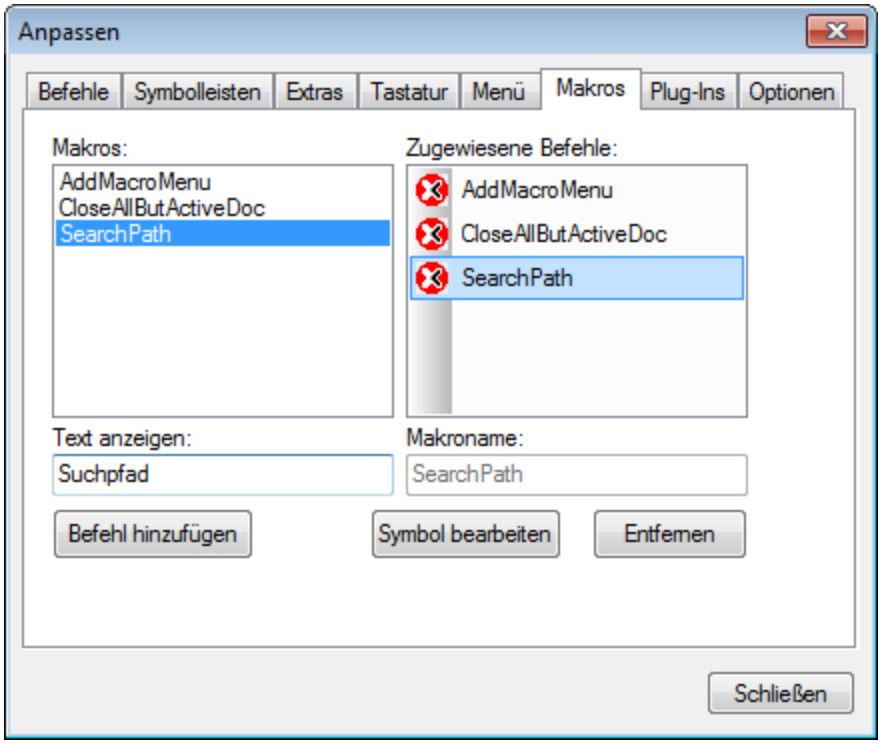
Makros funktionieren in XMLSpy folgendermaßen:

- Altova Skripting-Projekte (`.asprj` Dateien) werden im Skript-Editor[1662] von XMLSpy erstellt. In diesen Skripting-Projekten sind die in XMLSpy verwendeten Makros enthalten.
- In XMLSpy können gleichzeitig zwei Skripting-Projekte aktiv sein: (i) ein Applikations-Skripting-Projekt, das im Dialogfeld "Optionen" im Abschnitt "Skripting"[1645] definiert wird, und (ii) das Skripting-Projekt des aktiven XMLSpy-Projekts[1080], das im Dialogfeld "Skript-Einstellungen"[1344] (**Projekt | Skript-Einstellungen**[1344]) definiert wird.
- Die Makros in diesen beiden Skripting-Projekten stehen in der Applikation im Menü **Extras | Makros** (über das die Makros ausgeführt werden können) und im Dialogfeld "Anpassen" auf dem Register "Makros" (*Abbildung unten*), über das sie als Applikationsbefehle definiert werden können, zur Verfügung. Nachdem ein Makro als Applikationsbefehl definiert wurde, kann der Befehl in ein Menü und/oder eine Symbolleiste platziert werden.

Erstellen eines Applikationsbefehls für ein Makro

Erstellen Sie zuerst im Skript-Editor[1662] (**Extras | Skript-Editor**[1662]) das gewünschte Makro und speichern Sie es in einem Skripting-Projekt. Definieren Sie diese Datei entweder als das Applikations-Skripting-Projekt (über das Dialogfeld "Optionen", Register "Skripting")[1645] oder als Skripting-Projekt des aktiven Applikationsprojekts (über das Dialogfeld "Skript-Einstellungen"[1344] (**Projekt | Skript-Einstellungen**[1344])). Die Makros im Skripting-Projekt werden nun auf dem Register "Makros" im Bereich *Makros* angezeigt (*siehe Abbildung unten)*

Um einen Applikationsbefehl für ein Makro zu erstellen, wählen Sie das Makro im Bereich *Makros* aus, definieren Sie im Textfeld *Text anzeigen* den Text und klicken Sie auf **Befehl hinzufügen** *(siehe Abbildung unten***)**. Daraufhin wird ein mit dem ausgewählten Makro verknüpfter Befehl zum Listenfeld *Zugewiesene Befehle* hinzugefügt.



Um das Symbol eines zuwiesenen Befehls zu bearbeiten, wählen Sie den Befehl aus und klicken Sie auf die Schaltfläche **Symbol bearbeiten**. Um einen zugewiesenen Befehl zu löschen, klicken Sie auf die Schaltfläche **Entfernen.**

Platzieren eines mit einem Marko verknüpften Befehls in ein Menü oder eine Symbolleiste

Es gibt zwei Möglichkeiten, um einen mit einem Makro verknüpften Befehl in ein Menü oder eine Symbolleiste zu platzieren:

- Ziehen Sie den Befehl aus dem Listenfeld "Zugewiesene Befehle" an die gewünschte Stelle im Menü oder in der Symbolleiste.
- Verwenden Sie die Mechanismen, die im Dialogfeld "Anpassen" auf dem Register "Befehle" 1584 zur Verfügung stehen.

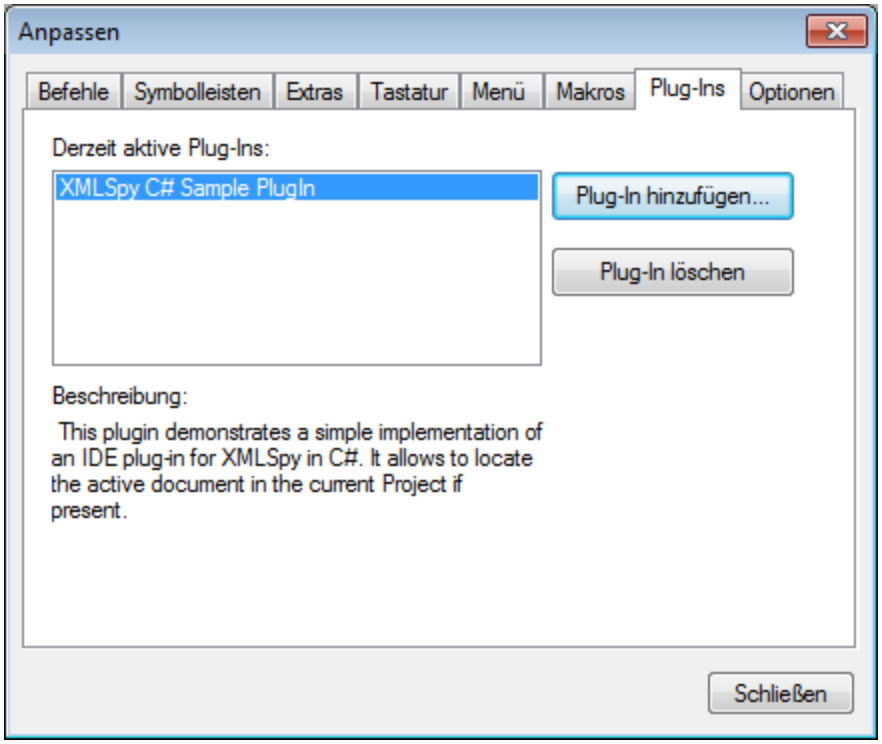
In beiden Fällen wird der Befehl an der gewünschten Stelle erstellt. Wenn Sie im Menü doer in der Symbolleiste auf den Befehl klicken, wird das Makro ausgeführt.

Anmerkung: Wenn ein Makro als "zugewiesener Befehl" ausgewählt wurde, können Sie dafür ein Tastaturkürzel definieren 1589. Wählen Sie auf dem Register "Tastatur" des Dialogfelds "Anpassen" 1589 in der Auswahlliste *Kategorie* die Kategorie *Makros* aus, klicken Sie auf das gewünschte Makro und definieren Sie das Tastenkürzel. Sie müssen ein Makro zuerst als "zugewiesenen Befehl" definieren, damit für Makros Tastenkürzel definiert werden können.

© 2016-2022 Altova GmbH

28.17.13.7 Plug-Ins

Auf dem Register **Plug-Ins** können Sie Plug-Inszu integrieren und entsprechend programmierte Befehle in ein Applikationsmenü und/oder eine Symbolleiste einzufügen. Klicken Sie auf dem Register "Plug-In" (*Abbildung unten*) auf **Plug-In hinzufügen** und navigieren Sie zur DLL-Datei des Plug-In (*siehe "Erstellen von Plug-ins" weiter unten*). Klicken Sie auf **OK**, um das Plug-In hinzuzufügen. Es können auch mehrere Plug-Ins hinzugefügt werden.



Nachdem ein Plug-In erfolgreich hinzugefügt worden ist, erscheint eine Beschreibung des Plug-In im Dialogfeld und die Schaltfläche **Plug-In löschen** wird aktiv. Wenn mit dem Plug-In Code Symbolleisten und Menüs erstellt werden, sind diese sofort auf der Benutzeroberfläche zu sehen. Um ein Plug-In zu entfernen, klicken Sie auf **Plug-In löschen**.

Erstellen von Plug-Ins

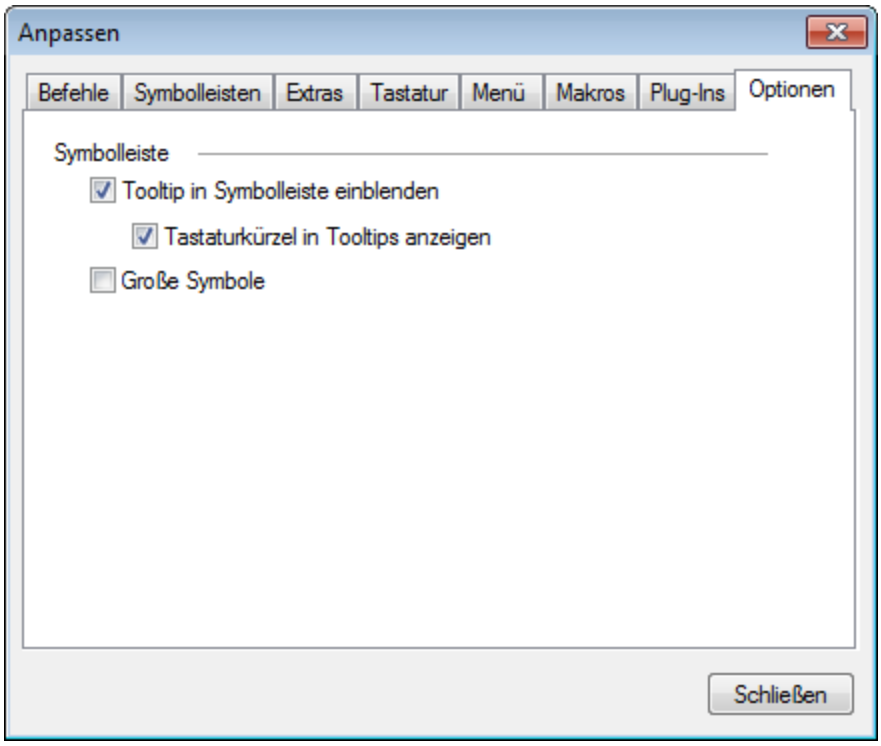
Im Ordner (Eigene) Dateien[34] `Examples\IDEPlugin` der Applikation finden Sie Quellcode für Beispiel-Plug-Ins. Um ein Plug-in anhand von solchem Quellcode zu erstellen, gehen Sie folgendermaßen vor:

1. Öffnen Sie die Lösung, die Sie als Plug-in in Visual Studio erstellen möchten.
2. Erstellen Sie das Plug-in mit dme Befehl im Menü "Build".
3. Die DLL-Datei des Plug-in wird im Ordner `Bin` oder `Debug` erstellt. Diese DLL ist die Datei, die als Plug-in hinzugefügt werden muss (*siehe oben*).

Nähere Informationen zum Erstellen von Plug-Ins finden Sie im Abschnitt IDE Plug-Ins[1690].

28.17.13.8 Optionen

Auf dem Register Optionen können Sie allgemeine Einstellungen für die Umgebung vornehmen.



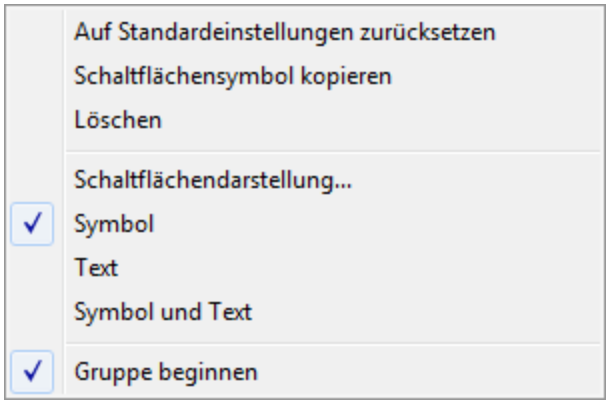
Klicken Sie auf die Kontrollkästchen, um die folgenden Optionen zu aktivieren bzw. zu deaktivieren.

- *Tooltip in Symbolleiste einblenden:* Wenn diese Option aktiviert ist, wird ein Popup beim Ziehen des Mauszeigers über das entsprechende Symbol in der Symbolleiste angezeigt. Das Popup enthält eine Kurzbeschreibung der Funktion, sowie, falls vorhanden und falls die Option *Tastaturkürzel in Tooltips anzeigen* aktiviert ist, ein Tastaturkürzel.
- *Tastaturkürzel in Tooltips anzeigen:* Gibt an, ob Tastaturkürzel in Tooltips angezeigt werden sollen.
- *Große Symbole:* Damit können Sie bei der Größe der angezeigten Symbole zwischen "Standard" und "Groß" wechseln.

28.17.13.9 Kontextmenü 'Anpassen'

Das **Kontextmenü "Anpassen"** (*Abbildung unten*) wird angezeigt, wenn Sie bei geöffnetem Dialogfeld "Anpassen" mit der rechten Maustaste auf ein Applikationsmenü, einen Menübefehl oder eine Symbolleisten-Schaltfläche klicken.

© 2016-2022 Altova GmbH

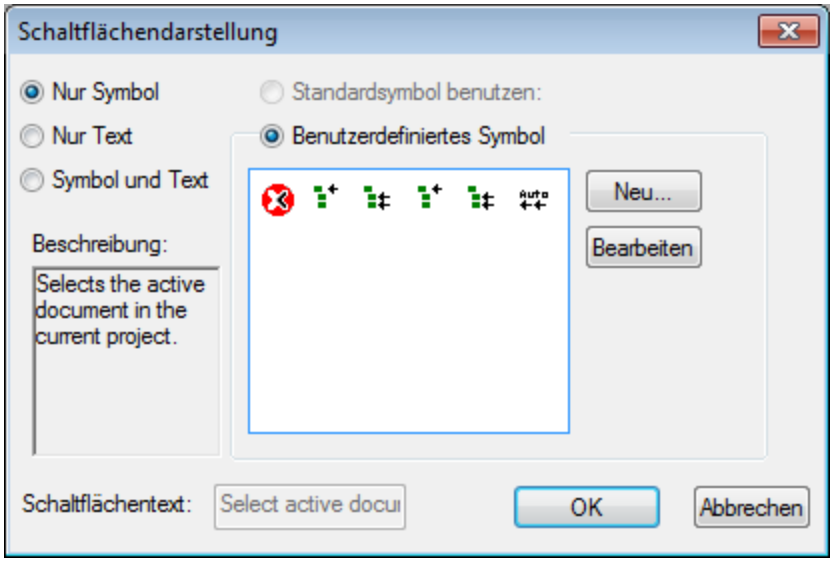


Es stehen die folgenden Funktionalitäten zur Verfügung:

- *Auf Standardeinstellungen zurücksetzen*: Derzeit keine Funktion.
- *Schaltflächensymbol kopieren:* Kopiert das Symbol, auf das Sie mit der rechten Maustaste klicken, in die Zwischenablage.
- *Löschen*: Löscht das ausgewählte Menü, den ausgewählten Menübefehl oder die ausgewählte Symbolleisten-Schaltfläche. Informationen, wie man gelöschte Elemente wiederherstellt, finden Sie weiter unten.
- *Schaltflächendarstellung*: Damit wird das Dialogfeld "Schaltflächendarstellung" (*siehe Abbildung unten*) aufgerufen, in dem Sie die Eigenschaften definieren können, mit denen das Aussehen der ausgewählten Symbolleisten-Schaltfläche definiert wird. Nähere Informationen dazu finden Sie in der Beschreibung unten.
- *Nur Symbol, Nur Text, Symbol und Text:* Einander gegenseitig ausschließende Optionen, mit denen festgelegt wird, ob die ausgewählte Symbolleisten-Schaltfläche nur als Symbol, nur als Text oder als Symbol und Text angezeigt werden soll. Wählen Sie eine dieser Opionen aus, um die Änderung vorzunehmen. Alternativ dazu können Sie diese Änderung auch im Dialogfeld "Schaltflächendarstellung" vornehmen.
- *Gruppe beginnen:* Fügt links von der ausgewählten Symbolleisten-Schaltfläche eine vertikale Trennlinie für die Gruppe ein. Dadurch wird die ausgewählte Symbolleisten-Schaltfläche zur ersten in einer Gruppe von Schaltflächen.

Das Dialogfeld "Schaltflächendarstellung"

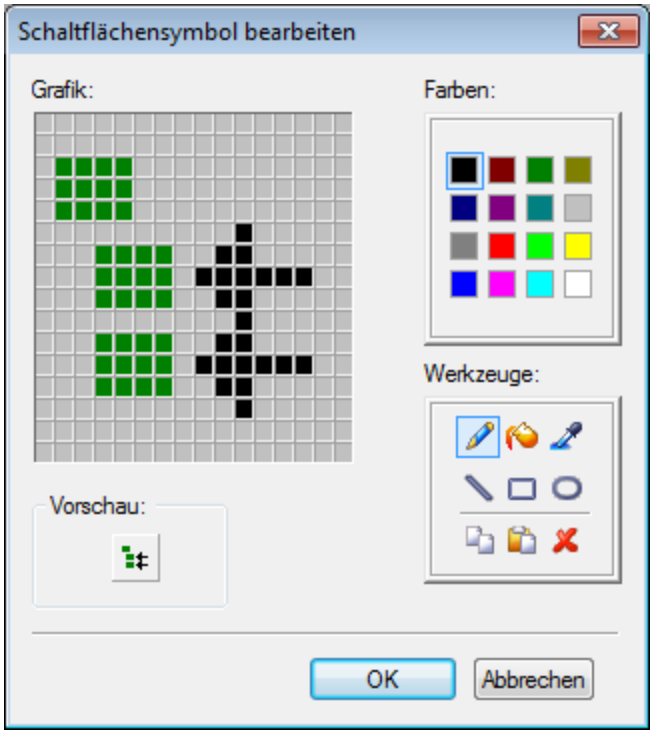
Klicken Sie mit der rechten Maustaste auf eine Symbolleisten-Schaltfläche und wählen Sie den Befehl **Schaltflächendarstellung** um das gleichnamige Dialogfeld aufzurufen (*Abbildung unten*). Über dieses Dialogfeld können Sie das Symbol der Symbolleisten-Schaltfläche sowie seinen Text bearbeiten. Zur Zeit können nur Symbolleisten-Schaltflächen für Makros und solche von Plug-Ins über dieses Dialogfeld bearbeitet werden.



Für die ausgewählte Symbolleisten-Schaltfläche (für die durch Rechtsklick das Kontextmenü aufgerufen wurde) stehen die folgenden Bearbeitungsmöglichkeiten zur Verfügung:

- *Nur Symbol, Nur Text, Symbol und Text:* Aktivieren Sie das gewünschte Optionsfeld, um festzulegen, wie die Symbolleisten-Schaltfläche aussehen soll.
- *Bearbeitung des Symbols:* Bei Auswahl der Option *Nur Symbol* oder *Symbol und Text* stehen die Optionen zur Bildbearbeitung zur Verfügung. Klicken Sie auf **Neu,** um ein neues Symbol zu erstellen, das zu den benutzerdefinierten Symbolen im Bereich "Benutzerdefinierte Symbole" hinzugefügt wird. Wählen Sie ein Symbol aus und klicken Sie auf **Bearbeiten** um es zu bearbeiten.

© 2016-2022 Altova GmbH



- *Auswahl des Symbols:* Wählen Sie im Bereich der Symbole ein Symbol aus und klicken Sie auf OK, um als ausgewählte Symbol als Symbolleisten-Schaltfläche zu verwenden.
- *Textbearbeitung und Auswahl:* Bei Auswahl der Option *Nur Text* oder *Symbol und Text* wird das Textfeld *Schaltflächentext* aktiviert. Geben Sie den Text ein oder bearbeiten Sie ihn und klicken Sie auf **OK,** um diesen Text als Beschriftung der Symbolleisten-Schaltfläche anzuzeigen.

Anmerkung: Über das Dialogfeld "Schaltflächendarstellung" kann auch der Text von Menübefehlen bearbeitet werden. Klicken Sie (während das Dialogfeld "Anpassen" geöffnet ist) mit der rechten Maustaste auf den Menübefehl, klicken Sie auf **Schaltflächendarstellung** und bearbeiten Sie anschließend den Text des Menübefehls im Textfeld *Schaltflächentext.*

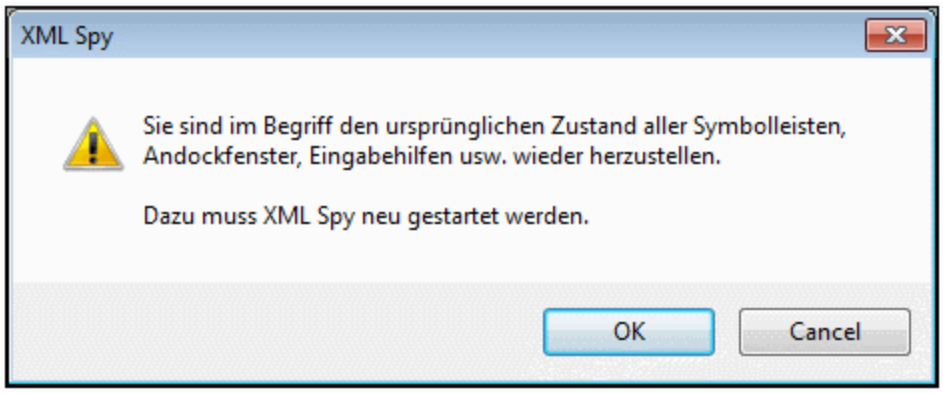
Wiederherstellen von gelöschten Menüs, Menübefehlen und Symbolleisten-Schaltflächen

Wenn ein Menü, ein Menübefehl oder eine Symbolleisten-Schaltfläche mit dem Befehl **Löschen** (Kontextmenü "Anpassen") gelöscht wurde, kann der Befehl folgendermaßen wiederhergestellt werden:

- *Menüs:* Gehen Sie zu **Extras | Anpassen | Menü** 1593 und klicken Sie im Bereich *Applikationsmenüs* auf die Schaltfläche **Zurücksetzen.** Gehen Sie alternativ dazu zu **Extras | Anpassen | Symbolleisten** 1586, wählen Sie die Menüleiste aus und klicken Sie auf die Schaltfläche **Zurücksetzen.-**
- *Menübefehle:* Gehen Sie zu **Extras | Anpassen | Befehle** 1584 und ziehen Sie den Befehl aus dem Listenfeld "Befehle" in das Menü.
- *Symbolleisten-Schaltflächen:* Gehen Sie zu **Extras | Anpassen | Befehle** 1584 und ziehen Sie den Befehl aus dem Listenfeld "Befehle" in die Symbolleiste.

28.17.14 Symbolleisten und Fenster wiederherstellen

Mit diesem Befehl wird XMLSpy geschlossen und mit den Standardeinstellungen neu gestartet. Vor dem Schließen wird ein Dialogfeld angezeigt, in dem Sie gebeten werden, den Befehl zu bestätigen (*Abbildung unten*).

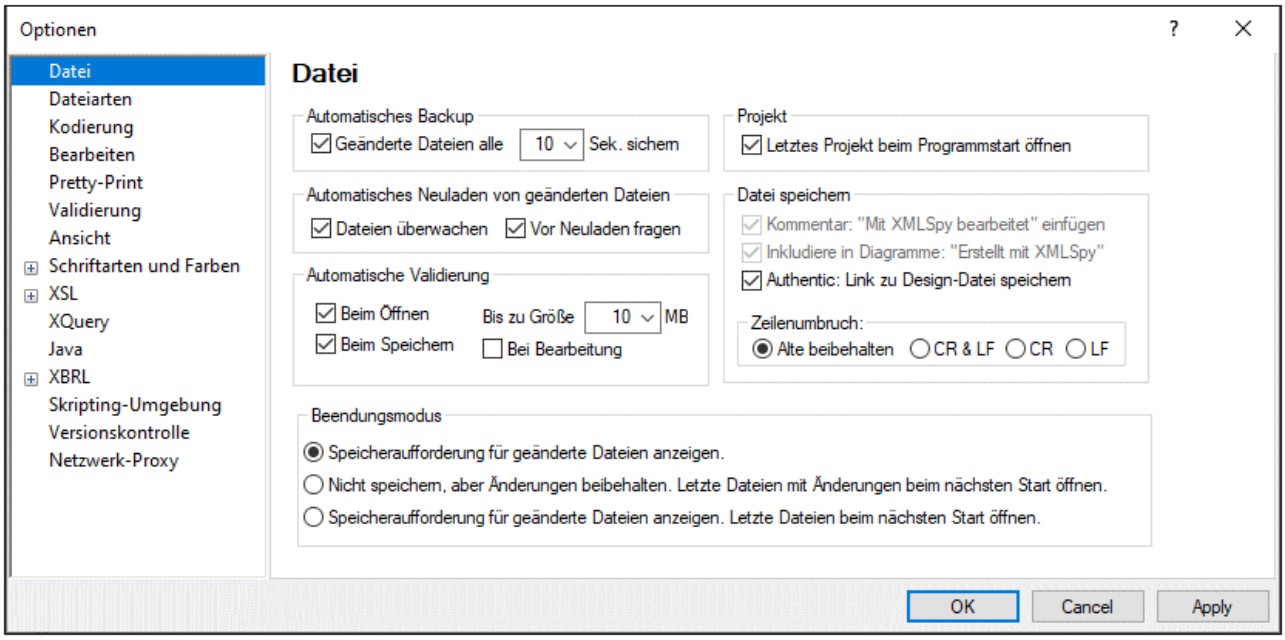


Dieser Befehl ist hilfreich, wenn Sie Symbolleisten oder Fenster verschoben oder ausgeblendet oder deren Größe angepasst haben und nun alle Originaleinstellungen wiederherstellen möchten.

28.17.15 Optionen

Mit dem Befehl **Extras** | **Optionen** können Sie globale Programmeinstellungen vornehmen. Diese Einstellungen sind in Abschnitte gegliedert (*siehe linker Bereich in der Abbildung unten*). So enthält etwa der Abschnitt "Datei" 1603 (*siehe Abb. unten)* Optionen, mit denen Sie festlegen, wie XMLSpy Dateien öffnen und speichern soll. Um die Optionen eines bestimmten Abschnitts zu definieren, wählen Sie diesen Abschnitt im linken Bereich aus und definieren Sie die gewünschten Eigenschaftswerte. Durch Klicken auf die Schaltfläche **OK** speichern Sie die Änderungen in der Registry und schließen das Dialogfeld. Durch Klicken auf die Schaltfläche **Übernehmen** werden die Änderungen in den derzeit offenen Dokumenten angezeigt.

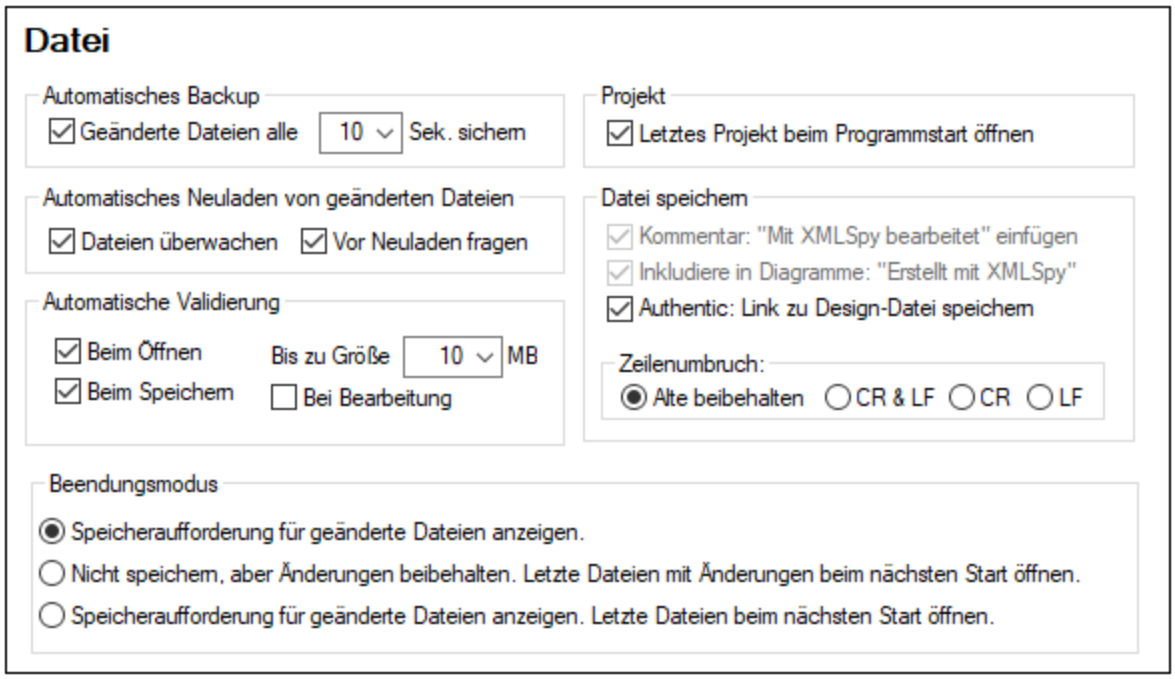
© 2016-2022 Altova GmbH



Die einzelnen Abschnitte des Dialogfelds "Optionen" werden in den Unterabschnitten dieses Abschnitts näher beschrieben.

28.17.15.1 Datei

Im Abschnitt "**Datei**" wird die Art definiert, wie XMLSpy Ihre XML-Dokumente öffnet und speichert. Weitere Einstellungen hierzu finden Sie im Abschnitt Kodierung[1608].



Automatisches Backup

Gerade in Bearbeitung befindliche Dateien werden automatisch gesichert, wenn diese Option aktiviert ist. Sie können aus der Auswahlliste ein Backup-Intervall zwischen 5 und 60 Sekunden auswählen oder einen benutzerdefinierten Wert bis zu 300 Sekunden eingeben. Nähere Informationen dazu finden Sie im Abschnitt *Automatisches Backup von Dateien*[142].

Automatisches Neuladen von geänderten Dateien

Wenn Sie in einer Multi-User Umgebung, oder mit Dateien arbeiten, die von Servern dynamisch erstellt werden, können Sie die Änderungen während des Bearbeitens direkt verfolgen. Jedes Mal, wenn XMLSpy eine Änderung in einem offenen Dokument bemerkt, werden Sie gefragt, ob die geänderte Datei neu geladen werden soll.

Automatische Validierung

Wenn Sie DTDs oder XML-Schemas zur Definition der Struktur Ihrer XML-Dokumente verwenden, können Sie Ihre Instanzdokumente in den folgenden Situationen automatisch validieren:

- beim Öffnen der Datei, wenn die Datei unter einer von Ihnen in MB definierten Größe ist
- beim Speichern der Datei
- beim Bearbeiten der Datei Wenn diese Option aktiviert ist, wird das Dokument während der Eingabe in der Text-[144] oder JSON-Grid-Ansicht[701] validiert. Nähere Informationen dazu finden Sie unter XML Validierung in der Textansicht[339].

Wenn das Dokument nicht gültig ist, wird eine Fehlermeldung angezeigt. Wenn es gültig ist, wird keine Meldung angezeigt und die Operation wird ohne weitere Benachrichtigung fortgesetzt.

© 2016-2022 Altova GmbH

Projekt

Beim Starten von XMLSpy können Sie das zuletzt verwendete Projekt automatisch anzeigen lassen.

Datei speichern

Beim Speichern eines XML-Dokuments fügt XMLSpy im oberen Bereich des Dokuments den kurzen Kommentar `<!-- Mit XMLSpy bearbeitet http://www.altova.com -->` ein. Dieser Kommentar kann nur von lizenzierten Benutzern deaktiviert werden und wird beim Bearbeiten oder Speichern von Dateien in der Grid-Ansicht oder der Schema-Ansicht verwendet.

Wenn Sie ein Content Model-Diagramm speichern (über die Menüoption **Schema-Design | Dokumentation generieren**), inkludiert XMLSpy das XMLSpy Logo. Diese Option kann nur von lizenzierten Benutzern deaktiviert werden.

Wenn ein StyleVision Power Stylesheet mit einer XML-Datei verknüpft ist, wird der Link zum StyleVision Power Stylesheet bei Auswahl der Option 'Authentic: Link in Design-Datei speichern' mit der XML-Datei gespeichert.

Zeilenumbrüche

Wenn Sie eine Datei öffnen, wird die Zeichenkodierung für Zeilenumbrüche beibehalten, wenn die Option **Alte beibehalten** aktiviert ist. Alternativ dazu können Sie festlegen, dass Zeilenumbrüche in einer dieser drei Kodierungsarten kodiert werden: **CR&LF** (für PC), **CR** (für MacOS) oder **LF** (für Unix).

Beendungsmodus

Mit diesen Optionen legen Sie fest, wie Dateien, die bei der Beendung von XMLSpy geöffnet sind, behandelt werden sollen. Es stehen die folgenden Optionen zur Verfügung:

- *Speicheraufforderung für geänderte Dateien anzeigen*: Wenn eine offene Datei nicht gespeicherte Änderungen enthält, wird eine Meldung angezeigt, in der Sie gefragt werden, ob Sie die Dateiänderungen speichern möchten. Je nach Antwort wird die Datei entweder gespeichert oder nicht gespeichert, bevor das Programm anschließend beendet wird.
- *Speicheraufforderung für geänderte Dateien anzeigen. Letzte Dateien beim nächsten Start öffnen*: Für offene Dateien, die nicht gespeicherte Änderungen enthalten, wird das Speicherdialogfeld angezeigt. Der Benutzer kann eine oder mehrere geänderte Dateien speichern oder diese nicht speichern. Beim Neustart des Programms werden alle bei der Beendung des Programms geöffneten Dateien erneut geöffnet. (Wenn Änderungen nicht gespeichert wurden, gehen sie verloren.)
- *Nicht speichern, aber Änderungen beibehalten. Letzte Dateien mit Änderungen beim nächsten Start öffnen*: Das Programm wird sofort beendet, ohne dass nicht gespeicherte Änderungen gespeichert werden. Beim Neustart des Programms werden alle bei der Beendung des Programms geöffneten Dateien geöffnet und enthalten die nicht gespeicherten Änderungen und Sie können dort weiterarbeiten, wo sie zuletzt aufgehört haben.

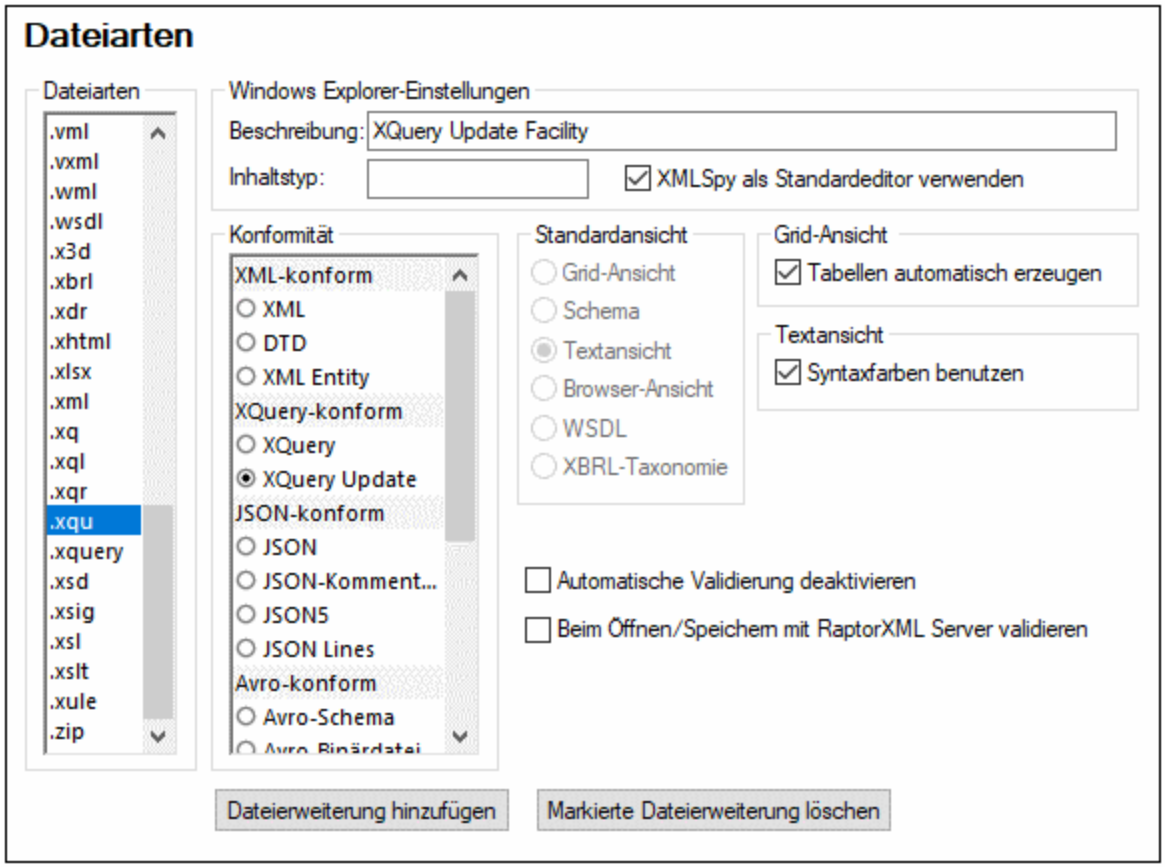
Wenn Sie das Programm zum ersten Mal beenden, werden Ihnen die Beendungsmodusoptionen angezeigt, sodass Sie das gewünschte Verhalten auswählen können. Diese Option stehen im Dialogfeld "Optionen" im Abschnitt "Datei" zur Verfügung.

Speichern und Beenden

Klicken Sie zum Abschluss auf **OK**.

28.17.15.2 Dateiarten

Im Abschnitt **Dateiarten** (*Abbildung unten*) können Sie für XMLSpy verschiedene Einstellungen für die einzelnen Dateiarten festlegen.



Wählen Sie eine Dateiart im Listenfeld "Dateiarten" aus und passen Sie anschließend die Funktionen für die ausgewählte Dateiart, wie unten beschrieben, an. Beachten Sie, dass die Liste "Dateiarten" zwei Sondereinträge enthält:

- <*Standard*>: Damit können Sie festlegen, wie Dateien mit einer Erweiterung, die nicht in der Liste der Dateiarten enthalten ist, behandelt werden sollen.
- <*keine*>: Damit können Sie festlegen, wie Dateien, die keine Erweiterung haben, behandelt werden sollen.

Windows Explorer-Einstellungen

Sie können die Beschreibung der Dateiart und den MIME-konformen Inhaltstyp, der von Windows Explorer verwendet wird, ändern. Weiters können Sie definieren, ob XMLSpy als der standardmäßige Editor für diese Dateiart verwendet werden soll.

© 2016-2022 Altova GmbH

Konformität

XMLSpy bietet für verschiedene Dateiarten spezifische intelligente Editierfunktionen und andere Funktionen. Die Funktionalitäten für eine bestimmte Dateiart werden von XMLSpy auf Basis der in dieser Option von Ihnen gewählten Konformität konfiguriert. So sind etwa Dateien mit der Dateierweiterung `.xqu` (Abbildung oben) als XQuery Update-konform definiert. XMLSpy öffnet `.xqu`-Dateien daher mit XQuery Update-Bearbeitungsfunktionen. Sie haben in XMLSpy die Wahl zwischen den folgenden Konformitätsoptionen: XML, XQuery[514], ZIP[961], JSON[676], Avro[779], andere Formate. Bei der XML-Konformität wird weiters zwischen XML-, DTD- und XML Entity-Dateitypen unterschieden. Bei JSON-konformen Dateien wird zwischen einfachen JSON-Dateien oder Avro-Schemas unterschieden. Mit der Option *Avro-konform* werden Funktionen für Avro-Binärdateien aktiviert. Vielen Dateiarten ist bereits eine für den jeweiligen Typ entsprechende Standardkonformität zugewiesen. Wir empfehlen Ihnen, diese Einstellungen nicht zu ändern, außer Sie fügen einen neuen Dateityp hinzu oder Sie möchten die Art der Konformität bei einem bestimmten Dateityp absichtlich ändern.

Standardansicht

In dieser Gruppe definieren Sie, welche Standardansicht für welche Dateiart verwendet werden soll. Wenn eine bestimmte Konformität nur in einer Ansicht angezeigt werden kann, so ist diese Ansicht standardmäßig ausgewählt und die Auswahl der Ansichten ist deaktiviert. XQuery Update-Dokumente können z.B. nur in der Textansicht angezeigt werden, daher ist diese Ansicht standardmäßig ausgewählt und die Auswahl der Ansicht ist deaktiviert; auf ähnliche Art können auch Avro-konforme Dokumente (Avro-Binärdateien) nur in der Grid-Ansicht angezeigt werden.

Grid-Ansicht

Dieses Kontrollkästchen definiert, ob in der Grid-Ansicht automatisch Tabellen erzeugt werden sollen.

Textansicht

In diesem Textfeld können Sie die Syntaxfärbung für bestimmte Dateiarten bestimmen.

Automatische Validierung deaktivieren

Mit dieser Option können Sie für bestimmte Dateiarten die automatische Validierung deaktivieren. Die automatische Validierung wird normalerweise beim Öffnen oder Speichern einer Datei oder beim Wechseln der Ansicht durchgeführt.

Beim Öffnen/Speichern mit RaptorXML Server validieren

Definiert, ob zum Validieren des ausgewählten Dateityps RaptorXML Server verwendet werden soll, wenn die Datei geöffnet oder gespeichert wird. Dazu muss RaptorXML Server zuerst eingerichtet und konfiguriert werden[1579].

Neue Dateierweiterung hinzufügen

Fügt eine neue Dateiart zur Liste der Dateiarten hinzu. Sie müssen anschließend die Einstellungen für diesen neuen Dateityp über die Optionen auf diesem Register definieren.

Markierte Dateierweiterung löschen

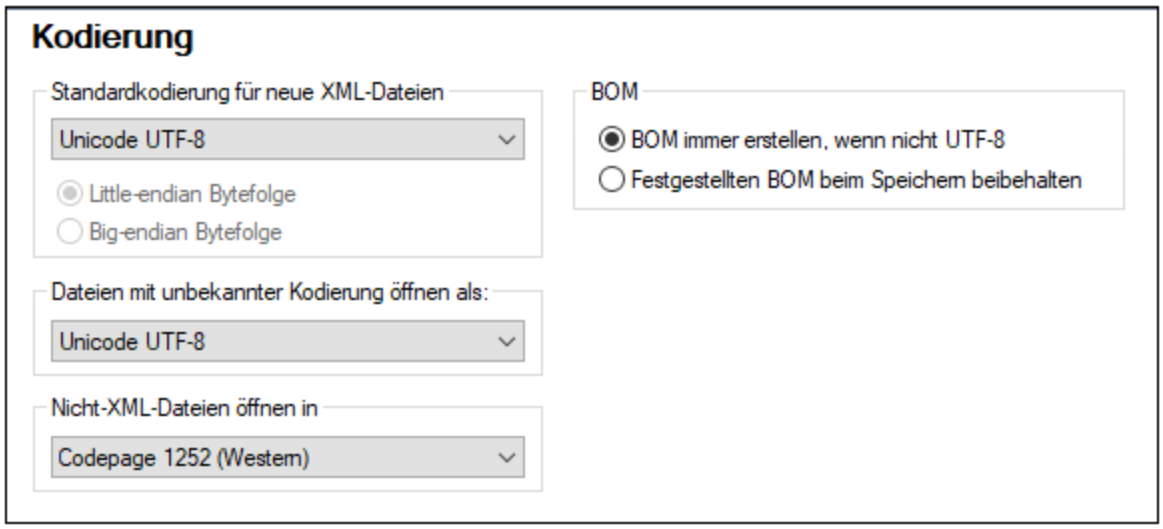
Löscht die aktuell ausgewählte Dateiart sowie alle damit in Zusammenhang stehenden Einstellungen.

Speichern und Beenden

Klicken Sie zum Abschluss auf **OK**.

28.17.15.3 Kodierung

Im Abschnitt **Kodierung** definieren Sie die Einstellungen für die Dateikodierung.



Standardkodierung für neue XML-Dateien

Die Standardkodierung für neue Dateien kann im Dialogfeld "Optionen" durch Auswahl einer Option aus der Dropdown-Liste definiert werden, sodass jedes neue Dokument automatisch mit der entsprechenden XML-Deklaration und der hier gewählten Kodierungsspezifikation erstellt wird. Wird eine 2- oder 4-Byte-Kodierung als standardmäßige Kodierung ausgewählt (z.B. UTF-16, UCS-2 oder UCS-4) können Sie weiters wählen, ob die Byteordnung für XML-Dokumente Little Endian oder Big Endian sein soll.

Die Kodierung für existierende Dateien wird immer beibehalten und kann nur mit dem Befehl **Datei | Kodierung**[1285] geändert werden.

Dateien mit unbekannter Kodierung öffnen als

Sie können in dieser Auswahlliste auswählen, mit welcher Kodierung eine Datei geöffnet werden soll, die keine Kodierungsspezifikation enthält oder deren Kodierung nicht automatisch bestimmt werden konnte.

© 2016-2022 Altova GmbH

Nicht-XML-Dateien öffnen in

In dieser Auswahlliste können Sie auswählen, welche Kodierung für nicht XML-Dateien verwendet werden soll. Die können die Kodierung des Dokuments mit dem Befehl **Datei | Kodierung**[1285] ändern.

BOM (Byte Order Mark)

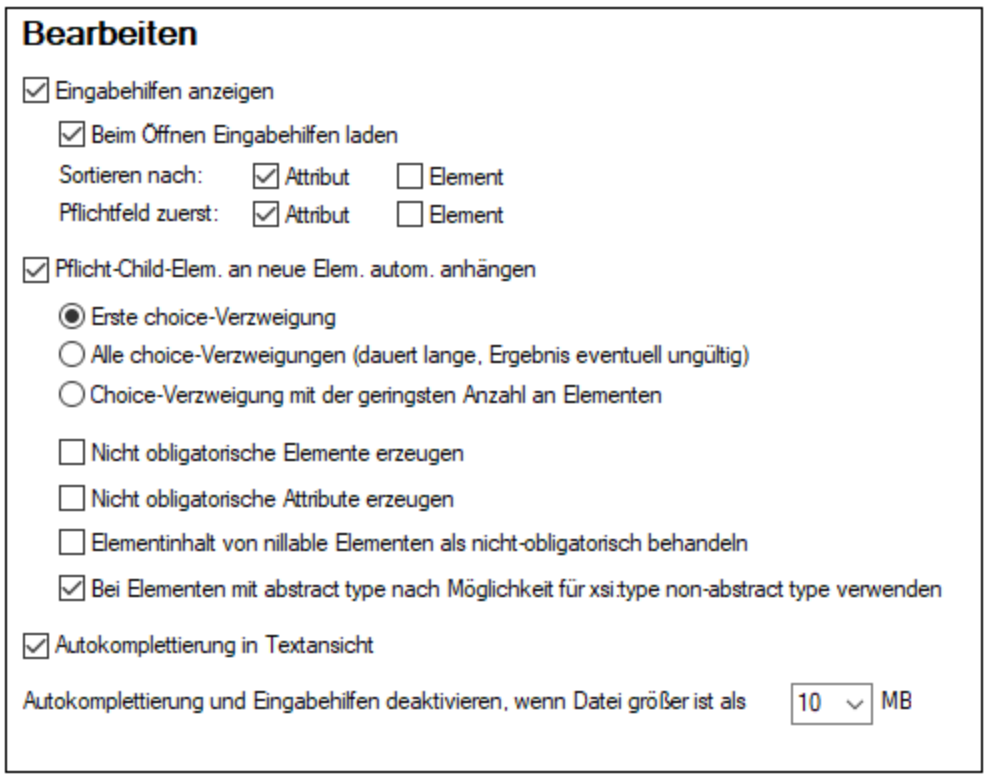
Bei Speicherung eines Dokuments mit einer 2-Byte- oder einer 4-Byte-Zeichenkodierung kann es entweder mit (i) der Bytefolge Little Endian und einer Little Endian BOM gespeichert werden (*BOM immer erstellen, wenn nicht UTF-8*); oder (ii) mit der ermittelten Bytefolge und der ermittelten BOM (*Festgestellten BOM beim Speichern beibehalten*).

Speichern und Beenden

Klicken Sie zum Abschluss auf **OK**.

28.17.15.4 Bearbeiten

Im Abschnitt "**Bearbeiten**" können Sie Bearbeitungseinstellungen für XMLSpy definieren.



Eingabehilfen

XMLSpy bietet Ihnen auf Basis dieser Einstellungen intelligente Editier-Hilfen zum Bearbeiten von Dokumenten. Verschiedene Bearbeitungseinstellungen für diese Eingabehilfen, darunter auch die Reihenfolge, in der Elemente in den Eingabehilfen angezeigt werden, lassen sich hier anpassen. Die hier vorgenommenen Einstellungen werden angewendet, wenn sie für den bearbeiteten Dateityp relevant sind. So kann z.B. die Option zum Laden der Eingabehilfen beim Öffnen der Datei und zum Sortieren der Attribute auf DTD- oder XQuery-Dokumente nicht angewendet werden.

Erstellen der XML-Struktur anhand eines XML-Schemas

Wenn Sie ein neues XML-Dokument erstellen, das auf einem XML-Schema basiert, wird das Dokument mit einer von den Definitionen im Schema abgeleiteten Struktur generiert. Mit den unten beschriebenen Einstellungen werden einige nicht eindeutige Aspekte im Zusammenhang mit der Erstellung der Struktur festgelegt.

Pflicht-Child-Elemente an neue Elemente automatisch anhängen
Obligatorische Child-Elemente von `choice`-Gruppen werden auf Basis der Einstellungen in diesem Bereich automatisch angehängt. Sie können auswählen, ob (i) die erste Verzweigung (das erste Element) der `choice`-Gruppe, (ii) alle Verzweigungen oder (iii) die Verzweigung mit der geringsten Anzahl an Nachfahrenelementen generiert werden soll. Beachten Sie, dass bei Auswahl von *Alle Verzweigungen* ein ungültiges Dokument erzeugt werden könnte, da nur eine Verzweigung aus einer `choice`-Gruppe zulässig ist.

Nicht obligatorische Nodes und Elemente eines abstrakten Typs
Um nicht obligatorische Elemente oder Attribute hinzuzufügen, aktivieren Sie die entsprechende Option. Wenn diese Optionen nicht aktiviert sind, werden nur obligatorische Nodes hinzugefügt. Sie können auch (i) Elementinhalt von nillable (auf Null setzbaren) Elementen als nicht obligatorisch definieren und (ii) versuchen, einen nicht abstrakten Typ als `xsi:type` eines Elements eines abstrakten Typs zu verwenden.

Textansicht

Mit der Option *Autokomplettierung* werden eindeutige Strukturkomponenten automatisch hinzugefügt. Wenn z.B. die schließende eckige Klammer des Start-Tags eines Elements eingegeben wird, so wird der End-Tag dieses Elements automatisch hinzugefügt, wenn diese Option aktiviert ist.

Die Autokomplettierung und die Eingabehilfen können in der Textansicht deaktiviert werden, wenn eine Datei die in der Auswahlliste *Autokomplettierung deaktivieren...* definierte Größe überschreitet. Diese Option ist nützlich, wenn Sie die Bearbeitungsgeschwindigkeit bei großen Dateien erhöhen möchten und die Autokomplettierungsfunktion sowie die Eingabehilfen nicht benötigen. Wenn die Datei die für diese Option definierte Größe übersteigt, so enthält das Kontextmenü der Textansicht einen Ein/Aus-Befehl zum Aktivieren bzw. Deaktivieren der Autokomplettierung und der Eingabehilfen. Dadurch können Sie diese Bearbeitungshilfen jederzeit während der Bearbeitung ein- und ausschalten (wenn Dateien die für diese Option definierte Größe überschreiten). Wenn der für diese Option definierte Wert kleiner als die Größe der geöffneten Datei ist, entsprechen in Fehlermeldungen angezeigte Stellen nicht genau der Stelle in der Textansicht.

Speichern und Beenden

Klicken Sie zum Abschluss auf **OK**.

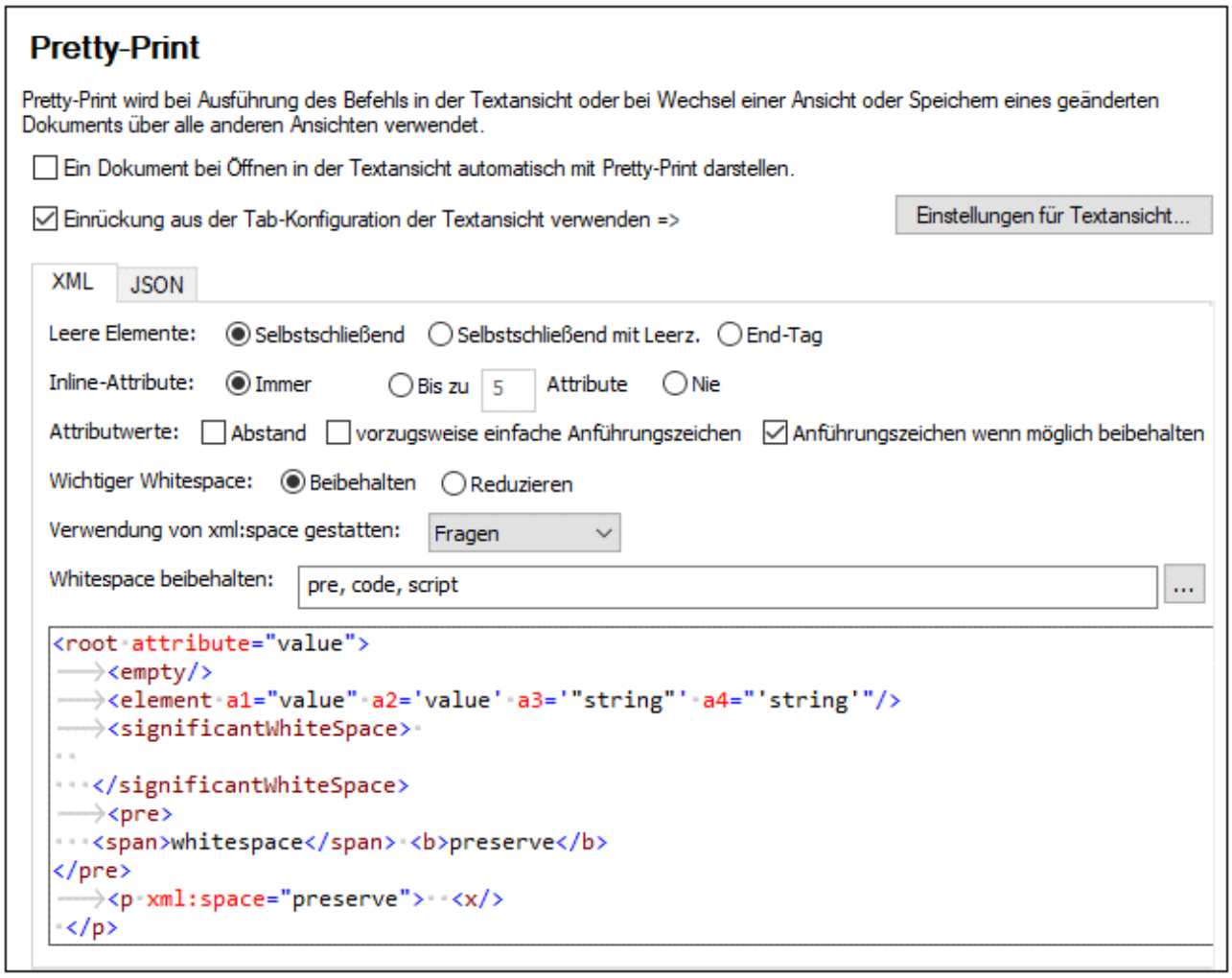
© 2016-2022 Altova GmbH

28.17.15.5 Pretty-Print

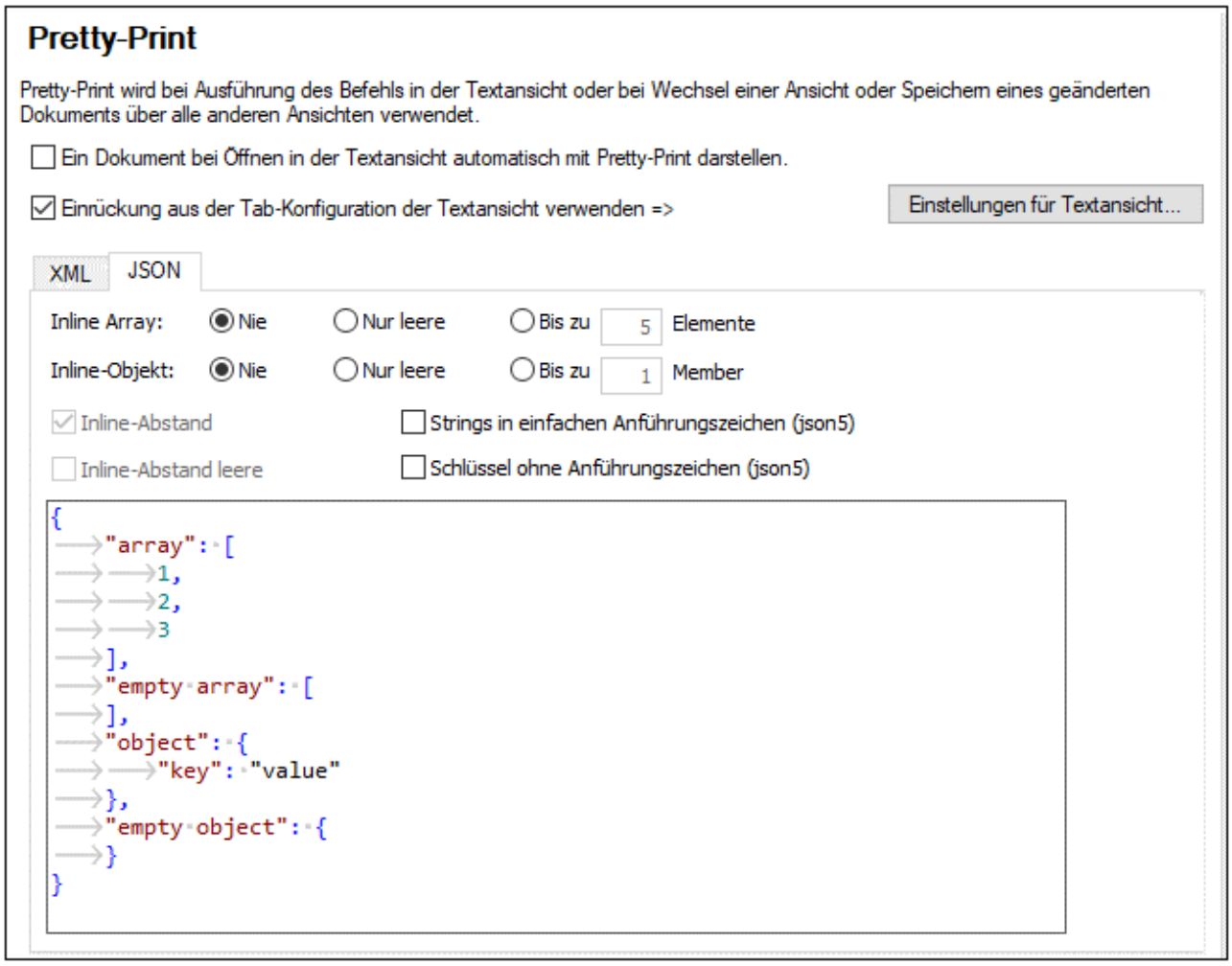
Im Abschnitt **Pretty-Print** *(siehe Abbildungen unten)* können Sie festlegen, wie Text in der Textansicht angezeigt werden soll. Die Definitionen in diesem Abschnitt werden in die folgenden Kategorien eingeteilt:

- XML-Einstellungen (Klicken Sie auf das Register "XML", um diese Einstellungen aufzurufen *Abbildung links unten*)
- JSON-Einstellungen (Klicken Sie auf das Register "JSON", um diese Einstellungen aufzurufen *Abbildung rechts unten*)
- Einstellungen für Textansicht (Klicken Sie auf diese Schaltfläche, um das Einstellungsdialogfeld aufzurufen)

Mit dem Kontrollkästchen *Einrückung verwenden* wird die Pretty-Print[145]-Darstellung aktiviert bzw. deaktiviert. Über das Kontrollkästchen *Ein Dokument bei Öffnen in der Textansicht automatisch mit Pretty-Print darstellen* kann ein Dokument beim Laden automatisch in der Pretty-Print-Darstellung angezeigt werden.



 © 2016-2022 Altova GmbH



XML-Einstellungen

Es stehen die folgenden Einstellungen zur Verfügung:

- wie leere Elemente im Dokument geschrieben und angezeigt werden sollen: mit einem Tag (*selbstschließend*) oder mit zwei Tags (*End-Tag*).
- ob Attribute inline (in derselben Zeile wie ihr Parent-Element) angezeigt werden sollen oder nicht. Attribute werden inline angezeigt, wenn *Immer* oder *Bis zu X Attribute* ausgewählt ist und die Anzahl der Attribute X nicht übersteigt. Attribute werden in neuen Zeilen angezeigt, wenn *Nie* oder *Bis zu X Attribute* ausgewählt ist und die Anzahl der Attribute X übersteigt.
- wie Attributwerte geschrieben werden: (i) mit Leerzeichen vor und nach dem Ist-gleich-Zeichen oder nicht; (ii) ob Werte in einfache oder doppelte Anführungszeichen gesetzt werden sollen; (iii) ob Anführungszeichen im Quelltext so, wie von Ihnen eingegeben, beibehalten werden sollen, oder ob sie durch andere Optionen (wie z.B. *vorzugsweise einfache Anführungszeichen*) außer Kraft gesetzt werden sollen; beachten Sie, dass Anführungszeichen bei Auswahl der Option *Anführungszeichen wenn möglich beibehalten* nur soweit beibehalten werden, sofern das Dokument dadurch nicht ungültig wird.

- ob Whitespace-Zeichen beibehalten oder reduziert werden sollen. Die Whitespace-Zeichen sind: Leerzeichen, Tabulator, Wagenrücklauf (Carriage Return) und Zeilenvorschub (Line Feed). Nähere Informationen dazu finden Sie im Abschnitt *Whitespace*[341].
- Wenn Elemente in einem Dokument das Attribut-Wert-Paar **`xml:space="preseve"`** enthalten, können Sie mit der Einstellung *Verwendung von xml:space gestatten* festlegen, wie dieses Attribut-Wert-Paar bei Pretty-Print behandelt werden soll. Mit der Option *Immer* definieren Sie, dass der Anweisung des Attributs beim Pretty-Printing eines jeden Dokuments immer Folge zu leisten ist: Whitespace-Zeichen in den entsprechenden Elementen werden beibehalten und an diesen Elemente wird kein Pretty-Print durchgeführt. Bei Auswahl der Option *Nie* wird das *xml:space*-Attribut ignoriert und die entsprechenden Elemente werden mit Pretty-Print dargestellt. Bei Auswahl der Option *Fragen* fragt XMLSpy jedes Mal, wenn an einem Dokument, das **`xml:space="preseve"`** enthält, Pretty-Print durchgeführt werden soll, was getan werden soll.
- Einstellung, in welchen Elementen Whitespaces beibehalten werden sollen.
- Ob die im Dialogfeld Einstellungen für Textansicht[1506] definierte Einrückung verwendet werden soll oder ob keine Einrückung verwendet werden soll (wird über das Kontrollkästchen oberhalb des Bereichs definiert).

Wenn Sie eine Option auswählen, sehen Sie die Auswirkung davon im Vorschaufenster am unteren Rand des Dialogfelds. Hier sehen Sie, wie sich Ihre Einstellungen auswirken, bevor Sie diese mit **OK** bestätigen.

JSON-Einstellungen

Für jede Option, die Sie auswählen, wird ihre Auswirkung sofort im Vorschaubereich am unteren Rand des Dialogfelds angezeigt.

- *Inline-Array:* Zeigt die Elemente eines Array in einer einzigen Zeile (oder inline) an. Andernfalls befinden sich die Elemente eventuell in separaten Zeilen. Sie können auswählen, ob Inline-Formatierung auf Arrays nie, auf leere Arrays oder auf Arrays bis zu einer bestimmten Größe angewendet werden soll.
- *Inline-Objekt:* Zeigt die Eigenschaften eines Objekts in einer einzigen Zeile (oder inline) an. Andernfalls befinden sich die Eigenschaften eventuell in separaten Zeilen. Sie können auswählen, ob Inline-Formatierung auf Objekte nie, auf leere Objekte oder auf Objekte bis zu einer bestimmten Größe angewendet werden soll.
- *Inline-Abstand:* Wenn diese Option aktiviert ist, wird ein Abstand zwischen den Elementen von nicht leeren Inline Arrays und nicht leeren Inline-Objekten eingefügt. Die Option ist nur aktiv, wenn entweder die Option für nicht leere Inline Arrays oder nicht leere Inline-Objekte aktiviert ist.
- *Inline-Abstand leere:* Wenn diese Option aktiviert ist, wird ein Abstand zwischen den Trennzeichen von leeren Inline Arrays und leeren Inline-Objekten eingefügt. Die Option ist nur aktiv, wenn entweder ein Array oder Objekt als Inline definiert wurde (leer oder nicht leer).
- *Strings in einfachen Anführungszeichen (JSON5):* Wenn diese Option aktiviert ist, werden alle Anführungszeichen in JSON5-Dokumenten in einfache Anführungszeichen konvertiert.
- *Schlüssel ohne Anführungszeichen (JSON5):* Wenn diese Option aktiviert ist, werden in JSON5-Dokumenten alle Anführungszeichen rund um Schlüssel (von `Wert:Schlüssel`-Paaren) entfernt.

Über das Kontrollkästchen oberhalb des Bereichs können Sie festlegen, ob die im Dialogfeld Einstellungen für Textansicht[1506] definierte Einrückung oder ob keine Einrückung verwendet werden soll.

Einstellungen für Textansicht

Klicken Sie auf **Einstellungen für Textansicht**, um das Dialogfeld "Einstellungen für Textansicht" zu öffnen. Hier können Sie die Eigenschaften der Textansicht wie z.B. die Einrückung, Lesezeichenleisten und die automatische Markierung aktivieren. Das Dialogfeld "Einstellungen für Textansicht" kann auch über den

 © 2016-2022 Altova GmbH

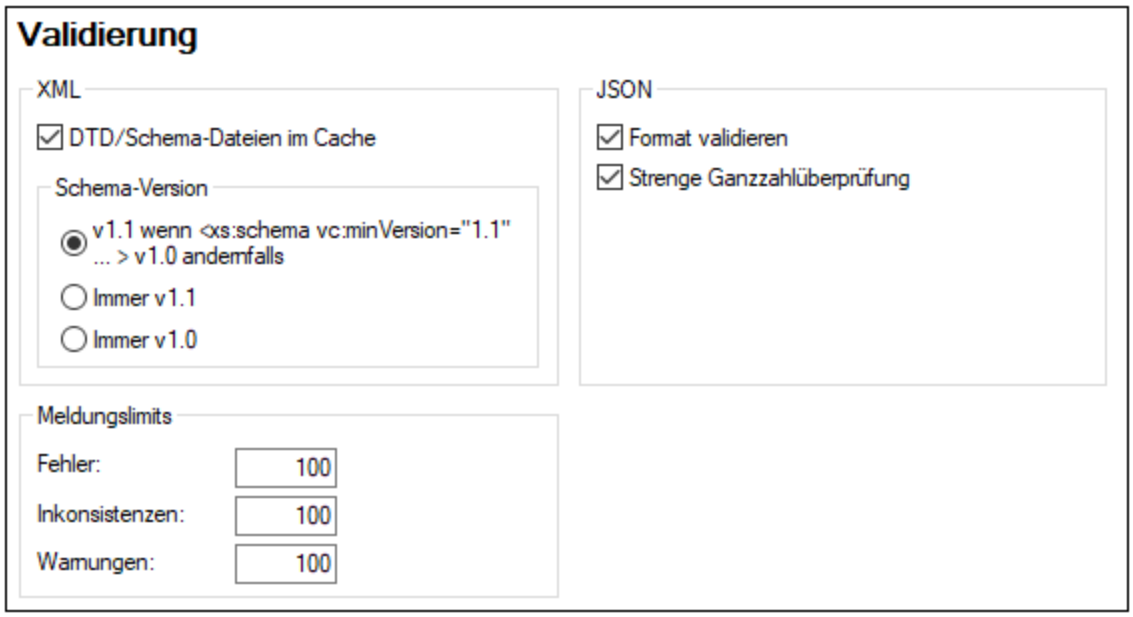
Menübefehl **Ansicht | Einstellungen für Textansicht** 1506 aufgerufen werden. Dort wird das Dialogfeld beschrieben.

Speichern und Beenden

Klicken Sie zum Abschluss auf **OK**.

28.17.15.6 Validierung

Im Abschnitt **Validierung** können Sie die Optionen für die Validierung von XML- und JSON-Dokumenten festlegen.

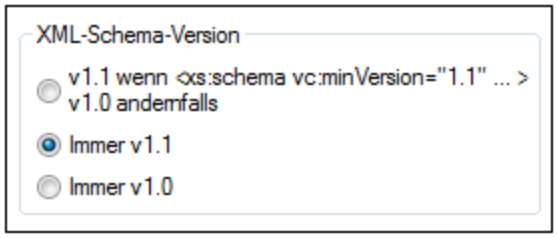


XML

XMLSpy kann DTD- und XML-Schema-Dateien im Arbeitsspeicher aufbewahren, um unnötiges Neuladen zu vermeiden (z.B. wenn ein Schema nicht lokal, sondern über eine URL aufgerufen wird). Beachten Sie jedoch, dass Änderungen bei Verwendung von zwischengespeicherten Schemas bei der Validierung nicht sofort berücksichtigt werden; Sie müssten die XML-Datei in diesem Fall neu laden oder XMLSpy neu starten.

Schema-Version

Welcher XSD-Modus in der Schema-Ansicht aktiviert wird, hängt sowohl (i) vom Vorhandensein bzw. Fehlen - und, falls vorhanden, vom Wert - des `/xs:schema/@vc:minVersion` Attributs des XSD-Dokuments als auch (ii) von der XML-Schema-Version ab, die im Dialogfeld "Optionen" (**Extras | Optionen**, *siehe Abbildung unten*) im Abschnitt "Datei" ausgewählt wurde.



Es sind die folgenden Situationen möglich. *XML-Schema-Version* in der Tabelle unten bezieht sich auf die Auswahl im oben gezeigten Bereich XML-Schema-Version. Die `vc:minVersion` Werte in der unten stehenden Tabelle beziehen sich auf den Wert des `xs:schema/@vc:minVersion` Attributs im XML-Schema-Dokument. Nähere Informationen dazu finden Sie im Abschnitt Bearbeitungsansichten | Schema-Ansicht | XSD-Modus 173.

XML-Schema-Version	`vc:minVersion` Attribut	XSD-Modus
Immer v 1.0	Fehlt oder ist mit einem beliebigen Wert vorhanden	1.0
Immer v 1.1	Fehlt oder ist mit einem beliebigen Wert vorhanden	1.1
Wert von @vc:minVersion	Das Attribut hat den Wert `1.1`	1.1
Wert von @vc:minVersion	Das Attribut fehlt oder ist vorhanden und hat einen anderen Wert als `1.1`	1.0

Meldungslimits

Mit Hilfe dieser Optionen können Sie für die Anzahl der angezeigten Fehler, XBRL-Inkonsistenzen und Warnungen separate Limits definieren. Der Standardwert für die einzelnen Kategorien ist 100. Ändern Sie ihn in den gewünschten Wert.

JSON

Die folgenden Validierungsoptionen stehen für die Validierung von JSON-Dokumenten zur Verfügung:

- *Format validieren:* Das Format von String-Typen 750 in JSON-Instanzdokumenten wird validiert.
- *Strenge Ganzzahlüberprüfung:* Es gibt zwei numerische JSON-Typen: `Zahl` und `Ganzzahl`. Mit dieser Option wird überprüft, ob Ganzzahlen vom Typ Ganzzahl (und nicht Gleitkommazahltypen (wie `7,0`), mit Vorzeichen (wie z.B, `+7`) oder Strings (wie `"7"`) sind.

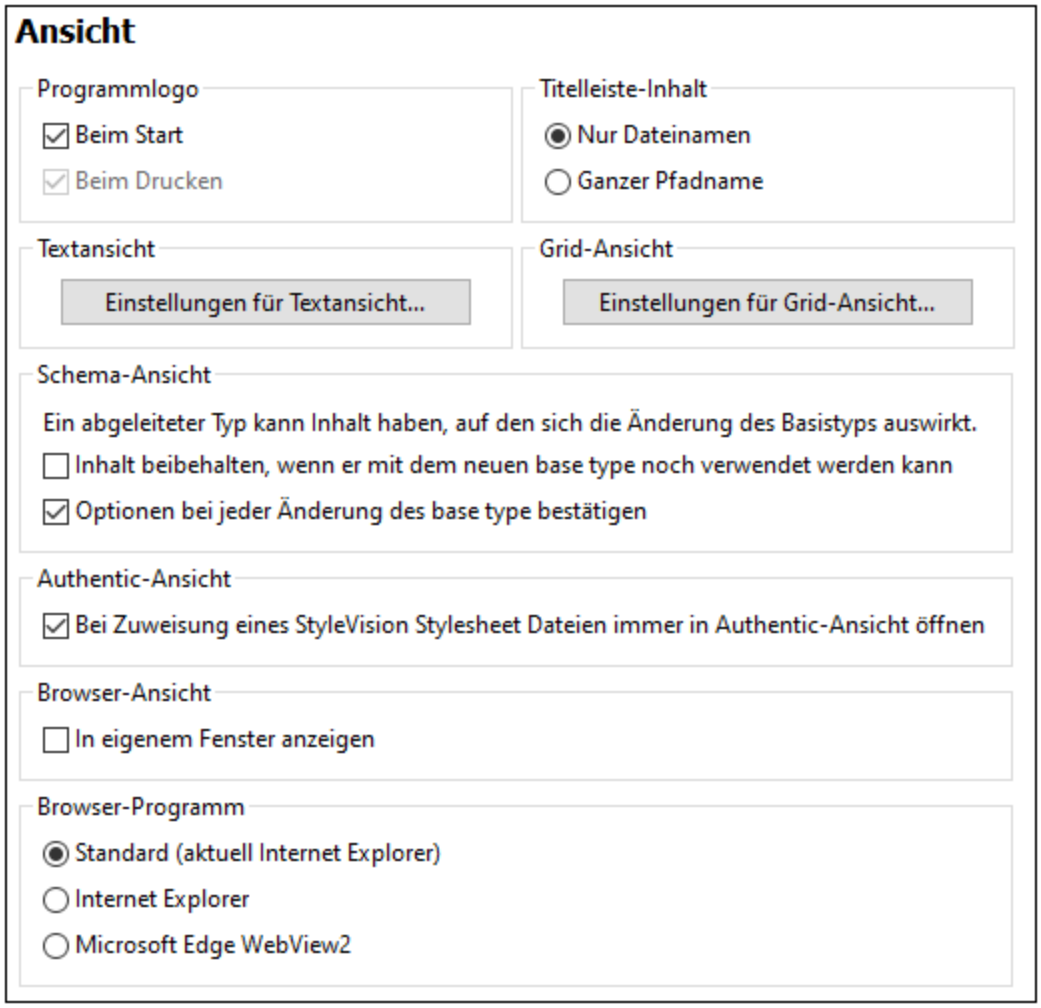
Speichern und Beenden

Klicken Sie zum Abschluss auf **OK**.

© 2016-2022 Altova GmbH

28.17.15.7 Ansicht

Im Abschnitt **Ansicht** können Sie die Darstellung der XML-Dokumente in XMLSpy anpassen.



Programmlogo

Das Programmlogo beim Starten kann deaktiviert werden, um die Applikation schneller starten zu können. Wenn Sie eine Lizenz erworben haben (und nicht mit einer Testversion arbeiten) haben Sie außerdem die Möglichkeit, das Programmlogo, die Copyright-Informationen und die Registrierungsinformationen beim Drucken eines Dokuments von XMLSpy aus zu deaktivieren.

Titelleiste-Inhalt

Im Fenstertitel jedes Dokumentfensters kann entweder nur der Dateiname oder der komplette Pfad angezeigt werden.

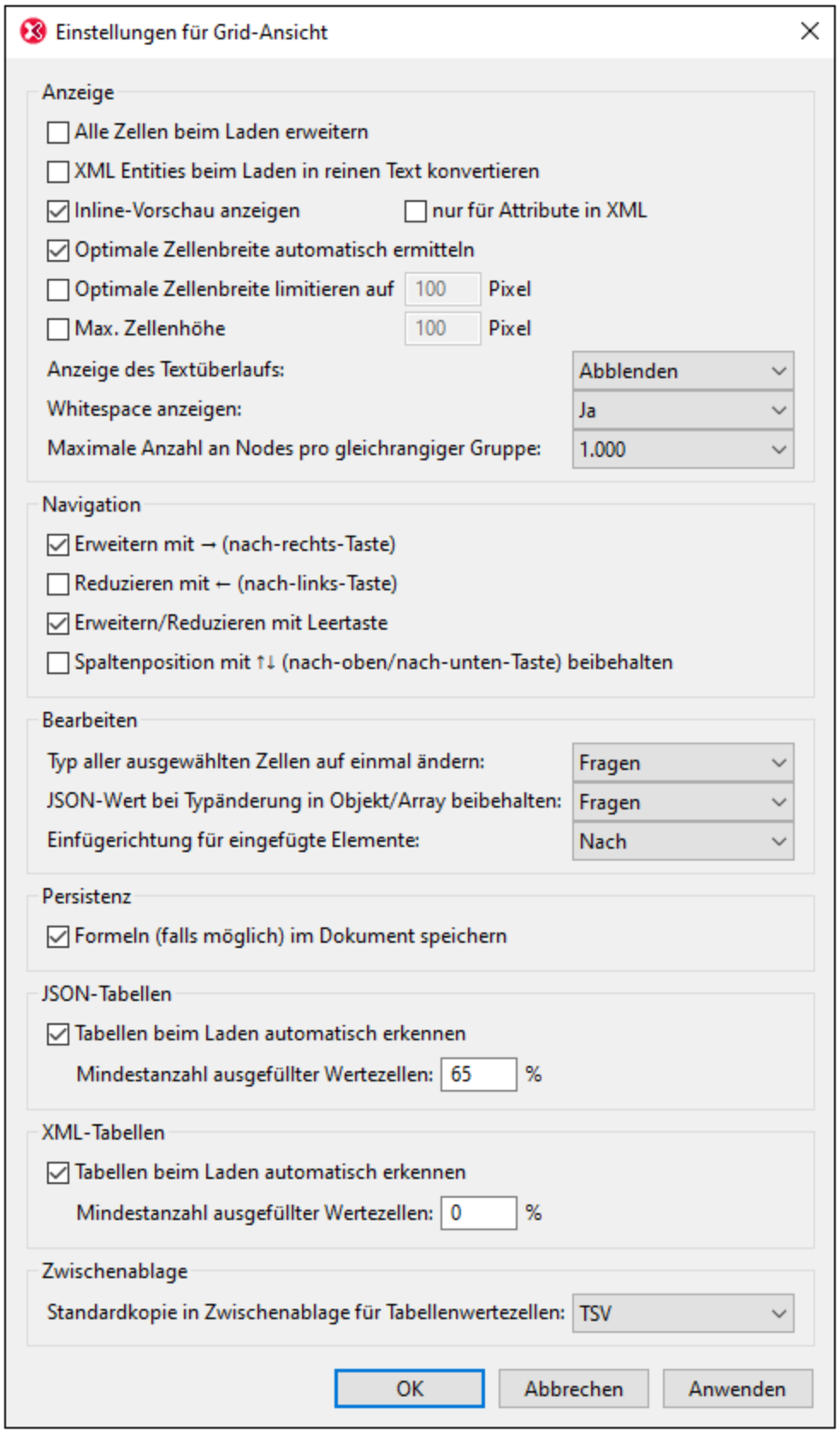
Einstellungen für Textansicht

Klicken Sie auf **Einstellungen für Textansicht**, um das Dialogfeld "Einstellungen für Textansicht" zu öffnen. Hier können Sie die Eigenschaften der Textansicht wie z.B. die Einrückung, Lesezeichenleisten und die automatische Markierung aktivieren. Das Dialogfeld "Einstellungen für Textansicht" kann auch über den Menübefehl **Ansicht | Einstellungen für Textansicht** 1506 aufgerufen werden. Dort wird das Dialogfeld beschrieben.

Einstellungen für Grid-Ansicht

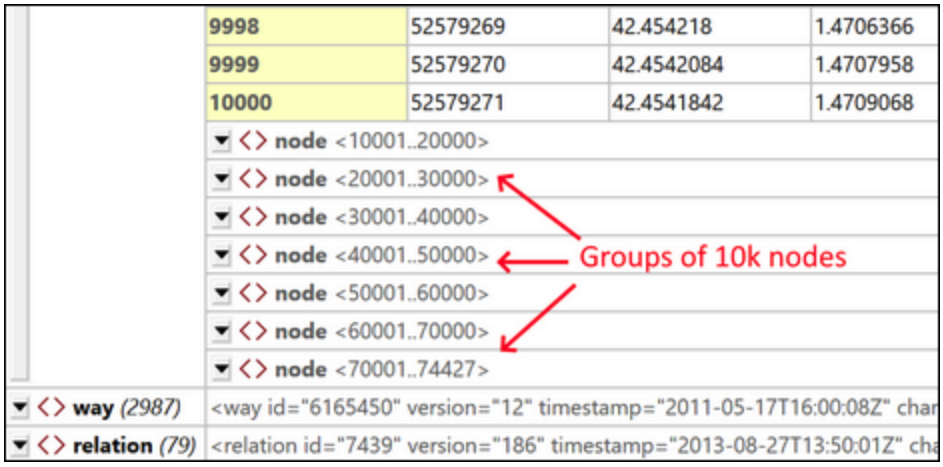
Klicken Sie auf **Einstellungen für Grid-Ansicht**, um das gleichnamige Dialogfeld zu öffnen. In der Grid-Ansicht können die folgenden Optionen konfiguriert werden:

© 2016-2022 Altova GmbH


Einstellungen für Grid-Ansicht
Anzeige
Alle Zellen beim Laden erweitern
XML Entities beim Laden in reinen Text konvertieren
Inline-Vorschau anzeigen
nur für Attribute in XML
Optimale Zellenbreite automatisch ermitteln
Optimale Zellenbreite limitieren auf 100 Pixel
Max. Zellenhöhe 100 Pixel
Anzeige des Textüberlaufs: Abblenden
Whitespace anzeigen: Ja
Maximale Anzahl an Nodes pro gleichrangiger Gruppe: 1.000
Navigation
Erweitern mit → (nach-rechts-Taste)
Reduzieren mit ← (nach-links-Taste)
Erweitern/Reduzieren mit Leertaste
Spaltenposition mit ↑↓ (nach-oben/nach-unten-Taste) beibehalten
Bearbeiten
Typ aller ausgewählten Zellen auf einmal ändern: Fragen
JSON-Wert bei Typänderung in Objekt/Array beibehalten: Fragen
Einfügerichtung für eingefügte Elemente: Nach
Persistenz
Formeln (falls möglich) im Dokument speichern
JSON-Tabellen
Tabellen beim Laden automatisch erkennen
Mindestanzahl ausgefüllter Wertezellen: 65 %
XML-Tabellen
Tabellen beim Laden automatisch erkennen
Mindestanzahl ausgefüllter Wertezellen: 0 %
Zwischenablage
Standardkopie in Zwischenablage für Tabellenwertezellen: TSV
OK
Abbrechen
Anwenden

Anzeige
Die Kontrollkästchen im Abschnitt *Anzeige* sind selbsterklärend. Im Folgenden finden Sie einige klarstellende Anmerkungen.

- Wenn beim Laden nicht alle Zelle erweitert werden, werden der Root-Node und alle seine Nachfahren reduziert angezeigt. Sie müssen dann jeden Node erweitern, wenn Sie tiefer in die Hierarchie eintauchen möchten.
- "Optimale Breite" kann automatisch aktiviert werden, (i) wenn das Dokument in der Grid-Ansicht geladen ist und (ii) wenn die Anzeige einer Gruppe von Elementen zwischen Tabellenansicht und normaler Grid-Ansicht gewechselt wird.
- Wenn die optimale Breite aktiviert ist, wird die Zelle in ihrer gesamten Breite angezeigt, sodass der Text in eine einzige Zeile passt. Um dies zu erreichen, könnte Folgendes geschehen: (i) die Tabelle könnte breiter als das Hauptfenster sein, sodass am unteren Rand des Fensters eine Bildlaufleiste angezeigt wird; (ii) Text in einigen Zellen könnte über den rechten (oder je nach Leserichtung linken) Rand der Zelle hinausreichen; dieser Textüberlauf wird durch Abblenden des Texts oder ein Auslassungszeichen (*siehe nächster Punkt*) angezeigt.
- Wenn Text nicht umbrochen wird und die Textlänge die Zellenbeite übersteigt, können Sie mit der Option *Textüberlauf* auswählen, ob der Überlauf durch Abblenden des Texts in der Nähe des Rands oder durch ein Auslassungszeichen angezeigt werden soll.
- Gleichrangige Nodes können in gleichrangigen Gruppen von 100, 1.000 oder 10.000 Nodes *(siehe Abbildung unten)* angezeigt werden. Dies ist nützlich, wenn Sie in Dateien mit sehr vielen gleichrangigen Nodes Platz sparen möchten.



Bearbeiten
Die Kontrollkästchen im Abschnitt *Ausschneiden* sind selbsterklärend. Im Folgenden finden Sie einige klarstellende Anmerkungen.

- Wenn Sie den Typ mehrerer ausgewählter Zellen ändern, stehen die folgenden Optionen dazu, wie Sie damit fortfahren möchten, zur Verfügung: *Immer, Nie* oder *Fragen* (für die Entscheidung durch den Benutzer).
- Wenn Sie einen JSON-Typ von einem atomaren Typ in ein Objekt oder Array oder umgekehrt ändern, stehen die folgenden Optionen dazu, wie Sie damit fortfahren möchten, zur Verfügung: (i) *Fragen* (ob der Wert des atomaren Typs als Wert eines unbenannten Child-`Schlüssel:Wert`-Paars beibehalten werden soll oder ob er verworfen werden soll); (ii) *Immer*: (Der Wert in einem nicht benannten Child-`Schlüssel:Wert`-Paar soll immer beibehalten werden); (iii) *Nie*: (Der Wert soll nie beibehalten werden, d.h. der Wert soll verworfen werden).
- Mit der Option *Einfügerichtung* wird festgelegt, ob eine Auswahl in der Zwischenlage oberhalb oder

 © 2016-2022 Altova GmbH

unterhalb der ausgewählten Zelle eingefügt werden soll.

Persistenz
Filter und Formeln können in den Metadaten von XML- und JSON-Dokumenten gespeichert werden. Formeln werden in XML-Dokumenten als Processing Instructions und in einigen Arten von JSON-Dokumenten als Kommentare gespeichert. Mit diesen Einstellungen wird festgelegt, wie beim Speichern von Filtern und Formeln verfahren werden soll. Im Fall von Formeln bezieht sich "*wenn möglich*" darauf, dass Formeln nur in JSON5- und JSONC-Dokumenten - und nicht in allen Dokumenten - als Kommentare gespeichert werden können.

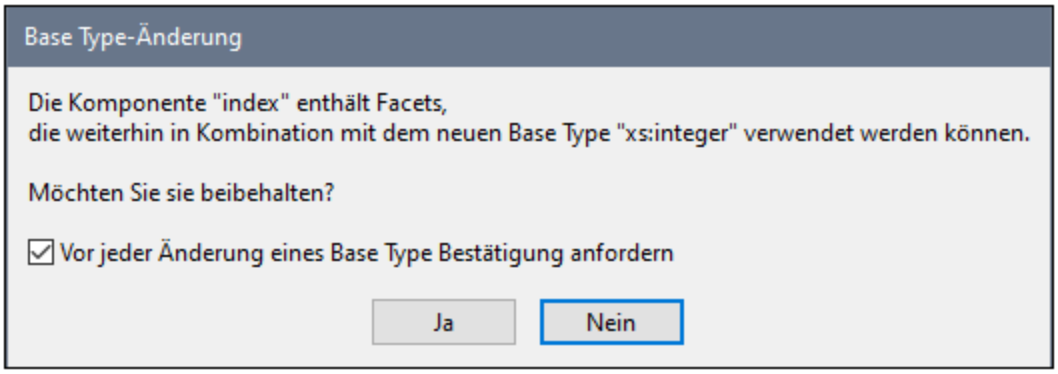
Tabellen
Es stehen die folgenden Einstellungen zur Verfügung:

- Wenn die Einstellung zum automatischen Erkennen von Tabellen beim Laden aktiviert wurde, können Sie auswählen, wie viele Tabellenzellen mindestens ausgefüllt sein müssen, damit Tabellen als Tabellen erkannt werden. Wenn die Anzahl der ausgefüllten Tabellenzellen diesen Wert nicht übersteigt, wird die Struktur als normales Raster angezeigt, wobei sich wiederholende Elemente nacheinander aufgelistet werden.
- Sie können auch auswählen, ob der Inhalt der Zwischenablage als tabellengetrennte Werte (TSV = tab-separated values) oder (je nach Dokumenttyp) als XML/JSON gespeichert werden soll. Dies ist eine sehr nützliche Funktion. Wenn Sie eine Tabelle aus der Zwischenablage in ein anderes Dokument kopieren möchten, können Sie mit dieser Einstellung auswählen, ob die kopierte Tabelle als TSV oder mit Markup-Code gespeichert werden soll. (Um den Unterschied zu sehen, fügen Sie eine Tabelle in einen Text-Editor ein, nachdem Sie diese in jedem dieser Formate in die Zwischenablage kopiert haben.)

Schema-Ansicht

Ein XML-Schema-Datentyp kann von einem anderen Datentyp abgeleitet werden. So kann z.B. ein Datentyp für E-Mail-Elemente von einem Basis-Datentyp `xs:string` abgeleitet werden (z.B. durch Einschränken des `xs:string` Datentyps auf einen bestimmten Zeichensatz). Wenn der Basis-Datentyp später geändert wird, können Sie die folgenden Optionen definieren:

- *Inhalt beibehalten:* Wenn die zum Definieren des "derived Type" verwendeten Definitionen mit dem neuen Basistyp verwendet werden können, werden die Definitionen automatisch beibehalten, wenn diese Option aktiviert ist.
- *Optionen bei jeder Änderung... bestätigen*: Nachdem Sie den Basistyp geändert haben (*siehe Abbildung unten*), wird ein Dialogfeld angezeigt, in dem Sie gefragt werden, ob die alten Definitionen beibehalten und mit dem neuen Basistyp verwendet werden sollen.



Authentic-Ansicht

XML-Dateien basierend auf einem **StyleVision Power Stylesheet** werden bei Aktivierung dieser Option automatisch in der Authentic-Ansicht geöffnet.

Browser-Ansicht

Sie können die Browser-Ansicht in einem separaten Fenster öffnen, sodass Sie die Ansicht, in der Sie arbeiten, und die Browser-Ansicht nebeneinander sehen.

Browser-Ansicht

Derzeit wird in der Authentic-und der Browser-Ansicht Internet Explorer (IE) verwendet. IE ist daher der Standard-Browser für diese beiden Ansichten. Alternativ dazu kann Microsoft Edge Web View 2 für die Browser-Ansicht verwendet werden. Wenn Edge auf Ihrem Rechner nicht installiert ist, gehen Sie zur WebView2-Download-Seite, um den Evergreen Bootstrapper zu installieren. Dadurch kann Microsoft Edge WebView 2 für die Browser-Ansicht verwendet werden.

Nähere Informationen dazu finden Sie im Abschnitt Browser-Ansicht 282.

Speichern und Beenden

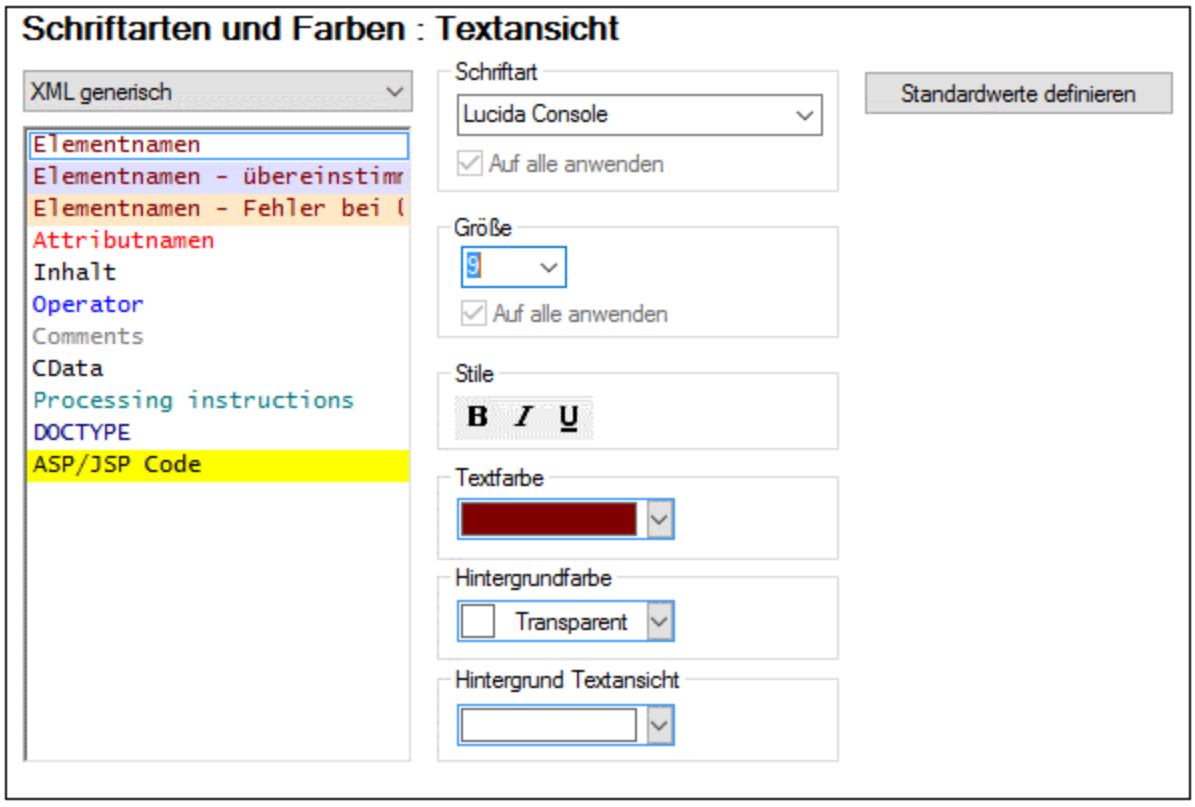
Klicken Sie zum Abschluss auf **OK**.

28.17.15.8 Schriftarten und Farben

Im Abschnitt **Schriftarten und Farben** finden Sie Optionen zum Anpassen der Darstellung von Texteinträgen in den verschiedenen Ansichten von XMLSpy.

Anmerkung: Die Optionen zu Schriftarten und Farben werden auf das aktuell aktive Design angewendet. Um Schriftarten und Farben in einem anderen Design zu ändern, machen Sie dieses zum aktiven Design, bevor Sie diese Optionen ändern.

© 2016-2022 Altova GmbH



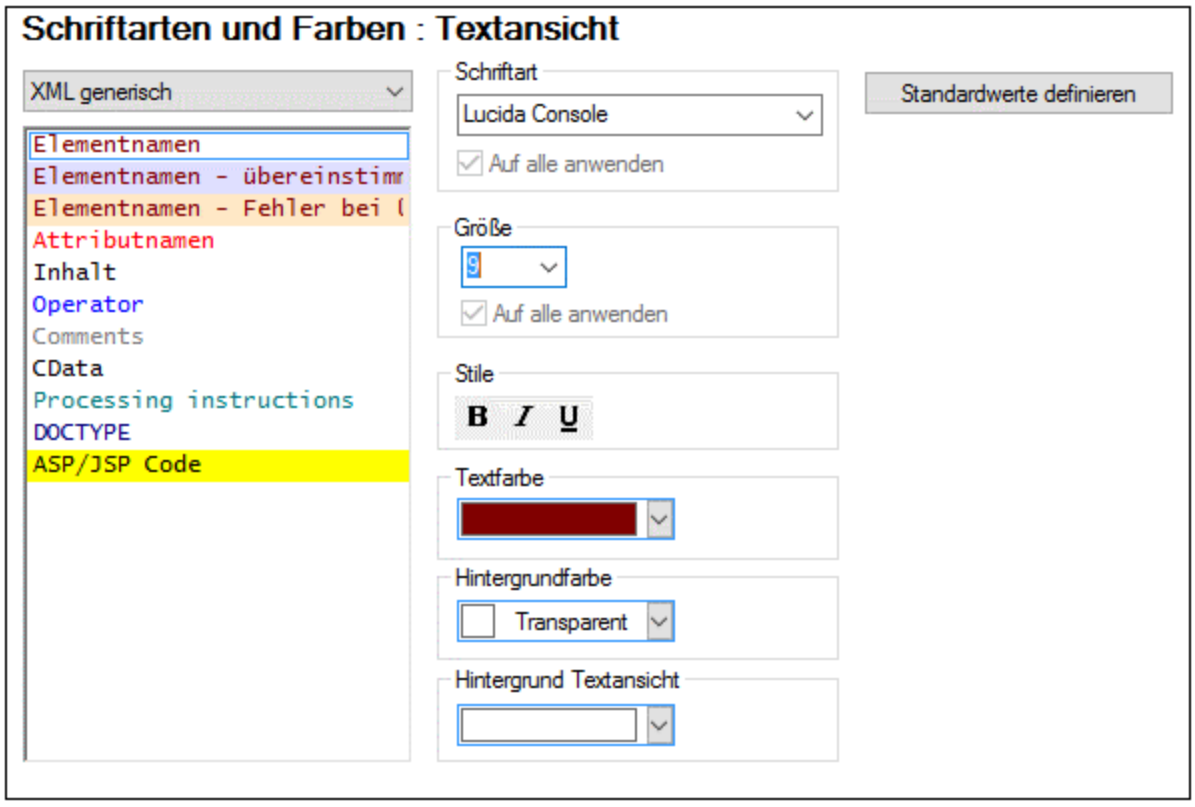
Es stehen Optionen für die folgenden Ansichten zur Verfügung:

- Textansicht 1623
- Grid-Ansicht 1626
- Schema-Design-Ansicht 1628
- WSDL-Design-Ansicht 1631
- XBRL-Taxonomieansicht 1632

Wählen Sie im linken Bereich des Dialogfelds die Ansicht, die Sie anpassen möchten, aus. Die Texteinträge, die formatiert werden können, werden im rechten Bereich angezeigt (*Abbildung oben*). Wählen Sie den Texttyp aus, den Sie formatieren möchten und weisen Sie ihm die gewünschten Formatierungseigenschaftswerte zu.

28.17.15.8.1 Textansicht

Im Abschnitt **Textansicht** können Sie das Aussehen von Texttypen in verschiedenen Arten von Dokumenten für das aktuell aktive Design anpassen (*siehe Abbildung unten und Liste der Dokumente im Abschnitt Dokumenttypen*). So können Sie z.B. eine unterschiedliche Formatierung für die Anzeige von Element- und Attributnamen in XML-Dokumenten oder für die Anzeige von Schlüsselwörtern und Variablen in XQuery-Dokumenten definieren.



Anmerkung: Die Formatierung wird für das gerade aktive Design[1650] definiert. Um die Formatierung eines andere Designs zu konfigurieren, machen Sie dieses zum aktiven Design.

Dokumenttypen

Sie können in der Auswahlliste links oben im Dialogfeld einen bestimmten Dokumenttyp auswählen und anschließend die Formatierung der Textelemente dieses Dokumenttyps definieren.

Es können die Textelemente der folgenden Dokumenttypen formatiert werden:

- XML generisch
- XQuery
- CSS
- JSON
- C-Familie
- Python
- XULE
- Verschiedenes
- Ausgabe

So nehmen Sie Anpassungen vor:

Um einzelne Textelementtypen eines bestimmten Dokumenttyps anzupassen, gehen Sie folgendermaßen vor:

© 2016-2022 Altova GmbH

1. Wählen Sie in der Auswahlliste links oben die Art des Dokuments aus, für das Sie Text anpassen möchten. Dabei werden die Textelementtypen für diesen Dokumenttyp im Kästchen unterhalb der Auswahlliste angezeigt. (*In der Abbildung oben wurde als Dokumenttyp* `XML generisch` *ausgewählt.*)
2. Wählen Sie den gewünschten Textelementtyp durch Anklicken aus. (*In der Abbildung oben wurde* `Elementnamen` *ausgewählt.)*
3. Definieren Sie mittels der Optionen im Bereich auf der rechten Seite die Schriftarteigenschaften des ausgewählten Textelementtyps. Über die Option *Hintergrund Textansicht* können Sie eine Hintergrundfarbe für die gesamte Textansicht auswählen.

Beachten Sie die folgenden Punkte:

- Für alle **Textelementtypen** eines bestimmten Dokumenttyps (z.B. eines generischen XML-Dokumenttyps) werden dieselbe Schriftart, dieselbe Größe und derselbe Stil verwendet. In einem Dokumenttyp können nur Textfarbe und Hintergrundfarbe einzelner Texttypen geändert werden. Dies aktiviert die Funktion "Syntaxfärbung".
- In der Kategorie *XML generisch* besteht der Texttyp *Elementnamen* aus drei Subtypen: (i) *Elementnamen* wird auf nicht ausgewählte Elementnamen angewendet; (ii) *Elementnamen - übereinstimmende* wird auf Elementnamen angewendet, die ausgewählt sind (Namen in denen sich der Cursor befindet) und bei denen der Name des Start-Tag mit dem des End-Tag übereinstimmt.; (i) *Elementnamen - Fehler bei Übereinstimmung* wird auf diejenigen Elementnamen angewendet, die ausgewählt sind, bei denen aber der Name des Start-Tag nicht mit dem des End-Tag übereinstimmt. Elementnamen, die gerade bearbeitet werden, werden daher je nachdem, ob der Name des Start-Tag mit dem des End-Tag übereinstimmt oder nicht, in einer unterschiedlichen Hintergrundfarbe markiert. Diese Markierungsfarben können durch Änderung der entsprechenden Hintergrundfarbe geändert werden. Die Markierung ist standardmäßig aktiviert und kann durch Deaktivieren der Option *Elemente markieren* im Dialogfeld Einstellungen für Textansicht[1506] deaktiviert werden.
- Die Einstellungen in der Kategorie "XULE" werden (i) auf das XULE-Dokument im Hauptfenster und (ii) die im Fenster "XULE"[941] eingegebenen XULE-Regeln angewendet.
- In der Kategorie *Verschiedenes*: (i) *Auswahl* bezieht sich auf den aktuell ausgewählten Textinhalt; *Auswahl -inaktiv* bezieht sich auf andere Instanzen desselben Textinhalts im Text; (ii) *Aktiven Marker suchen* bezieht sich auf die aktuell ausgewählte Instanz eines Suchergebnisses, während sich *Marker suchen* auf andere (nicht aktive) Instanzen des Suchergebnisses bezieht; (iii) *Debug/Aufruf-Marker* bezieht sich auf den aktuell ausgewählten Schritt in einer Debug-Sitzung; (iv) *Sichtbare Leerzeichen* bezieht sich auf die Whitespace-Markierungen in einem Dokument. Die Whitespace-Markierungen in einem Dokument können durch Einschalten der Whitespace-Markierungen[1506] sichtbar gemacht werden. (v) *Rückwärts-Mapping aktiv* bezieht sich auf aktuell ausgewählten Inhalt in den drei mit dem Rückwärts-Mapping[1418] in Zusammenhang stehenden Dokumenten (**`XML–XSLT/XQuery–Ergebnis`**); *Rückwärts-Mapping inaktiv* bezieht sich auf rückgemappten Inhalt in den beiden anderen Dokumenten.

Standardwerte definieren

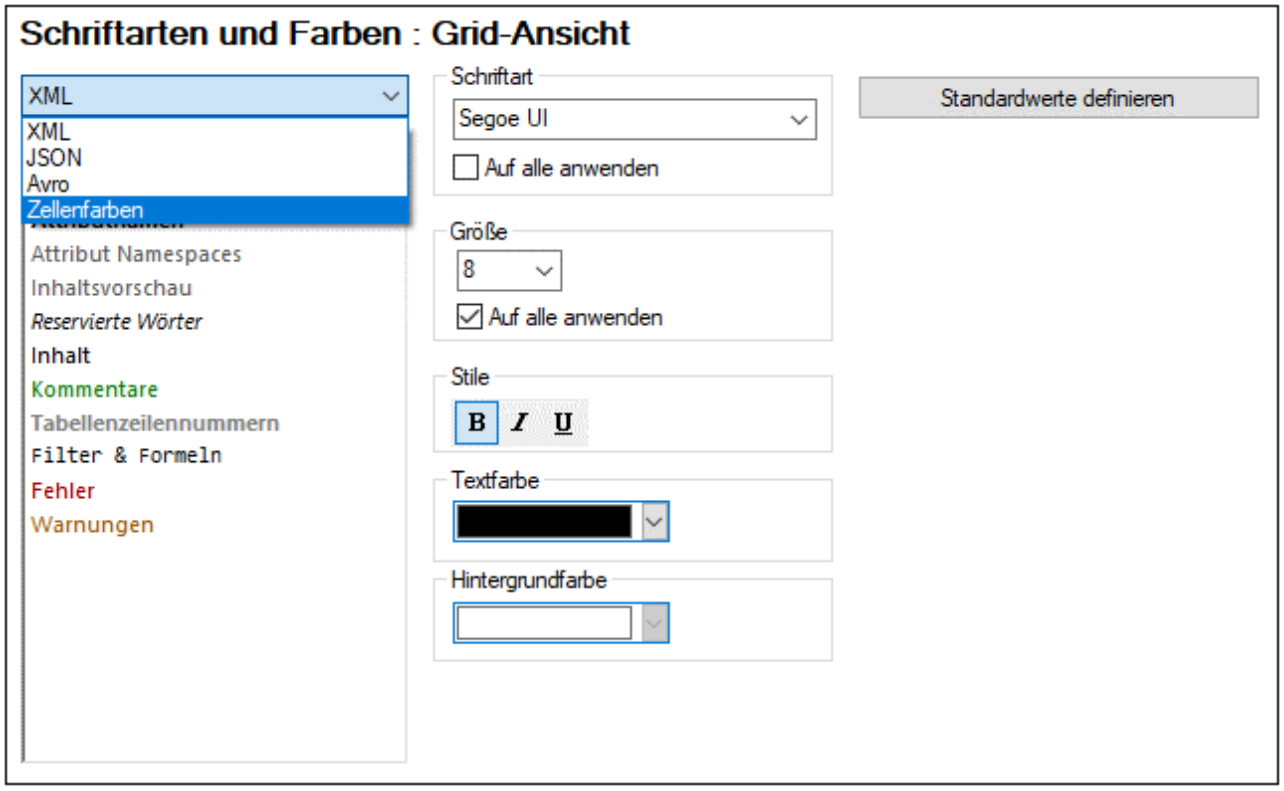
Über die Schaltfäche **Standardwerte definieren** können Sie Schriftarten auf die ursprünglichen Einstellungen zum Zeitpunkt der Installation zurücksetzen.

Speichern und Beenden

Klicken Sie zum Abschluss auf **OK**.

28.17.15.8.2 Grid-Ansicht

Im Abschnitt **Grid-Ansicht** (*Abbildung unten*) können Sie das Aussehen von Text in der XML-Grid-Ansicht von XML-Dokumenten[169], der JSON-Grid-Ansicht[690] und der Avro-Grid-Ansicht[786] anpassen. Über die Option *Zellenfarben (siehe Abbildung)* können Sie die Farbe von Zellen bei Anzeige von Grid-Komponenten als Tabelle einstellen. Wählen Sie in der Auswahlliste die Dokumentanzeige aus, für die Sie die Grid-Ansicht konfigurieren möchten. Wählen Sie anschließend den gewünschten Textelementtyp aus und weisen Sie diesem die gewünschten (*unten aufgelisteten*) Formatierungseigenschaften zu.



Anmerkung: Die Formatierung wird für das gerade aktive Design[1650] definiert. Um die Formatierung eines andere Designs zu konfigurieren, machen Sie dieses zum aktiven Design.

XML-Grid-Ansicht, JSON-Grid-Ansicht, Avro-Grid-Ansicht

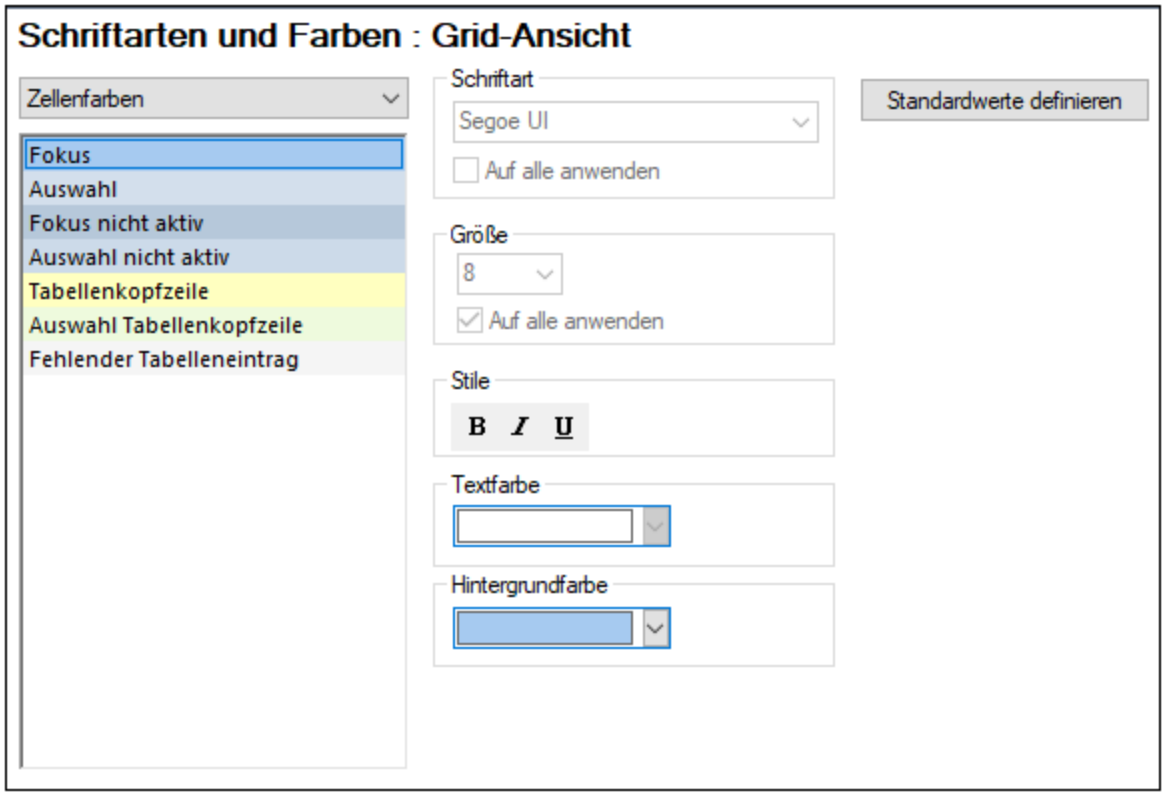
Sie können die folgenden Eigenschaften für den gewünschten Textelementtyp auswählen:

- *Schriftart und Größe:* Die ausgewählte Schriftart wird auch in Ausdrucken der Grid-Ansicht verwendet. Wenn Sie für alle Textelementtypen dieselbe Schriftart und -größe verwenden möchten, aktivieren Sie das entsprechende Kontrollkästchen *auf alle anwenden*.
- *Textstil, Textfarbe, Hintergrundfarbe:* Definiert den Stil, die Farbe und den Hintergrund einzelner Textelementtypen. Die aktuellen Einstellungen werden sofort in der Liste auf der linke Seite angezeigt, sodass Sie eine Vorschau auf das Aussehen des Textelements sehen.

© 2016-2022 Altova GmbH

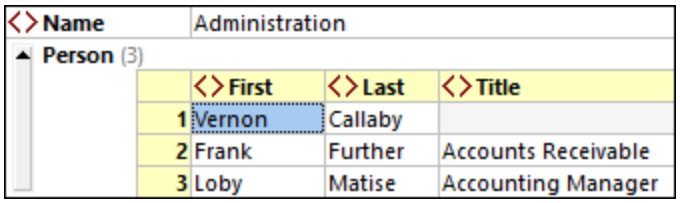
Zellenfarben (der Tabellenansicht in allen Grid-Ansichten)

Die Anzeige von Tabellen in allen Grid-Ansichten (XML, JSON, Avro) kann nicht nur auf Basis ihrer sprachspezifischen Semantik konfiguriert werden (so kann etwa ein XML-Element anders als ein XML-Kommentar formatiert werden), sondern Sie können die Grid-Ansicht auch auf Basis der Zellenfunktion formatieren, z.B. um zwischen dem selektierten und unselektiertem Status verschiedener Tabellenkomponenten zu unterscheiden. Am besten, Sie weisen den einzelnen Elementen dazu unterschiedliche Hintergrundfarben zu (*siehe Abbildung unten*), Sie können jedoch auch andere/zusätzliche Formatierungseigenschaften für unterschiedliche Zelltypen verwenden.

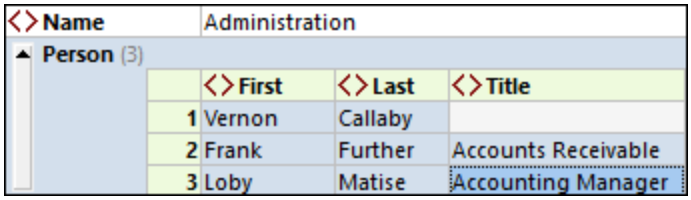


Es können Einstellungen für die folgenden Komponente definiert werden:

- *Tabellenkopfzeile (nicht ausgewählt)* und *Auswahl Tabellenkopfzeile:* Diese Einstellungen beziehen sich auf Spalten- und Zeilenüberschriften. In der Abbildung unten sehen sie nicht ausgewählte Kopfzeilen. Ihre Hintergrundfarbe wird wie im Dialogfeld oben definiert angezeigt. Die Farbe für *ausgewählte Tabellenkopfzeilen* wird aktiviert, wenn alle Überschriften ausgewählt sind und nicht, wenn nur eine einzelne Überschrift ausgewählt ist. Alle Überschriften können entweder durch Klick auf die Zelle am Kreuzungspunkt zwischen Spalten- und Zeilenüberschriften oder durch Auswahl des Elements, das als Tabelle erstellt wurde - oder jedes seiner übergeordneten Elemente - ausgewählt werden.



- *Fehlender Tabelleneintrag:* Bezieht sich auf fehlende Elemente oder Attribute im Dokument (*siehe Abbildung unten*).
- *Auswahl* und *Fokus*: Bezieht sich auf die Zellen, die ausgewählt sind und diejenigen, die den Fokus haben. So ist z.B. in der Abbildung unten die gesamte Tabelle `Person` ausgewählt. die Zelle rechts unten hat den Fokus.



- *Nicht aktive Auswahl, nicht aktiver Fokus:* Wenn ein Zellenblock ausgewählt wurde und eine oder mehrere Zellen in diesem Block den Fokus haben, sind sowohl die Auswahl als auch der Fokus aktiv, solange die Grid-Ansicht aktiv ist. Wenn allerdings ein anderes Dialogfeld oder Fenster aktiv gemacht wird, ohne dass die Auswahl oder der Fokus in der Grid-Ansicht aufgehoben wurde, so sind die Auswahl und der Fokus in der Grid-Ansicht nicht aktiv. Wenn die Grid-Ansicht wieder zur aktiven gemacht wird, werden die Auswahl und der Fokus wieder aktiv.

Anmerkung: Zusätzlich zu den hier definierten Farben werden in XMLSpy die regulären Auswahl- und Menüfarbeinstellungen verwendet, die in den Anzeigeeinstellungen der Systemsteuerung Ihres Windows-Systems gewählt wurden.

Standardwerte definieren

Über die Schaltfäche **Standardwerte definieren** können Sie Schriftarten auf die ursprünglichen Einstellungen zum Zeitpunkt der Installation zurücksetzen.

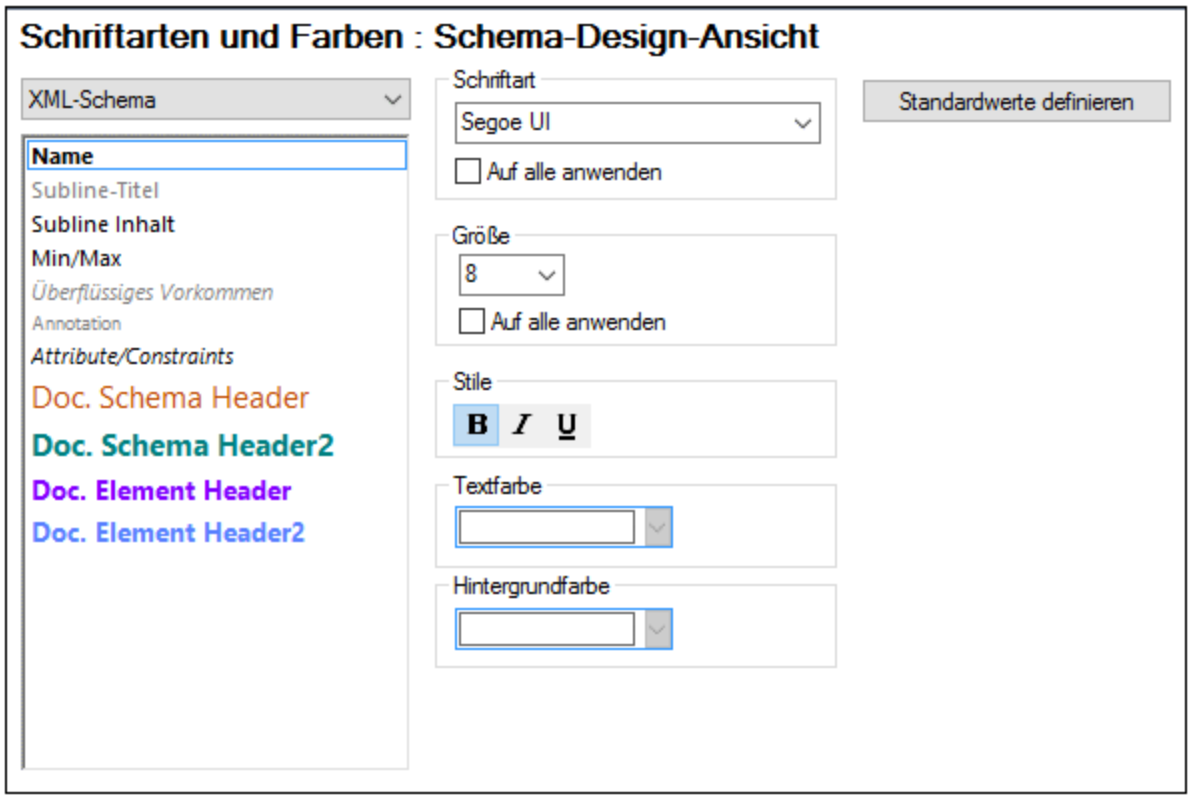
Speichern und Beenden

Klicken Sie zum Abschluss auf **OK**.

28.17.15.8.3 Schema-Design-Ansicht

Im Abschnitt **Schema-Design-Ansicht** können Sie das Aussehen von Texten in der Schema-Ansicht[171] von XML-Schema-[451] und JSON-Schema-[682]Dokumenten anpassen. Die Formatierung wird für das gerade aktive Design[1650] definiert. Um die Formatierung eines andere Designs zu konfigurieren, machen Sie dieses zum aktiven Design.

© 2016-2022 Altova GmbH

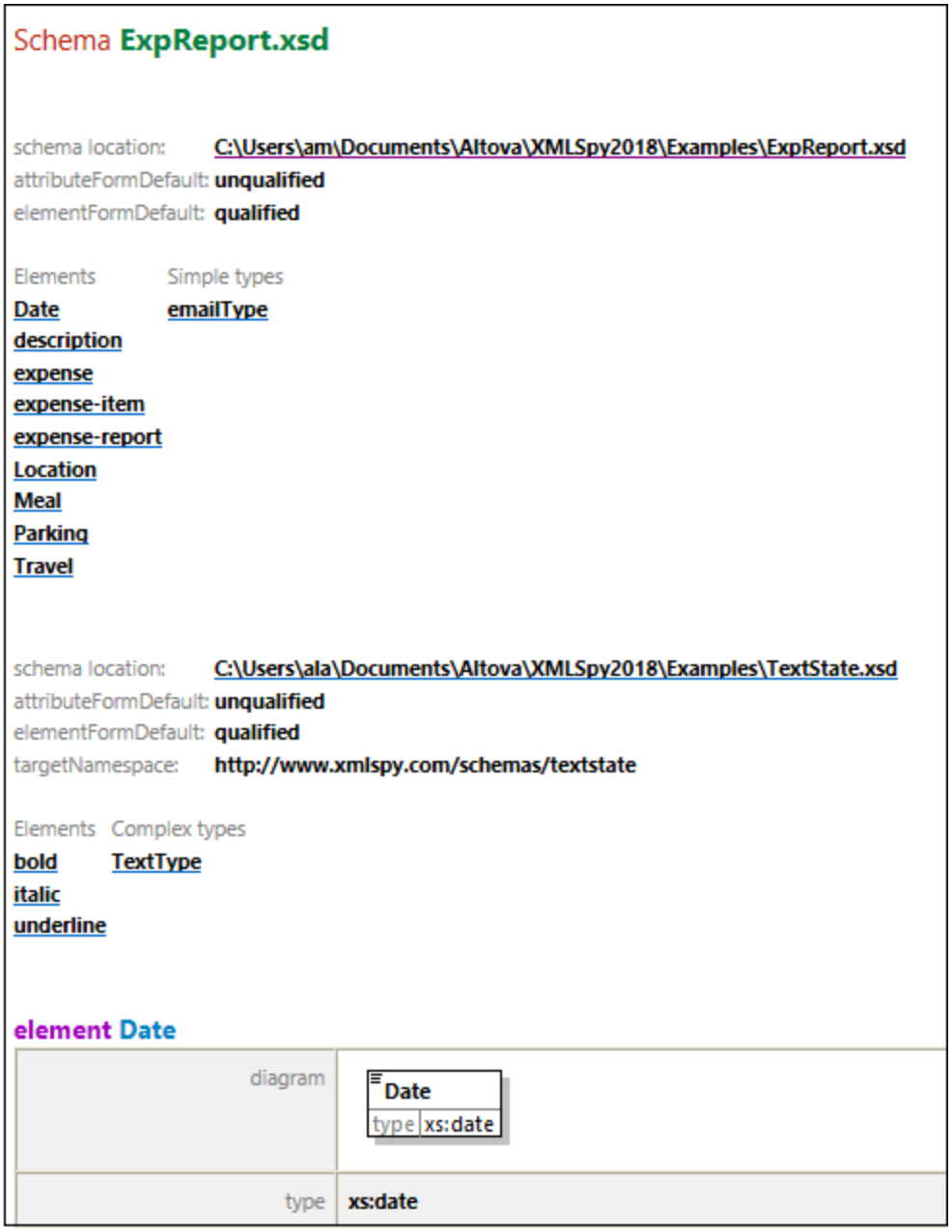


So nehmen Sie Anpassungen vor:

Um einzelne Textelementtypen eines bestimmten Dokumenttyps anzupassen, gehen Sie folgendermaßen vor:

1. Wählen Sie in der Auswahlliste links oben "XML-Schema" oder "JSON-Schema" aus. Die Textelementtypen für den ausgewählten Dokumenttyp werden im Kasten unterhalb der Auswahlliste angezeigt.
2. Wählen Sie den gewünschten Textelementtyp durch Anklicken aus.
3. Definieren Sie mittels der Optionen im Bereich auf der rechten Seite die Schriftarteigenschaften des ausgewählten Textelementtyps.

Anmerkung:
Die Textelementtypen *Doc.Schema Header(2)* und *Doc.Element Header(2)* beziehen sich auf die Schemaüberschrift bzw. auf die Elementüberschrift in der zum Schema generierten Dokumentation[1389]. Vergleichen Sie in der Abbildung unten die Farben dieser Eigenschaften im Dialogfeld oben mit den Farben der Schemaüberschrift und der Elementüberschrift der generierten Dokumentation.



Standardwerte definieren

Über die Schaltfäche **Standardwerte definieren** können Sie Schriftarten auf die ursprünglichen Einstellungen zum Zeitpunkt der Installation zurücksetzen.

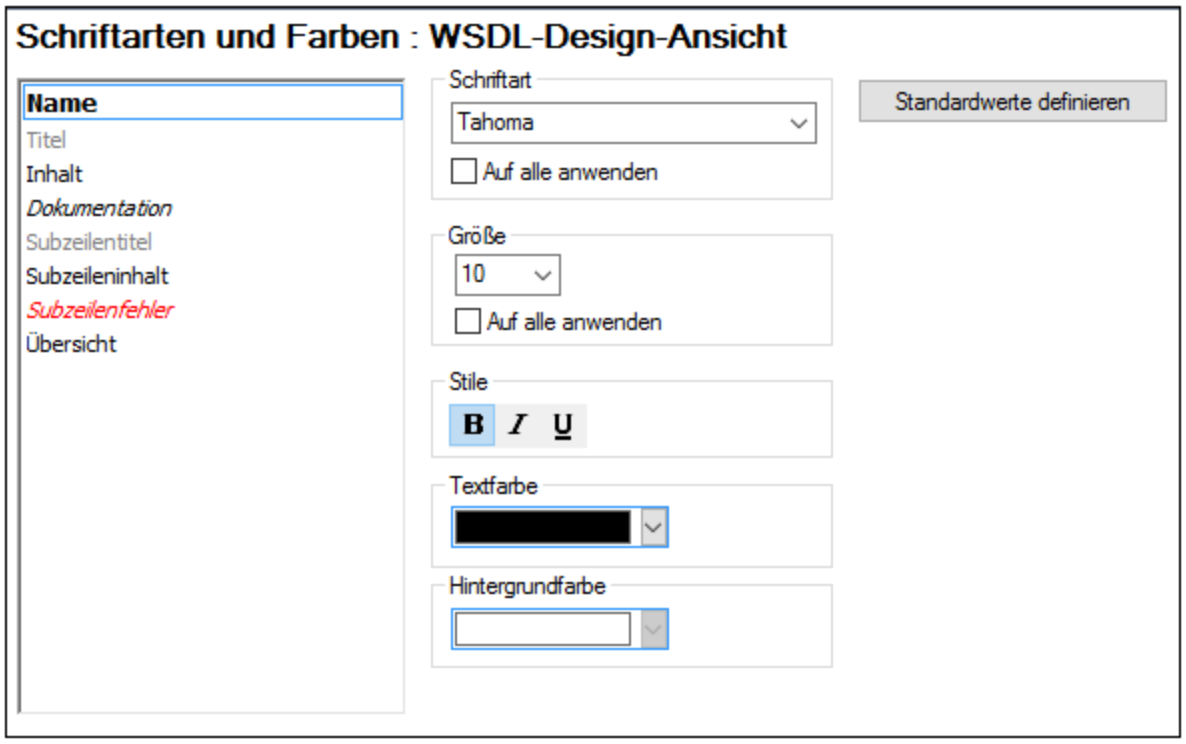
Speichern und Beenden

Klicken Sie zum Abschluss auf **OK**.

© 2016-2022 Altova GmbH

28.17.15.8.4 WSDL-Design-Ansicht

Im Abschnitt **WSDL-Design-Ansicht** können Sie die Optionen für das Aussehen von Textelementen in der WSDL-Ansicht[253] anpassen (*siehe Abbildung unten*). Die Formatierung wird für das gerade aktive Design[1650] definiert. Um die Formatierung eines andere Designs zu konfigurieren, machen Sie dieses zum aktiven Design.



Sie können die folgenden Eigenschaften für den gewünschten Textelementtyp auswählen:

- *Schriftart und Größe:* Die ausgewählte Schriftart wird auch in Ausdrucken der WSDL-Design-Ansicht verwendet. Wenn Sie für alle Textelementtypen dieselbe Schriftart und -größe verwenden möchten, aktivieren Sie das entsprechende Kontrollkästchen *auf alle anwenden*.
- *Textstil, Textfarbe, Hintergrundfarbe:* Definiert den Stil, die Farbe und den Hintergrund einzelner Textelementtypen. Die aktuellen Einstellungen werden sofort in der Liste auf der linke Seite angezeigt, sodass Sie eine Vorschau auf das Aussehen des Textelements sehen.

Anmerkung:
Die Formatierung der WSDL-Überschrift und der Service-Namen in der zum WSDL-Dokument generierten Dokumentation[1517] stammt aus den Formatierungsdefinitionen der Textelementtypen *Doc.Schema Header(2)* bzw. *Doc.Element Header(2)* der Schema-Design-Ansicht[1628].

Standardwerte definieren

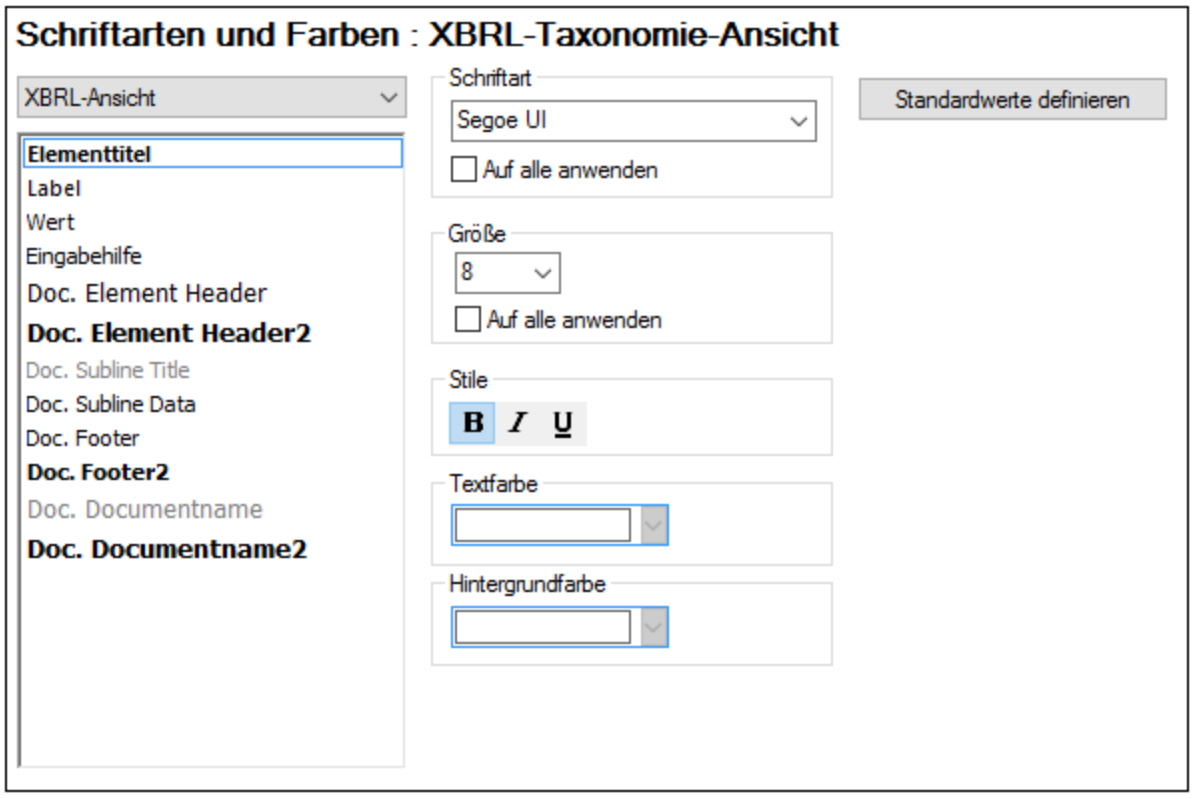
Über die Schaltfäche **Standardwerte definieren** können Sie Schriftarten auf die ursprünglichen Einstellungen zum Zeitpunkt der Installation zurücksetzen.

Speichern und Beenden

Klicken Sie zum Abschluss auf **OK**.

28.17.15.8.5 XBRL-Taxonomie-Ansicht

Im Abschnitt **XBRL-Taxonomie-Ansicht** können Sie die Optionen für das Aussehen von Textelementen in der XBRL-Ansicht[266] definieren (*siehe Abbildung unten*). Die Formatierung wird für das gerade aktive Design[1650] definiert. Um die Formatierung eines andere Designs zu konfigurieren, machen Sie dieses zum aktiven Design.



Sie können die folgenden Eigenschaften für den gewünschten Textelementtyp auswählen:

- *Schriftart und Größe:* Die ausgewählte Schriftart wird auch in Ausdrucken der XBRL-Taxonomie-Ansicht verwendet. Wenn Sie für alle Textelementtypen dieselbe Schriftart und -größe verwenden möchten, aktivieren Sie das entsprechende Kontrollkästchen *auf alle anwenden*.
- *Textstil, Textfarbe, Hintergrundfarbe:* Definiert den Stil, die Farbe und den Hintergrund einzelner Textelementtypen. Die aktuellen Einstellungen werden sofort in der Liste auf der linke Seite angezeigt, sodass Sie eine Vorschau auf das Aussehen des Textelements sehen.
- Mit Hilfe der XBRL-Tabellenvorschauoptionen können Einstellungen für Komponenten von XBRL-Tabellen, die auf dem Register "*Tabellen*" der XBRL-Ansicht als Tabellen angezeigt werden können, vorgenommen werden.

Standardwerte definieren

Über die Schaltfläche **Standardwerte definieren** können Sie Schriftarten auf die ursprünglichen Einstellungen zum Zeitpunkt der Installation zurücksetzen.

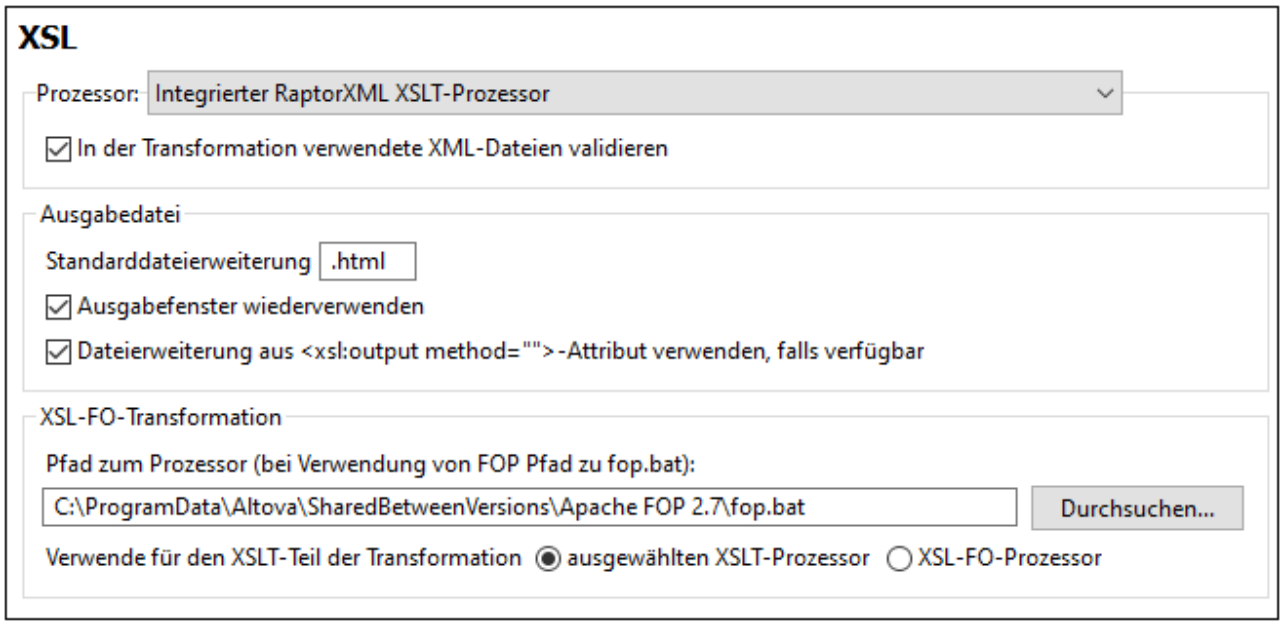
© 2016-2022 Altova GmbH

Speichern und Beenden

Klicken Sie zum Abschluss auf **OK**.

28.17.15.9 XSL

Im Abschnitt **XSL** (*Abbildung unten*) werden die Optionen für XSLT-Transformationen und über die Applikation ausgeführte XSL-FO-Transformationen definiert. Sie können für Transformationen in XMLSpy einen XSLT-Prozessor Ihrer Wahl auswählen. Die entsprechenden Optionen für die einzelnen Prozessoren werden im Abschnitt *Prozessor* angezeigt, wenn Sie einen Prozessor auswählen. Die Einstellungen *Ausgabedatei* und *XSL-FO-Transformation* sind für alle Prozessoren die gleichen. In der Abbildung unten sehen Sie die Einstellungen bei Verwendung des integrierten Altova RaptorXML-XSLT-Prozessors.



Prozessoreinstellungen

Sie können einen XSLT-Prozessor definieren, der bei Aufruf des Befehls **XSLT-Transformation** [1411] XSLT-Transformationen durchführt.

Zur Auswahl stehen die folgenden XSLT-Prozessor-Optionen:

- Integrierter RaptorXML XSLT-Prozessor
- Microsoft XML Parser (MSXML)
- Externes XSL-Transformationsprogramm

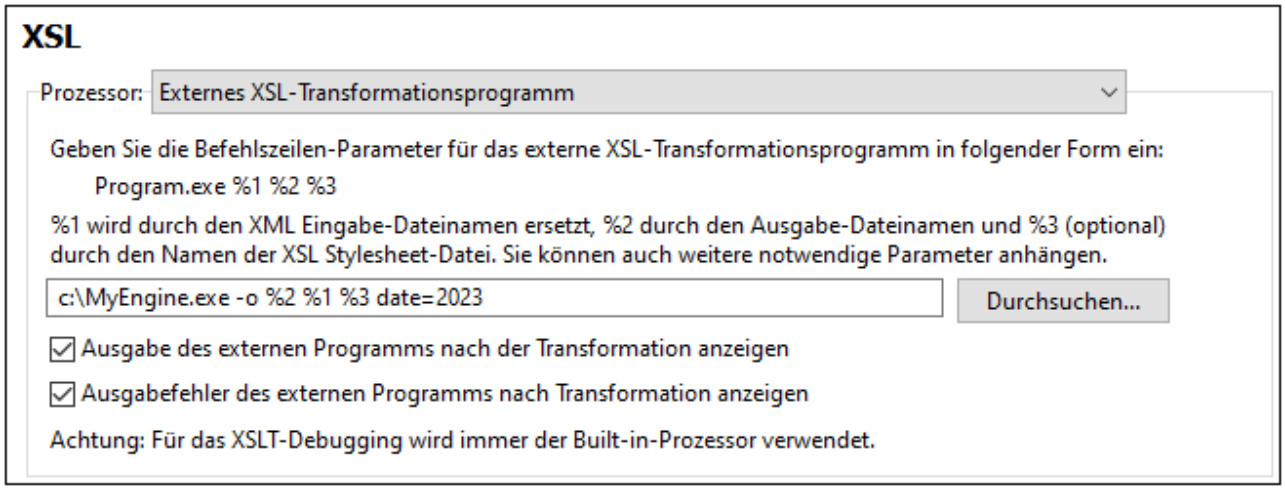
Anmerkung: Für das XSLT-Debugging wird in XMLSpy immer der integrierte RaptorXML XSLT-Prozessor verwendet, selbst wenn für die Transformation hier ein anderer XSLT-Prozessor ausgewählt wurde.

Altova RaptorXML XSLT-Prozessor
XMLSpy enthält den Altova RaptorXML XSLT 1.0-, XSLT 2.0- und XSLT 3.0-Prozessor, den Sie für Ihre XSLT-Transformationen verwenden können. Je nach dem Wert des `version`-Attributs im Element `xsl:stylesheet` oder `xsl:transform` wird immer der jeweilige XSLT-Prozessor für Transformationen (1.0, 2.0 oder 3.0) verwendet. Dies gilt sowohl für XSLT-Transformationen als auch für das XSLT Debugging mit dem XSLT/XQuery-Debugger von XMLSpy.

Wenn Sie die in der Transformation verwendeten XML-Dateien validieren möchten, wählen Sie die Option *Validieren* (*siehe Abbildung oben*) aus.

Microsoft XML Parser (MSXML)
Auf Ihrem Rechner sind ein oder mehrere MSXML 3.0-, 4.0- oder 6.0-Parser vorinstalliert. Wenn Sie wissen, welche installierte Version Sie verwenden möchten, können Sie diese auswählen. Andernfalls wird die Version automatisch von XMLSpy ausgewählt. (Die Option *Version automatisch auswählen* ist standardmäßig aktiv.) In diesem Fall versucht XMLSpy, die neueste verfügbare Version auszuwählen.

Externes XSL-Transformationsprogramm
Wählen Sie einen externen XSLT-Prozessor Ihrer Wahl aus, indem Sie den Pfad zur ausführbaren Datei eingeben.



Sie müssen den Befehlszeilenstring, den der externe XSLT-Prozessor zur Ausführung einer Transformation verwendet, definieren. Sie können den Befehlszeilenstring mit den folgenden Komponenten erstellen:

`%1` = zu verarbeitendes XML-Dokument
`%2` = zu generierende Ausgabedatei
`%3` = zu verwendendes XSLT Stylesheet (wenn das XML-Dokument keine Referenz auf ein Stylesheet enthält)

Angenommen, Sie haben einen Prozessor, der zur Ausführung einer XSLT-Transformation den folgenden Befehl verwendet:

```
myxsltengine.exe -o <output.xml> <input.xml> <stylesheet.xslt> <param-name>=<param-value>?
```

 © 2016-2022 Altova GmbH

Dann können Sie in XMLSpy die entsprechenden Variablen an der entsprechenden Stelle zur Generierung dieses Befehls verwenden. Beispiel:

```
c:\MyEngine.exe -o %2 %1 %3 date=2023
```

XMLSpy sendet die korrekten Input-Dateien zur Verarbeitung an den externen Prozessor und retourniert die Ausgabedatei(en) an einen Ausgabepfad, falls einer definiert wurde, und/oder an ein Applikationsfenster.

Aktivieren Sie die entsprechenden Kontrollkästchen, um die Ausgabe- und Fehlermeldungen des externen Programms im Fenster "Meldungen" in XMLSpy anzuzeigen.

Anmerkung: Die Parameter, die im XMLSpy-Dialogfeld XSLT-Eingabeparameter[1414] definiert sind, werden nur an die internen Altova-XSLT-Prozessoren übergeben und nicht an andere XSLT-Prozessoren, die als XSLT-Standardprozessor konfiguriert wurden.

Ausgabedateieinstellungen

Es stehen die folgenden Optionen zur Verfügung:

- *Standarddateierweiterung:* Definiert eine Standarddateierweiterung für Ausgabedateien, die durch die im XSLT-Element **`xsl:output`** definierte Dateierweiterung (*siehe letzter Eintrag in der Liste*) außer Kraft gesetzt werden kann.
- *Ausgabefenster wiederverwenden:* Dadurch wird das Ergebnisdokument bei nachfolgenden Transformationen im selben Ausgabefenster angezeigt. Wenn die XML-Input-Datei zu einem Projekt gehört und die Option *Ausgabefenster wiederverwenden* deaktiviert ist, wird diese Einstellung nur dann aktiv, wenn auch der unter dem Feld *In Ordner speichern* definierte Ausgabedateipfad (*Abbildung unten*) in den entsprechenden Projekteigenschaften[1344] **ebenfalls** deaktiviert ist.



- *Dateierweiterung aus xsl:output-Element verwenden:* Gibt an, ob die im Element **`xsl:output`** des XSLT-Stylesheet definierte Dateierweiterung die in der ersten Option dieser Liste definierte Standarddateierweiterung außer Kraft setzen soll.

XSL-FO-Transformationseinstellungen

FO-Dokumente werden mit Hilfe des FO-Prozessors verarbeitet. Der Pfad zur ausführbaren Datei des FO-Prozessors muss im Textfeld für den XSL-FO-Transformationsprozessor definiert werden. Die Transformation wird mit Hilfe des Menübefehls **XSL/XQuery | XSL-FO-Transformation**[1412] ausgeführt. Wenn es sich bei der Quelldatei (dem Dokument, das bei Ausführung des Befehls in der IDE aktiv ist) um ein XSL-FO-Dokument handelt, wird für die Transformation der FO-Prozessor aufgerufen. Wenn es sich beim Quelldokument um ein XML-Dokument handelt, muss das XML-Dokument zuerst mittels einer XSLT-Transformation in ein XSL-FO-Dokument konvertiert werden. Diese XSLT-Transformation kann entweder von dem für die Applikation als Standardprozessor definierten XSLT-Prozessor ausgeführt werden (*siehe oben*[1633]) oder vom XSLT-Prozessor, der möglicherweise bereits in den FO-Prozessor integriert ist, den Sie als Standard-FO-Prozessor für die Applikation definiert haben. Kicken Sie zur Auswahl zwischen den beiden Optionen auf das entsprechende Optionsfeld.

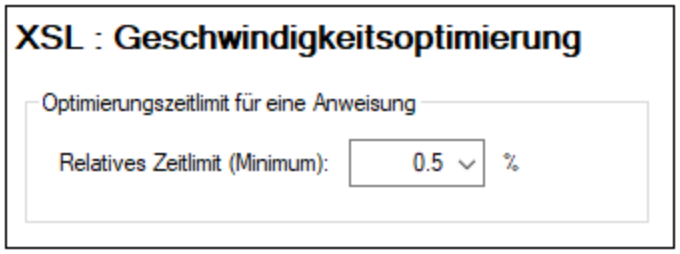
Anmerkung: Falls Sie die Option zur Installation des FOP-Prozessors des Apache XML-Projekts nicht deaktiviert haben, wurde dieser im Ordner `C:\ProgramData\Altova\SharedBetweenVersions` installiert. In diesem Fall wurde der Pfad zum Prozessor automatisch in das Eingabefeld "XSL-FO-Prozessor" eingegeben. Sie können den Pfad zu jedem gewünschten FO-Prozessor definieren. Beachten Sie, dass derselbe Pfad auch von anderen Altova-Produkten, die FO-Prozessoren verwenden (StyleVision und Authentic Desktop) verwendet wird.

Speichern und Beenden

Klicken Sie zum Abschluss auf **OK**.

28.17.15.9.1 Geschwindigkeitsoptimierung

Im Abschnitt **Geschwindigkeitsoptimierung** (*Abbildung unten*) können Sie Optionen für die XSL-Geschwindigkeitsoptimierung 512 definieren.



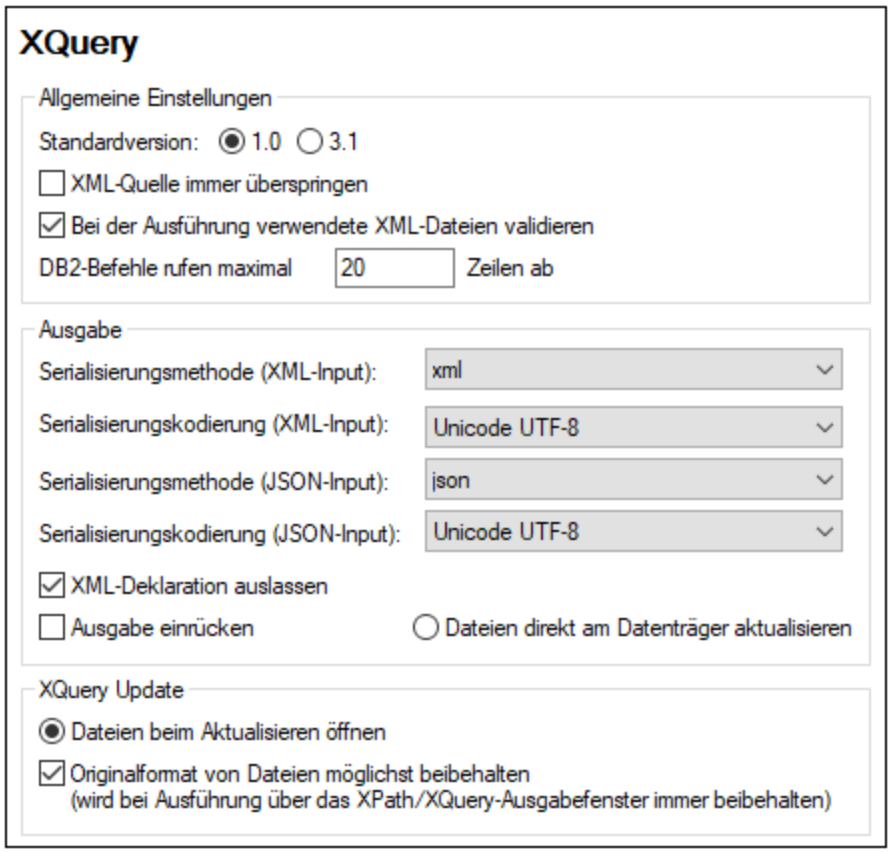
Sie können für die Optimierung ein Zeitlimit für einzelne XSLT-Anweisungen in einem XSLT Stylesheet definieren. Der Wertebereich reicht von 0,1% der Gesamttransformationsdauer bis 99%. Wenn die Ausführung einer Anweisung länger dauert, als im Zeitlimit definiert, so wird die Optimierungsanalyse aufgerufen. Andernfalls wird keine Analyse durchgeführt. Wenn die Optimierungsanalyse fehlschlägt, könnte der Grund dafür sein, dass das Zeitlimit zu hoch ist. Setzen Sie es in diesem Fall herab.

Speichern und Beenden

Klicken Sie zum Abschluss auf **OK**.

28.17.15.10 XQuery

Im Abschnitt **XQuery** (*Abbildung unten*) werden Optionen im Zusammenhang mit der Bearbeitung und Ausführung von XQuery- und XQuery Update-Dokumenten definiert.



Allgemeine XQuery-Optionen

Die folgenden Optionen stehen zur Verfügung:

- *Serialisierung:* Serialisierung bezieht sich auf die Art, auf die Text in das Ausgabedokument geschrieben wird. Sie können die Serialisierungsmethode (adaptive*, HTML, JSON, text, XHTML oder XML) und die Serialisierungskodierung der Ausgabe verschiedener Input-Arten auswählen. Die Ausgabeserialisierung kann für XML- und JSON-Inputs separat ausgewählt werden. (*Anmerkung*: bei der adaptiven Methode kann ein Instanzdokument fehlerfrei verarbeitet werden; damit wird die Serialisierungsmethode automatisch auf Basis des Input-Dokuments ermittelt.)
- *XML-Deklaration auslassen:* Lässt die XML-Deklaration im serialisierten Ausgabedokument weg.
- *Ausgabe einrücken:* Rückt das Ausgabedokument ein, um die Dokumenthierarchie darzustellen.
- *XML-Quelle immer überspringen:* Bei der Ausführung eines XQuery-Dokuments kann XMLSpy Sie nach einem XML-Quelldokument fragen, anhand dessen das XQuery-Dokument ausgeführt werden soll. Dabei wird ein Dialogfeld angezeigt, in dem Sie zur XML-Datei navigieren können. Aktivieren Sie diese Option, um dieses Dialogfeld zu überspringen und das XQuery-Dokument direkt auszuführen. Wenn diese Option ausgewählt ist, sollte das XQuery-Dokument korrekt ausgeführt werden können, ohne dass ein XML-Dokument übergeben wird, entweder, weil kein XML-Dokument benötigt wird oder weil XML-Daten über Funktionen im XQuery-Dokument aufgerufen werden.
- *XML-Dateien validieren:* Validiert die bei der Ausführung des XQuery-Dokuments verwendeten XML-Dateien. Ungültige XML-Dateien werden gekennzeichnet und das XQuery-Dokument wird nicht verarbeitet.

- *DB2-Zeilenabruf:* Bei Anzeige von Datenbankdaten können Sie festlegen, wie viele Spalten maximal abgerufen werden sollen. XMLSpy erkennt `.xqr` Dateierweiterungen[1606] als XQuery-Dateien für Datenbanken.
- *XQuery-Standardversion:* Definiert, welche XQuery-Prozessorversion für die Ausführung von XQuery-Dokumenten, die kein `version` Schlüsselwort aufweisen, verwendet werden soll. Diese Einstellung gilt sowohl für XQuery- als auch für XQuery Update-Dokumente und wählt den XQuery-Standardprozessor aus.

XQuery Update-Optionen

Die folgenden XQuery Update-Optionen stehen zur Verfügung:

- *Aktualisierung:* Wenn eine XQuery Update-Datei ausgeführt wird, können XML-Zieldateien entweder direkt am Datenträger aktualisiert werden oder in XMLSpy geöffnet und im Arbeitsspeicher aktualisiert werden. Mit der Option *Dateien beim Aktualisieren öffnen* haben Sie die Möglichkeit, die Aktualisierungen zu überprüfen und die Datei entweder auf der Festplatte zu speichern oder die Aktualisierungen abzulehnen (indem Sie die Datei schließen, ohne sie zu speichern).
- *Originalformat beibehalten:* Behält das Originalformat des aktualisierten Dokuments so weit wie möglich bei.

Speichern und Beenden

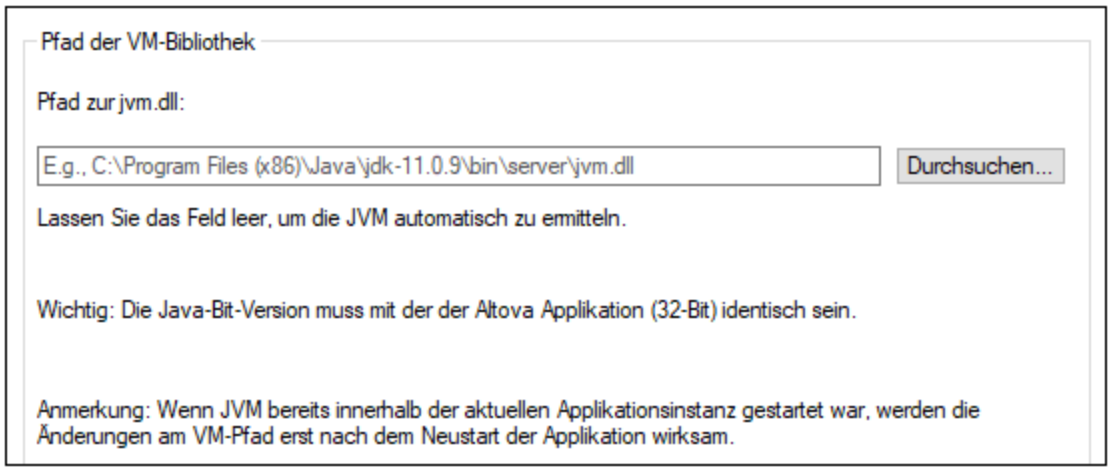
Klicken Sie zum Abschluss auf **OK**.

28.17.15.11 Java

Im Abschnitt *Java* (*siehe Abbildung unten*) haben Sie die Möglichkeit, den Pfad zu einer Java VM (Virtual Machine) auf Ihrem Dateisystem einzugeben. Beachten Sie, dass dies nicht immer notwendig ist. XMLSpy versucht standardmäßig den Java VM-Pfad automatisch zu ermitteln. Dazu wird zuerst die Windows Registry und anschließend die JAVA_HOME-Umgebungsvariable gelesen. Ein in dieses Dialogfeld eingegebener benutzerdefinierter Pfad hat Vorrang vor allen automatisch ermittelten Java VM-Pfaden.

Wenn Sie eine Java Virtual Machine verwenden, die keinen Installer hat und keine Registry-Einträge erstellt (z.B. OpenJDK von Oracle), müssen Sie eventuell einen benutzerdefinierten Java VM-Pfad angeben. Auch wenn Sie automatisch von XMLSpy ermittelte Java VM-Pfade aus irgendeinem Grund außer Kraft setzen müssen, müssen Sie diesen Pfad eventuell definieren.

© 2016-2022 Altova GmbH



Beachten Sie dazu Folgendes:

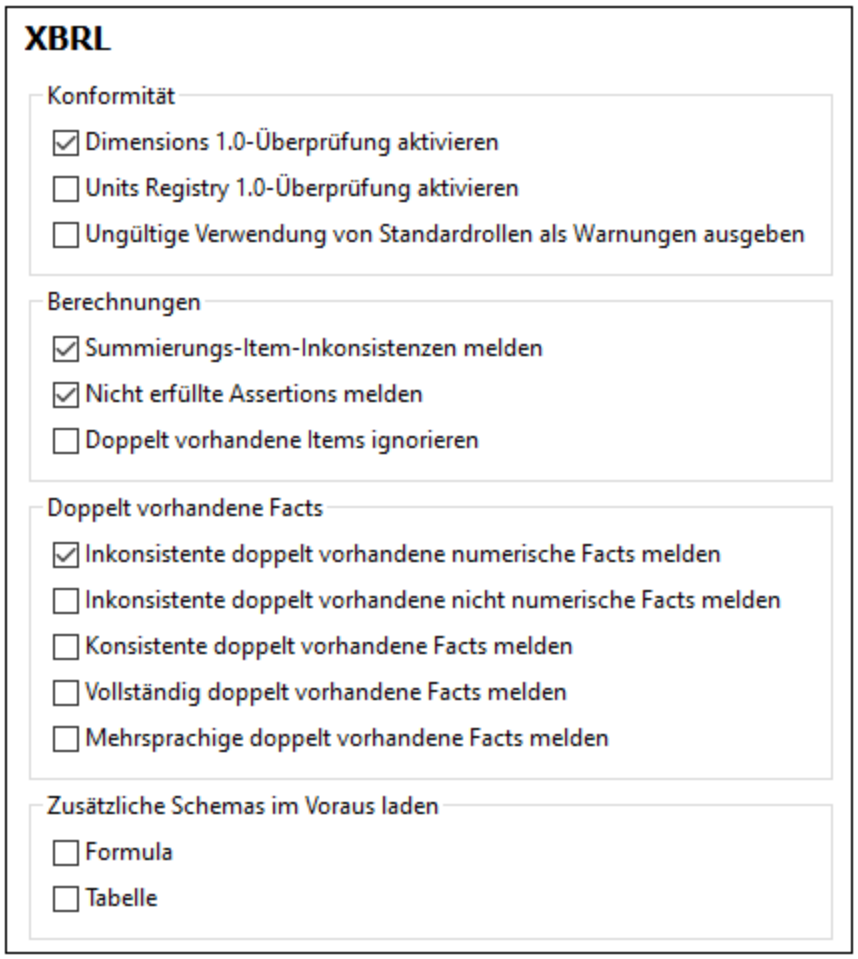
- Der Java VM-Pfad wird gemeinsam von allen Altova Desktop-Applikationen (nicht aber den Server-Applikationen) verwendet. Wenn Sie den Pfad daher in einer Applikation ändern, gilt dies automatisch auch für alle anderen Altova-Applikationen.
- Der Pfad muss auf die Datei **`jvm.dll`** im Verzeichnis **`\bin\server`** oder **`\bin\client`** (relativ zum Verzeichnis, in dem JDK installiert ist) verweisen.
- Die XMLSpy-Plattform (32-Bit, 64-Bit) muss mit der des JDK identisch sein.
- Nachdem Sie den Java VM-Pfad geändert haben, müssen Sie XMLSpy eventuell neu starten, damit die neuen Einstellungen wirksam werden.

Eine Änderung des Java VM-Pfads wirkt sich auf die folgenden Bereiche aus:

- JDBC-Verbindungen
- Java-Erweiterungsfunktionen für XSLT/XPath

28.17.15.12 XBRL

Der Abschnitt **XBRL** enthält Optionen, für die Validierung und Verarbeitung von XBRL- und XULE-Instanzdokumenten, von Inline XBRL und zum Hinzufügen und Verwalten von XBRL-Taxonomiepaketen[1641].



XBRL-Validierung

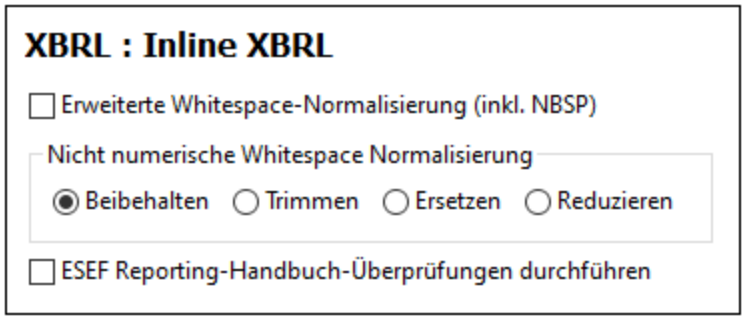
Auf dem obersten Register der XBRL-Optionen (*Abbildung unten*) können Sie die folgenden Validierungsoptionen definieren:

- Implementierung der Überprüfung auf Konformität mit Dimensions 1.0 und Units Registry 1.0. Es gibt auch eine Option, um die ungültige Verwendung von Standardrollen als Warnung auszugeben.
- Ob Meldungen über Berechnungen, (i) bei denen bei der Berechnung von Summen Inkonsistenzen auftreten, (i) bei denen Assertions nicht erfüllt werden, ausgegeben werden sollen.
- Nicht-Berücksichtigung von Duplikaten (vollständige und konsistente Duplikate) im Zusammenhang mit Berechnungen. Wenn ein Fact dupliziert wurde, wird das genauere Fact verwendet und die anderen ignoriert.
- Berichte über Duplikate. Nähere Informationen zur Einteilung von Duplikaten finden Sie in der folgenden Spezifikation: Handling Duplicate Facts in XBRL and Inline XBRL 1.0.
- Ob die Standard-Schemas für Formulas und/oder Tabellen im Vorhinein geladen werden sollen.

© 2016-2022 Altova GmbH

28.17.15.12.1 Inline XBRL

Das Register Inline XBRL (*Abbildung unten*) enthält Optionen für die Normalisierung von Whitespace-Zeichen (Leerzeichen, Tabulatoren, Zeilenschaltungen und Wagenrücklauf) in Inline XBRL,



Es stehen die folgenden Optionen zur Verfügung:

- Wenn die Option *Erweiterte Whitespace-Normalisierung* aktiviert ist, werden auch geschützte Leerzeichen als Whitespace-Zeichen betrachtet.
- Mit der Option *Beibehalten* werden Whitespace-Zeichen unverändert beibehalten.
- Mit der Option *Trimmen* werden Whitespace-Zeichen auf beiden Seiten des String entfernt.
- Mit der Option *Ersetzen* werden alle Instanzen von Tabulator, Zeilenschaltung und Wagenrücklauf durch ein Leerzeichen ersetzt.
- *Reduzieren* ist eine Erweiterung von *Ersetzen*. Hierbei wird eine Ersetzung durchgeführt. Anschließend werden alle aufeinanderfolgenden Leerzeichen zu einem einzigen Leerzeichen reduziert.
- *ESEF Reporting-Handbuch-Überprüfungen durchführen:* Ermöglicht bei der Validierung eines Inline XBRL-Dokuments die Überprüfung auf die Konformität mit dem ESEF Reporting-Handbuch. Nähere Informationen über ESEF finden Sie auf der European Single Electronic Format (ESEF)-Webseite.

28.17.15.12.2 Taxonomiepakete

Ein XBRL-Taxonomiepaket ist ein gezipptes Archiv, das eine Offline-Kopie einer Taxonomie enthält. Das Taxonomiepaket enthält eine XML-Katalogdatei, die URIs auf die Dateipfade der Offline-Taxonomie mappt, so dass die Taxonomie Applikationen offline zur Verfügung steht. Die Regeln zur Strukturierung und dem Aufbau von Taxonomiepaketen finden Sie in der Taxonomy Packages Recommendation von XBRL.org.

Nachdem Sie ein Taxonomiepaket heruntergeladen haben, können Sie XMLSpy so konfigurieren, dass die Eintrittspunkt-Katalogdateien automatisch erkannt und verwendet wird. Fügen Sie dazu das Paket zur Liste der aktiven Taxonomiepakete hinzu. Daraufhin werden die Katalogdateien von aktiven Paketen zum Auffinden von Ressourcen für Operationen wie die XBRL-Validierung verwendet. Sie können zwei Arten von Taxonomiepaketen hinzufügen:

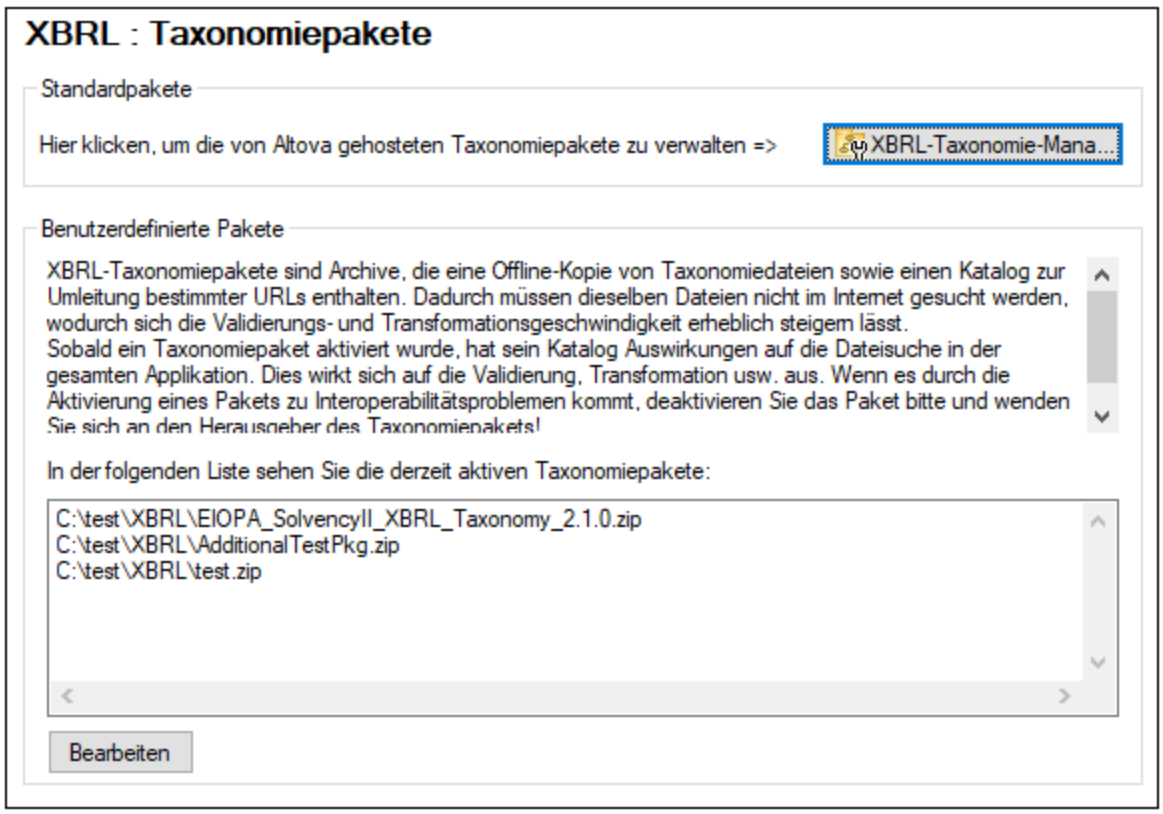
- Standardpakete, die für alle Altova-Produkte einfach und effizient mit dem Altova XBRL-Taxonomie-Manager 831 (Beschreibung hier 831) verwaltet werden können.
- Benutzerdefinierte Pakete, die Sie über dieses Dialogfeld hinzufügen und verwalten können.

Anmerkung: Eine Ressource, die durch die Katalogdatei eines aktiven Pakets referenziert wird, wird für alle XMLSpy-Operationen, die diese benötigen, verwendet. Wenn sich eine solche Ressource auf

irgendeine Art von der zuvor von XMLSpy verwendeten unterscheidet, kann dies bei der Ausführung von Operationen zu Fehlern führen. Lesen Sie dazu bitte den Warnhinweis am Ende dieses Kapitels.

Hinzufügen und Verwalten von Taxonomiepaketen

Wählen Sie den Befehl **Extras | Optionen | Taxonomiepakete,** um das Fenster "Taxonomiepakete" aufzurufen (*Abbildung unten*).

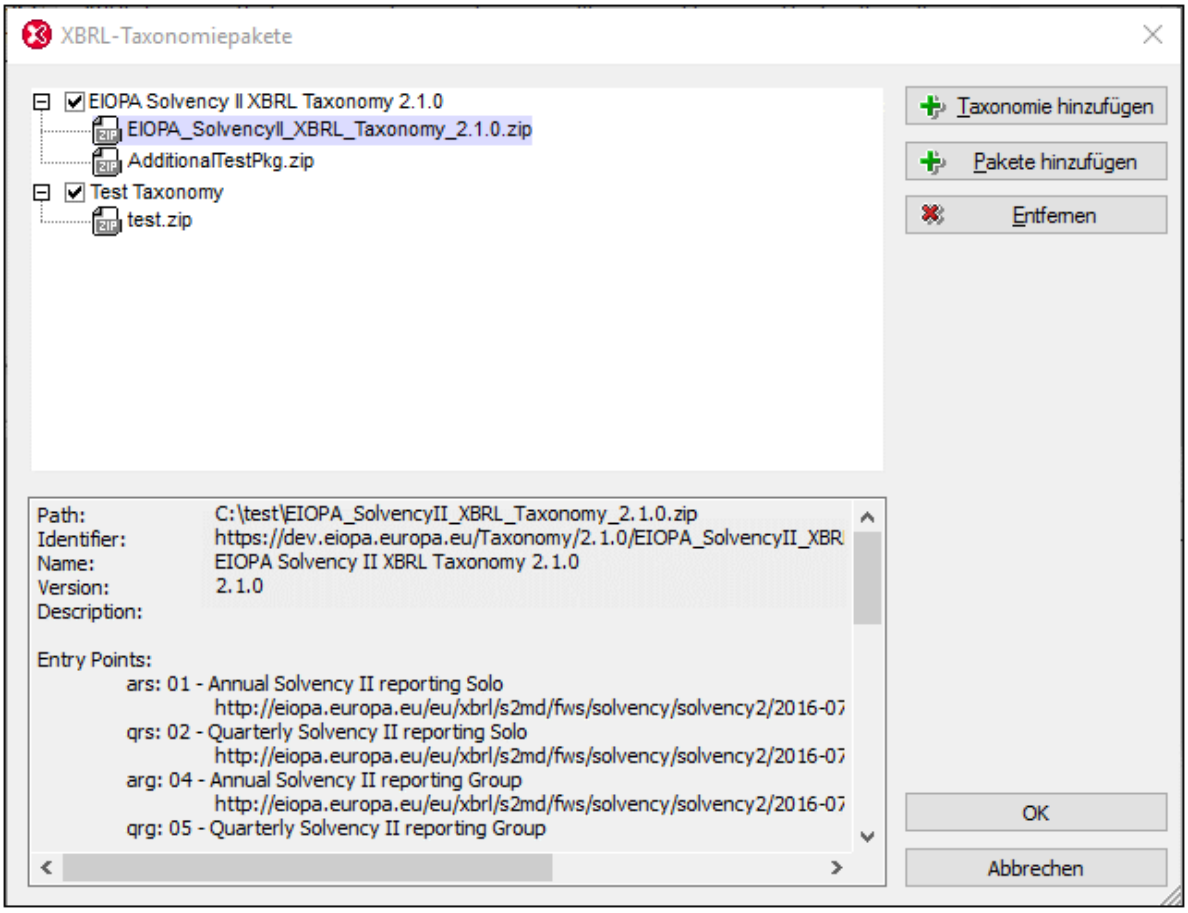


Um ein Standard-Taxonomiepaket hinzuzufügen, klicken Sie auf **XBRL-Taxonomie-Manager**. Eine Anleitung zur Verwendung des Altova XBRL-Taxonomie-Managers[831] finden Sie hier[831].

Um ein benutzerdefiniertes Paket hinzuzufügen, gehen Sie folgendermaßen vor:

1. Klicken Sie auf **Bearbeiten** (*siehe Abbildung oben*), um das Dialogfeld "XBRL-Taxonomiepakete" aufzurufen (*Abbildung unten*).

© 2016-2022 Altova GmbH



2. Klicken Sie auf **Taxonomie hinzufügen,** navigieren Sie zum gewünschten Taxonomiepaket, wählen Sie es aus und klicken Sie auf **Öffnen**. (Sie können auch mehrere Pakete auswählen, um diese auf einmal hinzuzufügen.) Das Taxonomiepaket wird daraufhin zur Liste der Taxonomiepakete im Dialogfeld hinzugefügt. Die Liste wird als Struktur bestehend aus zwei Ebenen angezeigt. Auf der ersten Ebene wird die Taxonomie angezeigt; auf der zweiten Ebene sehen Sie die Pakete der Taxonomie. Das Kontrollkästchen links vom Taxonomieeintrag zeigt an, ob diese Taxonomie aktiv ist oder nicht. Eine neu hinzugefügte Taxonomie ist standardmäßig aktiv.
3. Klicken Sie abschließend auf **OK**. Die neu hinzugefügten Taxonomiepakete werden auf dem Register "Taxonomiepakete" des Dialogfelds "Optionen" angezeigt (*erste Abbildung oben*).

Beachten Sie die folgenden Punkte:

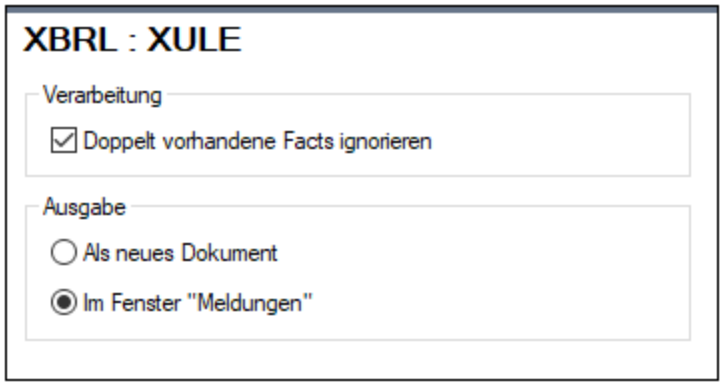
- Wenn Sie weitere Pakete zu einer Taxonomie hinzufügen möchten, wählen Sie die Taxonomie im Dialogfeld "XBRL-Taxonomiepakete" aus (*Abbildung oben*) und fügen Sie das/die zusätzliche(n) Paket(e) anschließend über die Schaltfläche **Pakete hinzufügen** hinzu. Das/Die hinzugefügte(n) Paket(e) wird/werden auf der zweiten Ebene dieser Taxonomie angezeigt.
- Wenn ein Taxonomiepaket im oberen Bereich des Dialogfelds "XBRL-Taxonomiepakete" ausgewählt ist, werden die Einzelheiten dazu (darunter auch der Offline-Pfad der Pakets) im unteren Bereich des Dialogfelds angezeigt (*siehe Abbildung oben*).
- Um eine Taxonomie zu deaktivieren, deaktivieren Sie das Kontrollkästchen dieser Taxonomie. Wenn Sie eine Taxonomie deaktivieren, wird/werden ihre Katalogdatei(en) nicht verwendet. Eine Taxonomie zu deaktivieren ist nützlich, wenn Sie z.B. zwischen zwei Versionen einer Taxonomie hin- und herwechseln möchten.

- Um ein Paket zu entfernen, wählen Sie es aus und klicken Sie auf **Entfernen**.
- Die folgenden Altova-Produkte unterstützen die Registrierung von Taxonomiepaketen: XMLSpy, MapForce und StyleVision. Die Liste der Taxonomiepakete wird von allen diesen Applikationen gemeinsam verwendet. Wenn Sie die Liste in einer Applikation bearbeiten, so wird die bearbeitete Liste auch in den anderen Applikationen angezeigt. Wenn Sie die Paketliste in einer Applikation bearbeiten und eine weitere Applikation zur selben Zeit offen ist, so wird in der anderen Applikation eine Meldung angezeigt, in der Sie gefragt werden, ob Sie die Paketliste neu laden möchten, damit die Änderungen zu sehen sind.

Vorsicht: Paketkataloge können eine Umleitung zu inkompatiblen Ressourcen verursachen
Eine Ressource, die durch die Katalogdatei eines aktiven Pakets referenziert wird, wird für alle XMLSpy-Operationen, die diese benötigen, verwendet. Ein Beispiel dafür wäre ein XML-Schema, das sowohl für die XML- als auch für die XBRL-Validierung verwendet wird. Wenn die von der Katalogdatei des Pakets referenzierte Offline-Ressource mit Ihrer vorhandenen Umgebung nicht kompatibel ist, kann dies zu Fehlern führen. Deaktivieren Sie in diesem Fall das Taxonomiepaket und teilen Sie den Fehler den Verfassern des Pakets mit.

28.17.15.12.3 XULE

Das Register XBRL XULE (*siehe Abbildung unten)* bietet Optionen für die XULE-Verarbeitung.

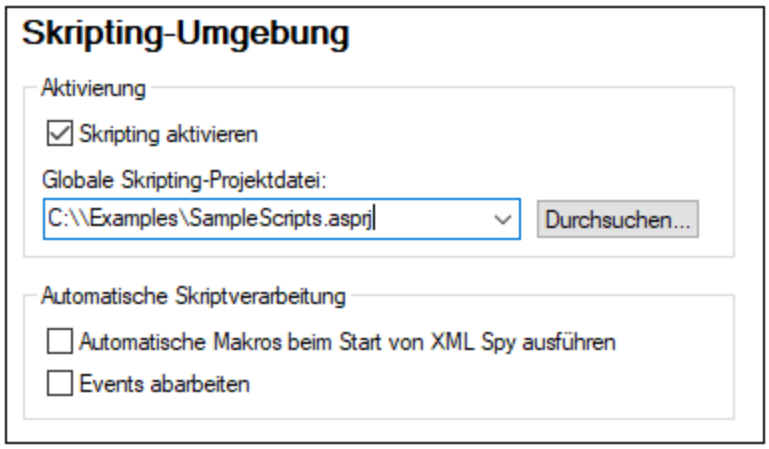


- *Doppelt vorhandene Facts ignorieren:* Ein doppelt vorhandenes Fact kommt - meist in Inline XBRL-Dokumenten - dann vor, wenn dasselbe Fact im HTML-Code mehrmals ausgegeben wird. Mit dieser Option legen Sie fest, dass das doppelt vorhandene Fact nur einmal ausgegeben wird.
- *Ausgabe:* Wenn ein XULE-Dokument mit dem Befehl **XBRL | XULE ausführen** verarbeitet wird, kann die Ausgabe (i) im Fenster "Meldungen" oder (ii) in einem neuen Dokument generiert werden, das in XMLSpy angezeigt und temporär im Arbeitsspeicher gespeichert wird; dieses Dokument kann mit dem Befehl **Datei | Speichern unter**[1286] in einer Datei gespeichert werden.

© 2016-2022 Altova GmbH

28.17.15.13 Skripting-Umgebung

Im Abschnitt **Skripting-Umgebung** (*Abbildung unten*) können Sie die Skripting-Umgebung[1557] beim Start der Applikation aktivieren. Aktivieren Sie dazu das Kontrollkästchen *Skripting aktivieren*. Anschließend können Sie die globale Skripting-Projektdatei definieren. (*siehe Abbildung unten*).



Um ein globales Skripting-Projekt für XMLSpy zu definieren, aktivieren Sie das Kontrollkästchen *Skripting aktivieren* und suchen Sie die gewünschte Altova Skripting-Projektdatei (`.asprj`-Datei). Außerdem können Sie folgende Einstellungen definieren: (i) ob automatische Makros im Skripting-Projekt beim Start von XMLSpy automatisch ausgeführt werden sollen und (ii) ob Application Event Handler-Skripts im Projekt automatisch ausgeführt werden sollen oder nicht; aktivieren bzw. deaktivieren Sie das entsprechende Kontrollkästchen.

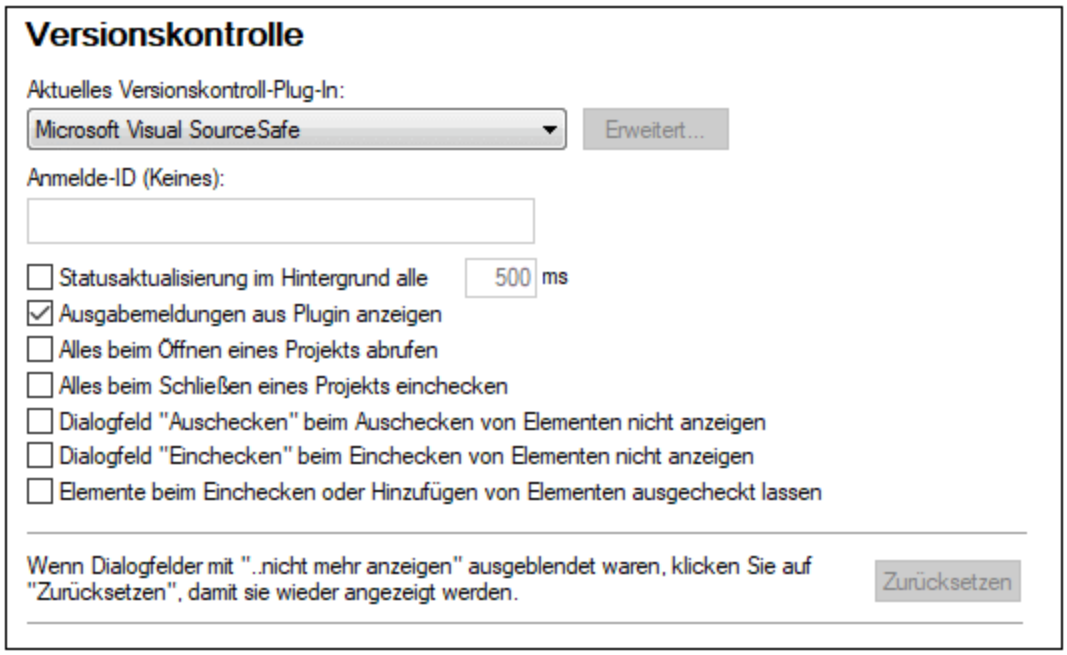
Klicken Sie anschließend auf **OK**. Die Makros im globalen Skripting-Projekt werden nun im Untermenü des Befehls **Makros** angezeigt.

Speichern und Beenden

Klicken Sie anschließend auf **OK**. Die Makros im globalen Skripting-Projekt werden nun im Untermenü des Befehls **Makros** angezeigt.

28.17.15.14 Versionskontrolle

Im Abschnitt **Versionskontrolle** (*Abbildung unten*) können Sie den Versionskontrollprovider, die Einstellungen und die Standard-Anmelde-IDs für die einzelnen Versionskontrollsysteme definieren.



Versionskontroll-Plugin

Das aktuelle Versionskontroll-Plugin kann aus den derzeit installierten Versionskontrollsystemen ausgewählt werden. Diese werden in der Dropdown-Auswahlliste aufgelistet. Definieren Sie nach Auswahl der gewünschten Versionskontrolle im nächsten Textfeld die Anmelde-ID dafür. Über die Schaltfläche **Erweitert** rufen Sie ein Dialogfeld zum jeweiligen Versionskontroll-Plugin auf, in dem Sie die Einstellungen für dieses Plugin definieren können. Diese Einstellungen unterscheiden sich je nach Plugin.

Benutzereinstellungen

Es steht eine Reihe von Benutzereinstellungen zur Verfügung, darunter die folgenden:

- Statusaktualisierungen können im Hintergrund in benutzerdefinierten Zeitabständen durchgeführt oder komplett deaktiviert werden. Sehr große Versionskontrolldatenbanken könnten beträchtlich viel Arbeitsspeicher beanspruchen. Indem Sie die Statusaktualisierungen im Hintergrund deaktivieren oder das Zeitintervall vergrößern, können Sie Ressourcen frei machen, sodass Ihr System schneller arbeitet.
- Dateien können beim Öffnen und Schließen von Projekten automatisch aus- bzw. eingecheckt werden.
- Die Anzeige der Dialogfelder "Einchecken" und "Auschecken" kann unterdrückt werden.
- Die Schaltfläche "**Zurücksetzen**" steht zur Verfügung, wenn Sie in einem der Dialogfelder die Option "*Wenn Dialogfelder mit..."*aktiviert haben. Wenn Sie auf die Schaltfläche **Zurücksetzen** klicken, wird diese Option wieder aktiviert.

Speichern und Beenden

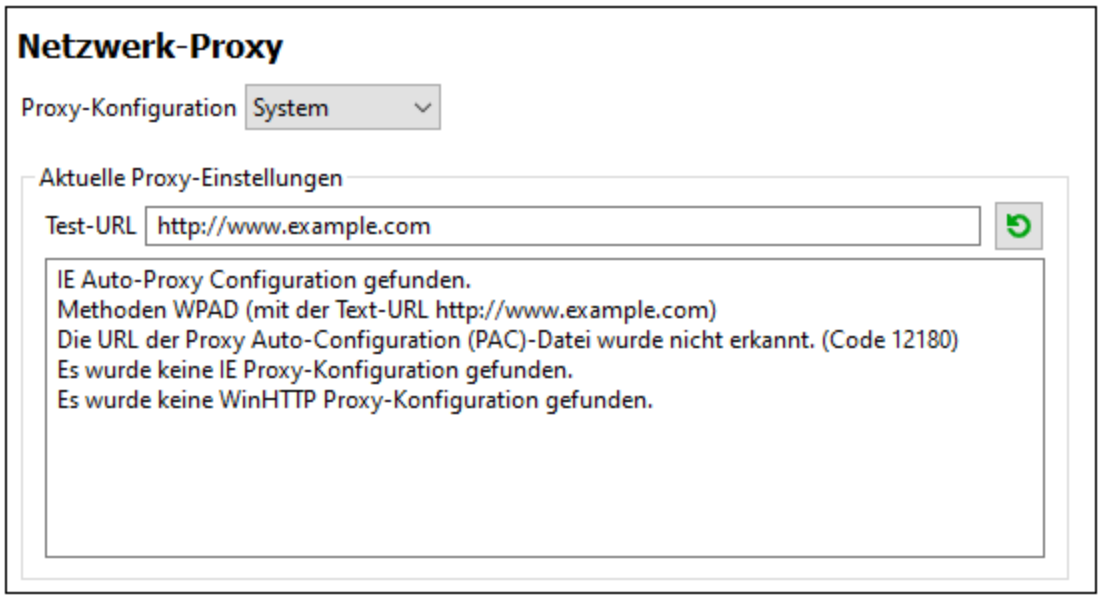
Klicken Sie zum Abschluss auf **OK**.

© 2016-2022 Altova GmbH

28.17.15.15 Netzwerk-Proxy

Im Abschnitt *Netzwerk-Proxy* können Sie die benutzerdefinierten Proxy-Einstellungen konfigurieren. Diese Einstellungen beeinflussen, wie die Applikation eine Verbindung mit dem Internet herstellt (z.B. zur XML-Validierung). Standardmäßig werden die Proxy-Einstellungen des Systems verwendet, d.h. die Einstellungen funktionieren, ohne dass der Benutzer etwas daran ändern muss. Falls nötig, können Sie jedoch einen anderen Netzwerk-Proxy-Server definieren. Wählen Sie dazu in der Auswahlliste *Proxy-Konfiguration* entweder die Option *Automatisch* oder *Manuell*, um die Einstellungen entsprechend zu konfigurieren.

Anmerkung: Die Nethwerk-Proxy-Einstellungen werden von allen Altova MissionKit-Applikationen gemeinsam verwendet. Wenn Sie daher die Einstellungen in einer Applikation ändern, wirkt sich dies automatisch auf alle anderen Applikationen aus.



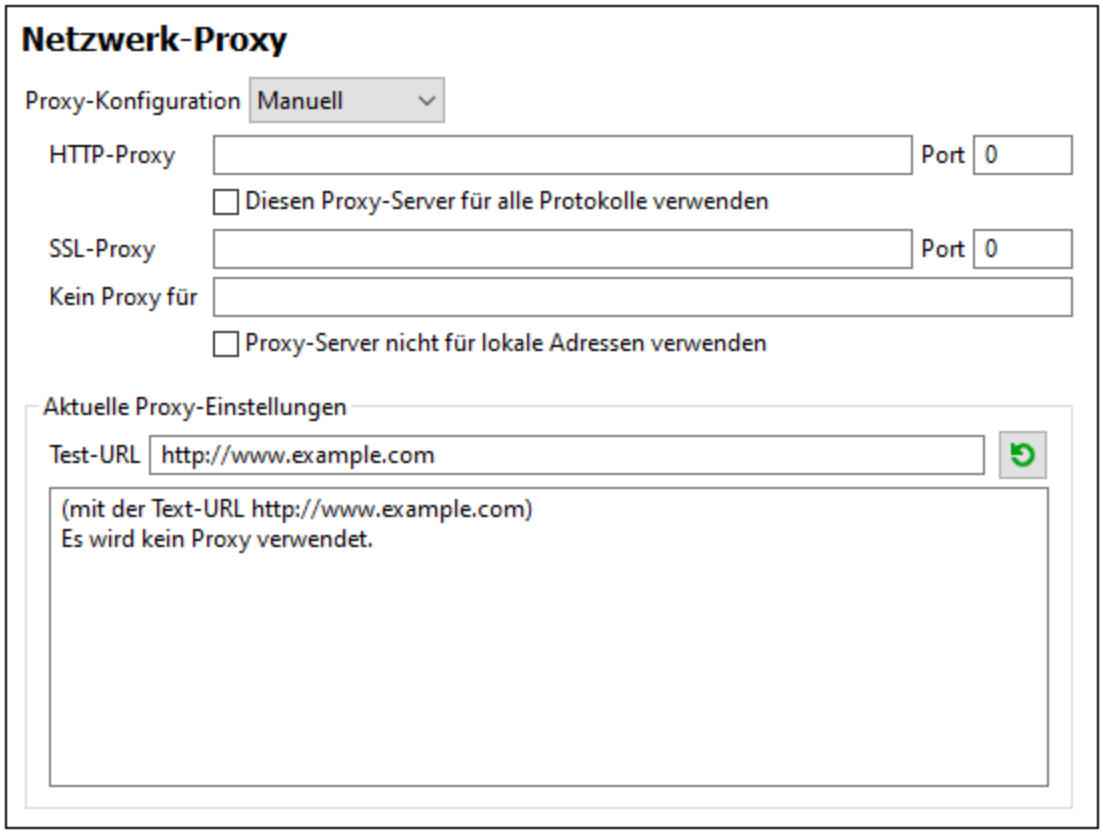
System-Proxy-Einstellungen verwenden
Dadurch werden die über die System-Proxy-Einstellungen konfigurierbaren Internet Explorer (IE)-Einstellungen verwendet. Führt außerdem eine Abfrage der mit **`netsh.exe winhttp`** konfigurierten Einstellungen durch.

Automatische Proxy-Konfiguration
Es stehen die folgenden Optionen zur Verfügung:

- *Einstellungen automatisch ermitteln:* **`Verwendet ein WPAD-Skript`** (http://wpad.LOCALDOMAIN/wpad.dat) über DHCP oder DNS, um die Einrichtung des Proxy-Servers zu konfigurieren.
- *Skript-URL:* Definieren Sie eine HTTP URL zu einem automatischen Proxy-Konfigurationsskript (**`.pac`**), mit dem der Proxy-Server eingerichtet wird.
- *Neu laden:* Setzt die aktuelle automatische Proxy-Konfiguration zurück und lädt sie neu. Dafür ist Windows 8 oder neuer erforderlich. Die Rücksetzung kann bis zu 30 Sekunden dauern.

Manuelle Proxy-Konfiguration
Definieren Sie den vollständig qualifizierten Host-Namen und Port für die Proxy-Server der jeweiligen Protokolle manuell. Im Host-Namen kann ein unterstütztes Schema inkludiert werden (z.B.: **`http://hostname`**). Das

Schema muss nicht mit dem entsprechenden Protokoll übereinstimmen, wenn der Proxy-Server das Schema unterstützt.



Es stehen die folgenden Optionen zur Verfügung:

- *HTTP-Proxy:* Verwendet den angegebenen Host-Namen und Port für das HTTP-Protokoll. Wenn *Diesen Proxy-Server für alle Protokolle verwenden* aktiviert ist, wird der angegebene HTTP-Proxy-Server für alle Protokolle verwendet.
- *SSL-Proxy:* Verwendet den angegebenen Host-Namen und Port für das SSL-Protokoll.
- *Kein Proxy für:* eine durch Semikola (`;`) getrennte Liste von voll qualifizierten Host-Namen, Domain-Namen oder IP-Adressen für Hosts, die ohne einen Proxy-Server verwendet werden sollen. IP-Adressen dürfen nicht abgeschnitten werden und IPv6-Adressen müssen innerhalb von eckige Klammern gesetzt werden (z.B.: `[2606:2800:220:1:248:1893:25c8:1946]`). Domain-Namen muss ein Punkt vorangestellt werden (z.B: `.example.com`).
- *Proxy-Server nicht für lokale Adressen verwenden:* Falls dieses Kontrollkästchen aktiviert ist, wird `<local>` zur *Kein Proxy für*-Liste hinzugefügt. Falls diese Option ausgewählt ist, wird für die folgenden Adressen kein Proxy-Server verwendet: (i) `127.0.0.1`, (ii) [::1], (iii) alle Host-Namen, die kein Punktzeichen (`.`) enthalten.

Aktuelle Proxy-Einstellungen
Stellt ein ausführliches Protokoll der Proxy-Ermittlung bereit. Es kann über die Schaltfläche **Aktualisieren** rechts vom Feld *Test-URL* aktualisiert werden (z.B. bei Wechsel zu einer anderen Test-URL oder wenn die Proxy-Einstellungen geändert wurden).

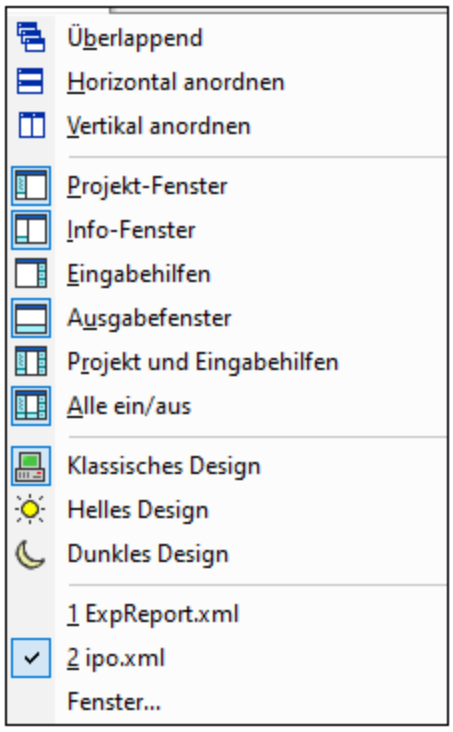
- *Test-URL:* Anhand einer Test-URL kann ermittelt werden, welcher Proxy-Server für diese bestimmte

 © 2016-2022 Altova GmbH

URL verwendet wird. Mit dieser URL erfolgt kein I/O. Dieses Feld darf nicht leer sein, wenn die automatische Proxy-Konfiguration verwendet wird (entweder über *System-Proxy-Einstellungen verwenden* oder *Automatische Proxy-Konfiguration*).

28.18 Menü "Fenster"

Das Menü **Fenster** enthält Befehle, mit denen Sie einzelne Applikations- und Dokumentfenster der Benutzeroberfläche verwalten können. Sie können die offenen Dokumente überlappend oder nebeneinander anzeigen sowie Eingabehilfenfenster und Ausgabefenster anordnen und ein- oder ausblenden.



Überlappend, horizontal/vertikal anordnen

Mit dem Befehl **Überlappend** ordnen Sie Dokumentfenster gestaffelt übereinander von hinten nach vorne an.

Mit den Befehlen **Horizontal anordnen** und **Vertikal anordnen** arrangieren Sie alle offenen, nicht minimierten Dokumente horizontal nebeneinander bzw. vertikal übereinander, sodass alle Dokumentfenster gleichzeitig im Applikationsfenster sichtbar sind.

Projekt-Fenster, Info-Fenster, Eingabehilfen, Ausgabefenster

Mit diesen Befehlen blenden Sie das Projekt-Fenster[118], bzw. das Info-Fenster[120], die Eingabehilfen[121] bzw. Ausgabefenster[122] ein oder aus.

Jedes dieser Fenster kann an- und abgedockt werden. Wenn Sie es an der Titelleiste ziehen, können Sie es von der aktuellen Position abdocken und es frei schwebend anzeigen. Wenn Sie mit der rechten Maustaste auf die Titelleiste klicken, können Sie das Fenster andocken oder ausblenden.

© 2016-2022 Altova GmbH

Projekt und Eingabehilfen

Mit diesem Befehl können Sie das Projekt-Fenster[118] und die Eingabehilfen[121] gleichzeitig ein- und ausblenden. Sie müssen diese Fenster dadurch nicht separat ein- und ausblenden.

Alle ein/aus

Mit diesem Befehl blenden Sie alle an- und abdockbaren Fenster (*siehe Liste unten*) ein oder aus.

- Projekt-Fenster[118]
- Info-Fenster[120]
- Eingabehilfen[121]
- Ausgabefenster[122]

Diese Funktion ist besonders hilfreich, wenn Sie schnell alle anderen Fenster ausblenden möchten, um das/die Dokument(e), an dem/denen Sie gerade arbeiten, in maximaler Größe anzeigen zu können.

Designs

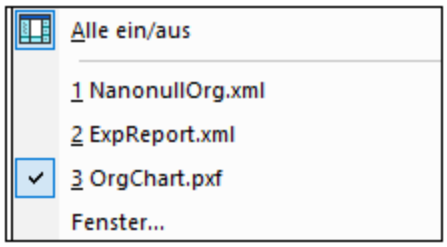
In XMLSpy stehen drei Designs zur Auswahl. Bei Auswahl eines Designs wird dieses sofort angewendet.

- Klassisch (Standardeinstellung)
- Hell
- Dunkel

Sie können die Formatierung der einzelnen Textkomponenten eines Dokumenttyps anpassen. Gehen Sie dazu zum Register *Schriftarten und Farben*[1622] des Dialogfelds "Optionen" (**Extras | Optionen**[1602]).

Liste der derzeit geöffneten Fenster

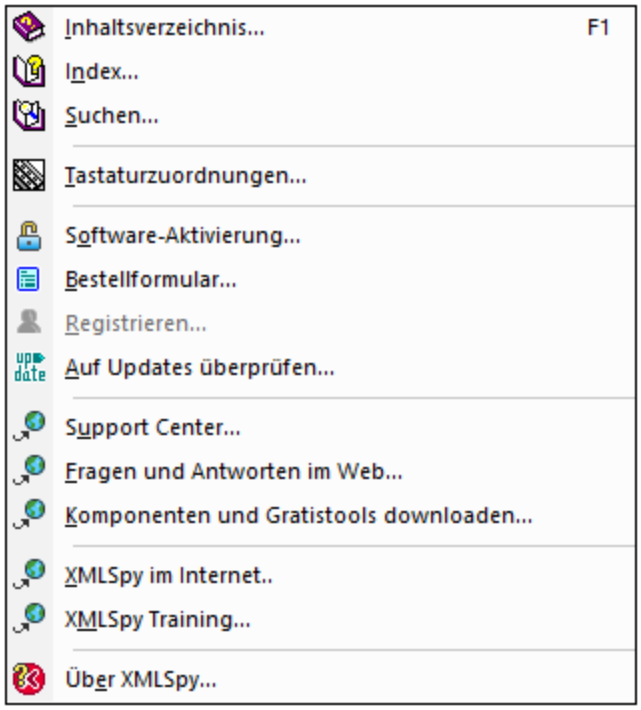
In der Liste sehen Sie alle offenen Fenster und können jederzeit zwischen diesen Fenstern wechseln.



Sie können auch mit Hilfe der Tastenkürzel **STRG+F6** zwischen den Fenstern wechseln.

28.19 Menü "Hilfe"

Im Hilfemenü finden Sie Befehle, die Sie benötigen, um weitere Informationen zu spezifischen Themen von XMLSpy zu erhalten, sowie Links zu Informationen und Supportseiten auf dem Altova Webserver.



Im Hilfemenü finden Sie weiters den Registrierungsdialog[1654], über den Sie Ihren Keycode eingeben können, sobald Sie das Produkt erworben haben.

28.19.1 Inhaltsverzeichnis, Index, Suche

- Inhaltsverzeichnis

  Mit dem Befehl "Inhaltsverzeichnis" wird die Bildschirmhilfe für XMLSpy geöffnet, wobei das Inhaltsverzeichnis auf der linken Seite des Hilfefensters angezeigt wird. Das Inhaltsverzeichnis bietet eine Übersicht über das gesamte Hilfedokument. Wenn Sie auf einen Eintrag im Inhaltsverzeichnis klicken, gelangen Sie zu diesem Thema.

- Index

  Mit dem Befehl Index wird die Bildschirmhilfe für XMLSpy geöffnet, wobei auf der linken Seite des Hilfefensters der Stichwortindex angezeigt wird. Im Index werden Stichwörter aufgelistet und Sie können durch Doppelklick auf das Stichwort zum jeweiligen Thema navigieren. Wenn ein Stichwort mit mehr als einem Thema verknüpft ist, können Sie das Thema aus einer Liste von Themen auswählen.

- Suchen

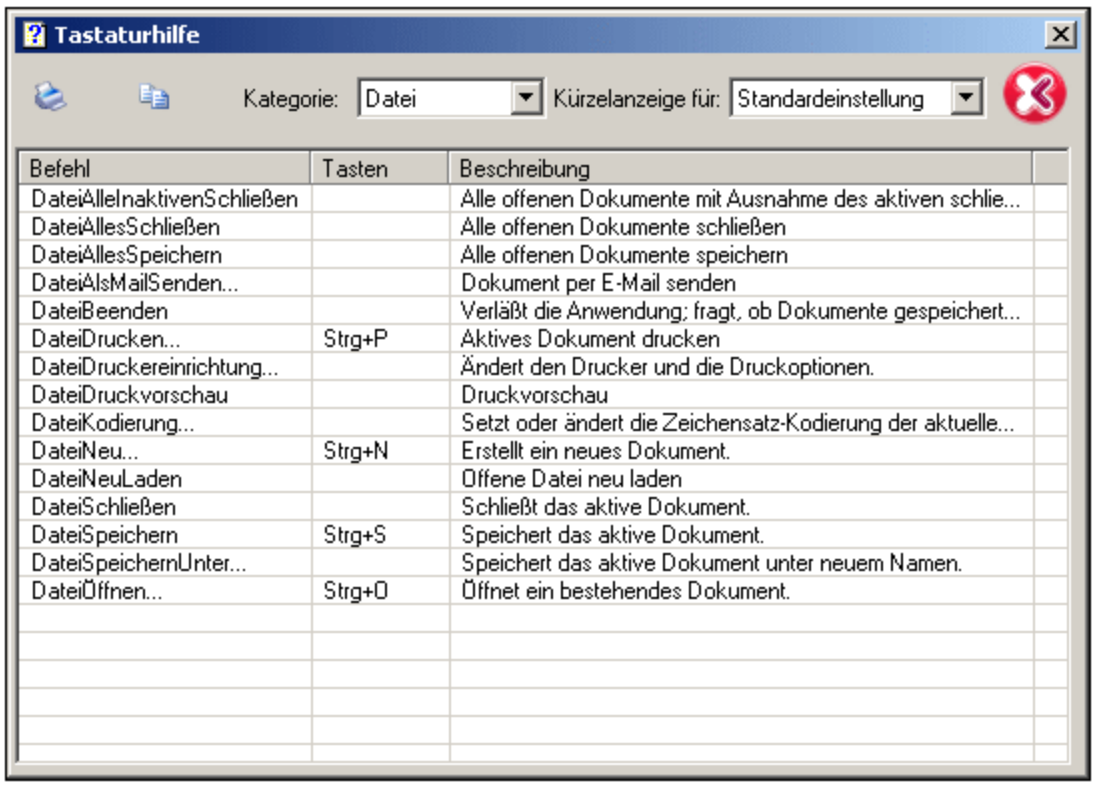
  Mit dem Befehl Suchen wird die Bildschirmhilfe für XMLSpy geöffnet, wobei auf der linken Seite des

 © 2016-2022 Altova GmbH

Hilfefensters das Suchdialogfeld angezeigt wird. Um nach einem Begriff zu suchen, geben Sie ihn in das Eingabefeld ein und drücken Sie die **Eingabetaste** oder klicken Sie auf **Kapitel auflisten**. Das Hilfesystem führt eine Volltextsuche am gesamten Hilfedokument durch und gibt eine Trefferliste zurück. Doppelklicken Sie auf einen Eintrag, um das Thema anzuzeigen.

28.19.2 Tastaturzuordnungen

Bei Auswahl des Befehls **Tastenzuordnungen** wird eine Infobox angezeigt, in der alle Befehle in XMLSpy nach Menüs geordnet angezeigt werden. Die Menübefehle werden mit einer Beschreibung und dem entsprechenden Tastaturkürzel für den jeweiligen Befehl angezeigt.



Um die Befehle eines bestimmten Menüs anzuzeigen, wählen Sie den Menünamen in der Dropdown-Liste "Kategorie" aus. Durch Klicken auf das Druckersymbol können Sie den Befehl ausdrucken.

Beachten Sie bitte die folgenden Anmerkungen zu Tastaturkürzeln:

- Bestimmte Befehle (und Ihre Tastaturkürzel) können nur in einer bestimmten Ansicht angewendet werden. So können z.B. die Befehle im Menü "XML" nur in der Grid-Ansicht angewendet werden. Andere Befehle (wie z.B. **Datei | Speichern** oder **XML | Wohlgeformtheit prüfen**) stehen in mehreren Ansichten zur Verfügung.
- Weitere nützliche Tastaturkürzel: Mit **Umschalt+F10** wird das Kontextmenü in der Textansicht und in der Schema-Ansicht aufgerufen; wenn Sie **Strg+E** drücken, während sich der Cursor in der Textansicht

innerhalb eines Start- oder End-Tags eines Elements befindet, springt der Cursor zum Ende- bzw. Start-Tag.
- Im Dialogfeld "Anpassen" können Sie auf dem Register "Tastatur" 1589 Ihre eigenen Tastaturkürzel für verschiedene Menübefehle definieren.

28.19.3 Aktivierung, Bestellformular, Registrieren, Updates

Software-Aktivierung

Lizenzieren Ihres Produkts
Nachdem Sie Ihre Altova-Software heruntergeladen haben, können Sie sie entweder mit Hilfe eines kostenlosen Evaluierungs-Keycode oder eines käuflich erworbenen permanenten Lizenzkeycode lizenzieren oder aktivieren.

- ***Kostenlose Evaluierungs-Lizenz.*** Wenn Sie die Software zum ersten Mal starten, wird das Dialogfeld **Software-Aktivierung** angezeigt. Es enthält eine Schaltfläche, über die Sie eine kostenlose Evaluierungs-Lizenz anfordern können. Geben Sie in das Dialogfeld Ihren Namen, den Namen Ihrer Firma und Ihre E-Mail-Adresse ein und klicken Sie auf **Anfordern**. Daraufhin wird an die von Ihnen angegebene E-Mail-Adresse eine Lizenz gesendet und sollte in wenigen Minuten eintreffen. Speichern Sie die Lizenzdatei in einem geeigneten Ordner.

  Wenn Sie auf **Anfordern** klicken, wird am unteren Rand des Dialogfelds ein Eingabefeld angezeigt. Über dieses Feld gelangen Sie zum Pfad der Lizenzdatei. Navigieren Sie zur Lizenzdatei oder geben Sie den Pfad dazu ein und klicken Sie auf **OK**. (Sie können im Dialogfeld **Software-Aktivierung** auch auf **Neue Lizenz hochladen** klicken, um das Dialogfeld, in das der Pfad zur Lizenzdatei eingegeben wird, aufzurufen). Die Software wird für einen Zeitraum von 30 Tagen aktiviert.

- ***Permanenter Lizenz-Keycode.*** Über das Dialogfeld **Software-Aktivierung** können Sie einen permanenten Lizenz-Keycode erwerben. Wenn Sie auf diese Schaltfläche klicken, gelangen Sie zum Altova Online Shop, in dem Sie einen permanenten Lizenzschlüssel für Ihr Produkt erwerben können. Ihre Lizenz wird Ihnen in Form einer Lizenzdatei, die Ihre Lizenzdaten enthält, per E-Mail zugesendet.

  Es gibt drei Arten von permanenten Lizenzen: *Einzelplatzlizenzen*, *Parallellizenzen* und *Named User-Lizenzen* (benutzerdefinierte Nutzung). Mit einer Einzelplatzlizenz wird die Software auf einem einzigen Rechner freigeschaltet. Wenn Sie eine Einzelplatzlizenz für *N* Rechner erwerben, gestattet Ihnen die Lizenz, die Software auf bis zu *N* Rechnern zu verwenden. Mit einer Parallellizenz für *N* Parallelbenutzer dürfen *N* Benutzer die Software gleichzeitig ausführen. (Die Software darf auf 10N Rechnern installiert sein.) Mit einer Named User-Lizenz darf ein bestimmter Benutzer die Software auf bis zu 5 verschiedenen Rechnern verwenden. Um Ihre Software zu aktivieren, klicken Sie auf **Neue Lizenz hochladen** und geben Sie im daraufhin angezeigten Dialogfeld den Pfad zur Lizenzdatei ein und klicken Sie auf **OK**.

Anmerkung: Bei Mehrplatzlizenzen wird jeder Benutzer aufgefordert, seinen eigenen Namen einzugeben.

Ihre Lizenz-E-Mail und die verschiedenen Methoden, Ihr Altova-Produkt zu lizenzieren
Die Lizenz-E-Mail, die Sie von Altova erhalten, enthält Ihre Lizenzdatei im Anhang. Die

© 2016-2022 Altova GmbH

Lizenzdatei hat die Dateierweiterung **.altova_licenses**.

Sie haben folgende Möglichkeiten, Ihr Altova-Produkt zu aktivieren:

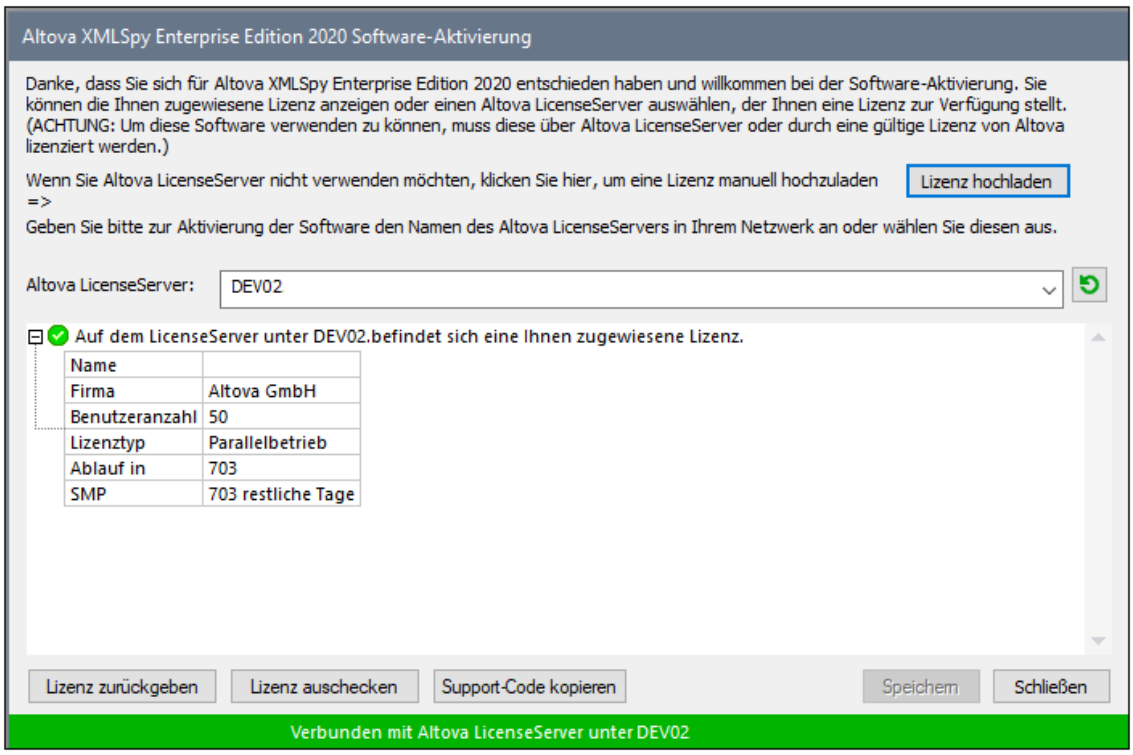
- Speichern Sie die Lizenzdatei (**.altova_licenses**) in einem geeigneten Ordner, doppelklicken Sie auf die Lizenzdatei, geben Sie etwaige erforderliche Informationen in das Dialogfeld ein, das daraufhin angezeigt wird und beenden Sie den Vorgang durch Klicken auf **Lizenzschlüssel anwenden**.
- Speichern Sie die Lizenzdatei (**.altova_licenses**) in einem geeigneten Ordner. Wählen Sie in Ihrem Altova-Produkt den Menübefehl **Hilfe | Software-Aktivierung** und klicken Sie anschließend auf **Neue Lizenz hochladen.** Navigieren Sie zur Lizenzdatei oder geben Sie den Pfad dazu ein und klicken Sie auf **OK**.
- Speichern Sie die Lizenzdatei (**.altova_licenses**) in einem geeigneten Ordner und laden Sie diese von dort aus in den Lizenz-Pool Ihres Altova LicenseServer hoch. Sie können die Lizenz anschließend (i) entweder von Ihrem Altova-Produkt über das Dialogfeld "Software-Aktivierung" abrufen (*siehe unten*) oder (ii) dem Produkt die Lizenz von Altova LicenseServer aus zuweisen. *Nähere Informationen zur Lizenzierung über LicenseServer finden Sie weiter unten in diesem Kapitel.*

Das Dialogfeld **Software-Aktivierung** (*Abbildung unten*) kann über den Befehl **Hilfe | Software-Aktivierung** aufgerufen werden.

Aktivieren Ihrer Software
Sie können die Software durch Registrieren der Lizenz im Dialogfeld "Software-Aktivierung" oder durch Lizenzierung über Altova LicenseServer (*nähere Informationen siehe unten*) aktivieren.

- *Registrierung der Lizenz im Dialogfeld "Software-Aktivierung".* Klicken Sie im Dialogfeld auf **Neue Lizenz hochladen** und navigieren Sie zur Lizenzdatei. Klicken Sie auf **OK,** um den Pfad zur Lizenzdatei und alle eingegebenen Daten (im Fall einer Mehrplatzlizenz Ihren Namen) zu bestätigen und abschließend auf **Speichern**.

- *Lizenzierung über einen Altova LicenseServer in Ihrem Netzwerk:* Um eine Lizenz über einen Altova LicenseServer in Ihrem Netzwerk abzurufen, (klicken Sie am unteren Rand des Dialogfelds **Software-Aktivierung** auf **Altova LicenseServer verwenden**). Wählen Sie den Rechner aus, auf dem der gewünschte LicenseServer installiert wurde. Beachten Sie, dass die automatische Ermittlung von License Servern durch die Aussendung eines Signals ins LAN erfolgt. Da diese Aussendung auf ein Subnetz beschränkt ist, muss sich der LicenseServer im selben Subnetz wie der Client-Rechner befinden, damit die Ermittlung von License Servern funktioniert. Falls die automatische Ermittlung nicht funktioniert, geben Sie den Namen des Servers ein. Der Altova LicenseServer muss in seinem Lizenzpool eine Lizenz für Ihre Altova-Produkt haben. Wenn im LicenseServer-Pool eine Lizenz verfügbar ist, wird dies im Dialogfeld **Software-Aktivierung** angezeigt (*siehe Abbildung unten, in der Sie das Dialogfeld in Altova XMLSpy sehen*) und Sie können auf **Speichern** klicken, um die Lizenz abzurufen.



Eine rechnerspezifische Lizenz (Einzelplatzlizenz) kann erst nach Ablauf von sieben Tagen wieder an LicenseServer zurückgegeben werden. Danach können Sie die rechnerspezifische Lizenz durch Klick auf **Lizenz zurückgeben** an den Server zurückgeben, sodass sie von einem anderen Client vom LicenseServer abgerufen werden kann. Ein LicenseServer-Administrator kann die Zuweisung einer abgerufenen Lizenz jedoch über die Web-Benutzeroberfläche von LicenseServer jederzeit aufheben. Beachten Sie, dass eine Rückgabe von Lizenzen nur bei rechnerspezifischen Lizenzen, nicht aber bei Parallellizenzen möglich ist.

Lizenz-Check-Out
Über den Lizenzpool können Sie eine Lizenz für einen Zeitraum von bis zu 30 Tagen auschecken, sodass die Lizenz auf dem lokalen Rechner gespeichert wird. Dadurch können Sie offline arbeiten, was nützlich ist, wenn Sie z.B. in einer Umgebung arbeiten möchten, in der Sie keinen Zugriff auf Ihren Altova LicenseServer haben (z.B. wenn Ihr Altova-Produkt auf einem Laptop installiert ist und Sie gerade unterwegs sind). Solange die Lizenz ausgecheckt ist, zeigt LicenseServer die Lizenz als in Verwendung an. Diese Lizenz kann dann von keinem anderen Rechner verwendet werden. Die Lizenz wird nach Ablauf des Check-Out-Zeitraums automatisch wieder eingecheckt. Alternativ dazu kann eine ausgecheckte Lizenz jederzeit über die Schaltfläche **Einchecken** des Dialogfelds **Software-Aktivierung** wieder eingecheckt werden.

Um eine Lizenz auszuchecken, gehen Sie folgendermaßen vor: (i) Klicken Sie im Dialogfeld **Software-Aktivierung** auf **Lizenz auschecken** (*siehe Abbildung oben*); (ii) Wählen Sie im daraufhin angezeigten Dialogfeld **Lizenz-Check-Out** den gewünschten Check-Out-Zeitraum aus und klicken Sie auf **Auschecken**. Daraufhin wird die Lizenz ausgecheckt. Nachdem Sie eine Lizenz ausgecheckt haben, geschehen zwei Dinge: (i) Die Check-Out-Informationen und das Ende des Check-Out-Zeitraums werden im Dialogfeld **Software-Aktivierung** angezeigt; (ii) Die Schaltfläche **Lizenz auschecken** im Dialogfeld ändert sich nun in **Einchecken**. Sie können die Lizenz jederzeit durch Klicken auf **Einchecken** einchecken. Da die Lizenz nach Ablauf des

 © 2016-2022 Altova GmbH

Check-Out-Zeitraums automatisch wieder in den Zustand "Eingecheckt" zurück wechselt, sollte der von Ihnen ausgewählte Zeitraum für das Check-Out den gewünschten Zeitraum, in dem Sie offline arbeiten möchten, entsprechend abdecken.

Wenn eine Lizenz wieder eingecheckt wird, muss diese Lizenz für dieselbe Hauptversion eines Altova-Produkts ausgestellt sein, wie die Lizenz, die ausgecheckt wurde. Stellen Sie daher sicher, dass die Lizenz eingecheckt ist, bevor Sie für Ihr Altova-Produkt ein Upgrade auf die nächste Hauptversion installieren.

Anmerkung: Damit Lizenzen ausgecheckt werden können, muss die Check-Out-Funktion auf dem LicenseServer aktiviert werden. Wenn diese Funktion nicht aktiviert wurde, erhalten Sie eine entsprechende Fehlermeldung, wenn Sie versuchen die Lizenz auszuchecken. Wenden Sie sich in diesem Fall an Ihren LicenseServer-Administrator.

Support-Code kopieren
Klicken Sie auf **Support-Code kopieren,** um Lizenzinformationen in die Zwischenablage zu kopieren. Dies sind die Daten, die Sie bei einer Support-Anfrage über das Online Support-Formular benötigen.

Altova LicenseServer bietet IT-Administratoren einen Echtzeitüberblick über alle Altova-Lizenzen in einem Netzwerk. Dazu werden die Einzelheiten zu jeder Lizenz sowie Client-Zuweisungen und die Verwendung von Lizenzen durch Clients angezeigt. Der Vorteil der Verwendung von LicenseServer liegt in seinen Funktionen zur Verwaltung großer Altova-Lizenzpools. Altova LicenseServer steht kostenlos auf der Altova Website zur Verfügung. Nähere Informationen zu Altova LicenseServer und der Lizenzierung mittels Altova LicenseServer finden Sie in der Dokumentation zu Altova LicenseServer.

- Bestellformular

  Sobald Sie eine lizenzierte Version des Software-Produkts bestellen möchten, klicken Sie im Dialogfeld **Software-Aktivierung** (*siehe oben*) auf die Schaltfläche **Permanenten Key-Code erwerben...** oder wählen Sie den Befehl **Bestellformular**, um zum sicheren Online-Shop von Altova weitergeleitet zu werden.

- Registrierung

  Bei Aufruf dieses Befehls wird die Altova-Produktregistrierungsseite auf einem Register Ihres Browsers geöffnet. Durch Registrierung Ihrer Altova-Software stellen Sie sicher, dass Sie immer die neuesten Produktinformationen erhalten.

- Auf Updates überprüfen

  Überprüft, ob am Altova Server eine neuere Version Ihres Produkts vorhanden ist und zeigt eine entsprechende Meldung an.

28.19.4 Weitere Befehle

- Support Center

  Der Befehl "Support Center" ist ein Link zum Altova Support Center im Internet. Im Support Center finden Sie Antworten auf häufig gestellt Fragen, Diskussionsforen, in denen Sie Software-Probleme besprechen

können und ein Formular, um unsere Mitarbeiter vom technischen Support zu kontaktieren.

- Fragen und Antworten im Web

  Der Befehl "Fragen und Antworten im Web" ist ein Link zur Altova-Fragen- und Antworten-Datenbank im Internet. Diese Datenbank wird ständig aktualisiert, wenn die Mitarbeiter von Altova von unseren Kunden mit neuen Fragen konfrontiert werden.

- Komponenten und Gratistools downloaden

  Dieser Befehl ist ein Link zum Komponenten Download Center von Altova im Internet. Von hier können Sie Software-Komponenten verschiedener anderer Anbieter herunterladen, die Sie mit Altova Produkten verwenden können. Dabei handelt es sich um XSLT- und XSL-FO-Prozessoren, Applikationsserverplattformen usw. Die im Komponenten Download Center verfügbare Software ist normalerweise kostenlos.

- XMLSpy im Internet

  Der Befehl XMLSpy im Internet ist ein Link zur Altova Website im Internet. Hier erfahren Sie mehr über XMLSpy und verwandte Technologien und Produkte auf der Altova Website.

- XMLSpy Training

  Ein Link zur Online Training-Seite der Altova Website. Hier können Sie zwischen den von Altova-Experten gehaltenen Online-Kursen wählen.

- Über XMLSpy

  Mit dem Befehl Über XMLSpy wird das Willkommensfenster und die Versionsnummer Ihres Produkts angezeigt. Wenn Sie die 64-Bit-Verson von XMLSpy verwenden, wird dies durch das Suffix (x64) nach dem Applikationsnamen angezeigt. Die 32-Bit-Version hat kein Suffix.

© 2016-2022 Altova GmbH

28.20 Befehlszeile

Bestimmte XMLSpy-Aktionen können über die Befehlszeile ausgeführt werden. Diese Befehle sind im Folgenden aufgelistet:

Datei öffnen

xmlspy.exe `file.xml`

Öffnet die Datei `file.xml`, in XMLSpy

Mehrere Dateien öffnen

xmlspy.exe `file1.xml file2.xml`

Öffnet die Dateien `file1.xml` und `file2.xml` in XMLSpy.

Einer XML-Datei eine SPS-Datei für die Bearbeitung in der Authentic-Ansicht zuweisen

xmlspy.exe `myxml.xml /sps mysps.sps`

Öffnet die Datei `myxml.xml` in der Authentic-Ansicht mit `mysps.sps` als ihrer SPS-Datei. Das Flag `/sps` gibt an, dass (für die Bearbeitung in der Authentic-Ansicht) die nachfolgende SPS-Datei mit der XML-Datei zu verwenden ist, die vor dem `/sps` Flag steht.

Neue XML-Vorlagendatei über eine SPS-Datei öffnen

xmlspy.exe `mysps.sps`

Öffnet eine neue XML-Datei in der Authentic-Ansicht. Die Anzeige basiert auf dem SPS und die neue XML-Datei hat eine Skelettstruktur, die auf dem SPS-Schema basiert. Der Name der neu erstellten XML-Datei muss der XML-Datei beim Speichern zugewiesen werden.

SPS-Datei in der Textansicht als XML-Dokument öffnen

xmlspy.exe `/raw mysps.sps`

Öffnet die Datei `mysps.sps` als XML-Dokument in der Textansicht. Das Flag `/raw` gibt an, dass die nachfolgende SPS-Datei als XML-Datei zu bearbeiten ist.

29 Referenz für Programmierer

XMLSpy ist ein Automation Server, d.h. eine Applikation, die anderen Applikationen, so genannten Automation Clients, programmierbare Objekte zur Verfügung stellt. Ein Automation Client hat direkten Zugriff auf die vom Automation Server bereitgestellten Objekte und Funktionalitäten, sodass sich der Automation Client die Funktionalitäten von XMLSpy zunutze machen kann. Dadurch können Sie von anderen Applikationen aus die fertigen Funktionen von XMLSpy nutzen.

Die programmierbaren Objekte von XMLSpy stehen den Automation Clients über die Applikations-API von XMLSpy, eine COM API, zur Verfügung. Die Applikations-API XMLSpy wird in dieser Dokumentation in der Folge auch als Applikations-API bezeichnet. Das Objektmodell der API und eine vollständige Beschreibung aller verfügbaren Objekte finden Sie in dieser Dokumentation (siehe Abschnitt Applikations-API[1706]).

Ausführungsumgebungen

Die Applikations-API kann von den folgenden Umgebungen aus aufgerufen werden:

- Skript-Editor[1662]
- IDE Plug-ins[1690]
- Externe Programme[1706]
- ActiveX Integration[2073]

Im Folgenden finden Sie eine Beschreibung der einzelnen Umgebungen.

Skript-Editor: Anpassen von XMLSpy und Ändern der Funktionalität
Sie können Ihre Installation von XMLSpy anpassen, indem Sie sie ändern und Funktionalitäten dazu hinzufügen. Sie können auch Formulare für die Benutzereingabe erstellen und neue Menübefehle und Symbolleisten-Schaltflächen zur Benutzeroberfläche hinzufügen. Zu diesem Zweck werden Skripts geschrieben, die mit Objekten der Applikations-API interagieren. Zur effizienten Ausführung dieser Aufgaben steht in XMLSpy ein integrierter Skript-Editor zur Verfügung. Eine ausführliche Beschreibung der im Skript-Editor verfügbaren Funktionalitäten und eine Anleitung zur Verwendung der Scripting-Umgebung finden Sie im Abschnitt Skript-Editor[1662] dieser Dokumentation. Unterstützt werden die Programmiersprachen **JScript** und **VBScript**.

IDE Plug-ins: Erstellen von Plug-ins für XMLSpy
Sie haben in XMLSpy die Möglichkeit Ihre eigenen Plug-Ins zu schreiben und in XMLSpy zu integrieren. Dazu können Sie die spezielle XMLSpy Schnittstelle für Plug-ins verwenden. Eine Beschreibung zum Erstellen von Plug-ins finden Sie im Abschnitt XMLSpy Plug-ins[1690].

An die meisten Methoden, die von einem IDE Plug-in implementiert werden müssen und das von der Applikation aufgerufen wird, wird ein Applikationsobjekt übergeben. Häufig werden zur Implementierung eines IDE Plug-in die Programmiersprachen **C#** und **C++** verwendet. Nähere Informationen dazu finden Sie im Abschnitt XMLSpy IDE Plugins[1690].

Externe Programme

Außerdem können Sie XMLSpy mittels externer Skripts bedienen. So könnten Sie z.B. ein Skript schreiben, um XMLSpy zu einem bestimmten Zeitpunkt zu öffnen, dann eine XML-Datei in XMLSpy zu öffnen, die Datei zu validieren und auszudrucken. Externe Skripts würden sich zur Ausführung dieser Aufgaben wiederum der

© 2016-2022 Altova GmbH

Applikations-API bedienen. Eine Beschreibung der Applikations-API finden Sie im Abschnitt Applikations-API[1706].

Um die Applikations-API außerhalb von XMLSpy verwenden zu können, muss zuerst eine Instanz von XMLSpy gestartet werden. Wie dies erfolgt, hängt von der verwendeten Programmiersprache ab. Nähere Informationen zu den einzelnen Sprachen finden Sie unter Programmiersprachen[1708].

Im Wesentlichen wird XMLSpy über seine COM Registrierung gestartet. Anschließend wird das mit der XMLSpy-Instanz verknüpfte`Application` Objekt zurückgegeben. Je nach COM-Einstellungen kann ein mit einer bereits laufenden XMLSpy-Instanz verknüpftes Objekt zurückgegeben werden. Es kann jede Programmiersprache verwendet werden, die die Erstellung und den Aufruf von COM-Objekten unterstützt. Die am häufigsten dazu verwendeten Sprachen sind unten aufgelistet.

- JScript[1732] und VBScript[1719] Skript-Dateien haben eine einfache Syntax und dienen dazu COM-Objekte aufzurufen. Sie können direkt über eine DOS-Befehlszeile oder durch Doppelklick im Windows Explorer ausgeführt werden. Am besten eignen sich diese Sprachen für einfache Automationsaufgaben.
- C#[1722] ist eine umfangreiche Programmiersprache mit einer breiten Palette an Funktionalitäten. Der Zugriff auf COM-Objekte kann in C# automatisch über Wrapper erfolgen.
- C++ ermöglicht die direkte Steuerung des COM-Zugriffs. Allerdings muss dafür im Vergleich zu anderen Sprachen relativ viel Code geschrieben werden.
- Java[1732]: Im Lieferumfang von Altova-Produkten sind native Java-Klassen inkludiert, die als Wrapper für die Applikations-API verwendet werden und dadurch eine Java-Umgebung ermöglichen.
- Weitere nützliche Alternativen sind Visual Basic für Applikationen, Perl und Python.

ActiveX Integration

Ein Sonderfall ist der Aufruf der Applikations-API über das XMLSpy ActiveX Control. Diese Funktion steht nur zur Verfügung, wenn das XMLSpy Integration Package[2073] installiert ist. Jedes ActiveX Control hat eine Eigenschaft, die das entsprechende COM-Objekt für die zugrunde liegende Funktion zurückgibt. Das Manager Control liefert ein `Application` Object, das Document Control ein `Document` Object und das Platzhalterobjekt gibt, wenn es die Projektstruktur enthält, das `Project` Objekt zurück. Die von diesen Objekten unterstützten Methoden sind genau diegleichen, wie im Abschnitt "Schnittstellen" des Abschnitts Applikations-API[1745] beschrieben. Achten Sie darauf, keine Methoden zu verwenden, die keinen Sinn im Zusammenhang mit der ActiveX Control Integration ergeben. Nähere Informationen dazu finden Sie unter ActiveX Integration[2073].

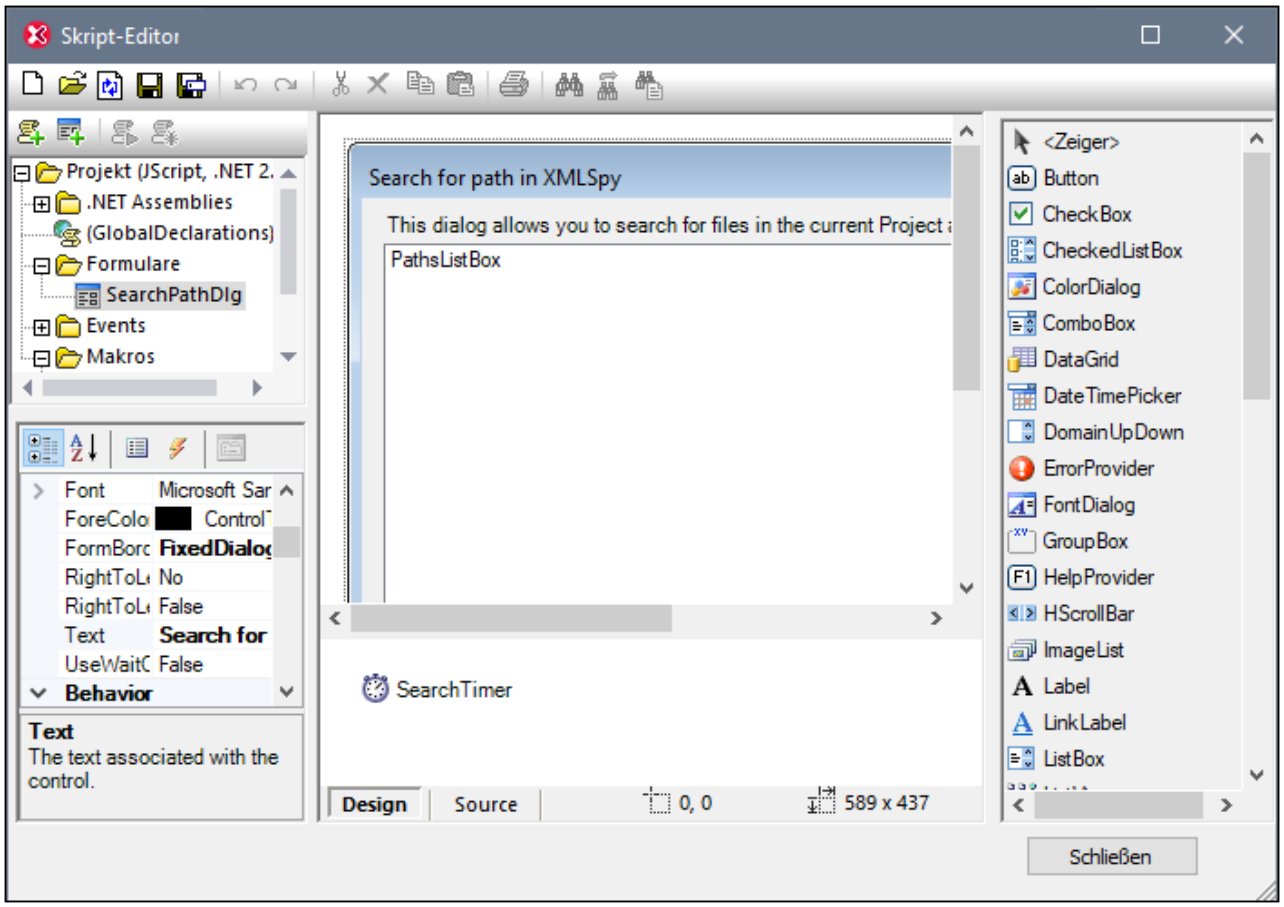
Informationen über die Referenz für Programmierer

Die XMLSpy Referenz für Programmierer besteht aus den folgenden Abschnitten:

- Skript-Editor[1662]: eine Benutzerreferenz für die Skripting-Umgebung in XMLSpy
- IDE Plug-ins[1690]: eine Anleitung zum Erstellen von Plug-ins für XMLSpy
- Applikations-API[1706]: eine Referenz für die Applikations-API
- ActiveX Integration[2073]: Anleitung und Referenz zur Integration der Benutzeroberfläche und Funktionalitäten von XMLSpy mit Hilfe eines ActiveX Control.

29.1 Skript-Editor

Der Skript-Editor ist eine in XMLSpy integrierte Entwicklungsumgebung, über die Sie die Funktionalitäten von XMLSpy mit Hilfe von JScript- oder VBScript-Skripts anpassen können. So können Sie z.B. einen neuen Menüeintrag zur Durchführung einer benutzerdefinierten Projektaufgabe hinzufügen oder Sie können festlegen, dass jedes Mal, wenn ein Dokument in XMLSpy geöffnet oder geschlossen wird, ein bestimmtes Verhalten ausgelöst wird. Zu diesem Zweck werden Skripting-Projekte - Dateien mit der Erweiterung .asprj (Altova Skripting-Projekt) - erstellt.



Skript-Editor

Skripting-Projekte enthalten normalerweise ein oder mehrere Makros, d.h. Programme, die bei Aufruf verschiedene benutzerdefinierte Aufgaben ausführen. Makros können entweder explizit über einen Menübefehl (oder ggf. eine Symbolleisten-Schaltfläche) gestartet werden oder so konfiguriert werden, dass sie beim Start von XMLSpy automatisch ausgeführt werden. Die Skripting-Umgebung ist auch mit der XMLSpy COM API integriert. So können Ihre VBScript- oder JScript-Skripts etwa Applikations- oder Dokument-Ereignisse wie das Starten oder Beenden von XMLSpy, das Öffnen oder Schließen eines Projekts, usw. behandeln. Skripting-Projekte können Windows-Formulare enthalten, die Sie ähnlich wie in Visual Studio visuell entwerfen können. Zusätzlich dazu steht eine Reihe von vordefinierten Befehlen, die Ihnen dabei helfen, .NET-Klassen aus VBScript oder JScript-Code zu instantiieren und zu verwenden zur Verfügung.

© 2016-2022 Altova GmbH

Sobald Ihr Scripting-Projekt fertig gestellt wurde, können Sie es in XMLSpy entweder global oder nur für bestimmte Projekte aktivieren.

Für den Skript-Editor muss vor der Installation von XMLSpy .NET Framework 2.0 oder höher installiert sein.

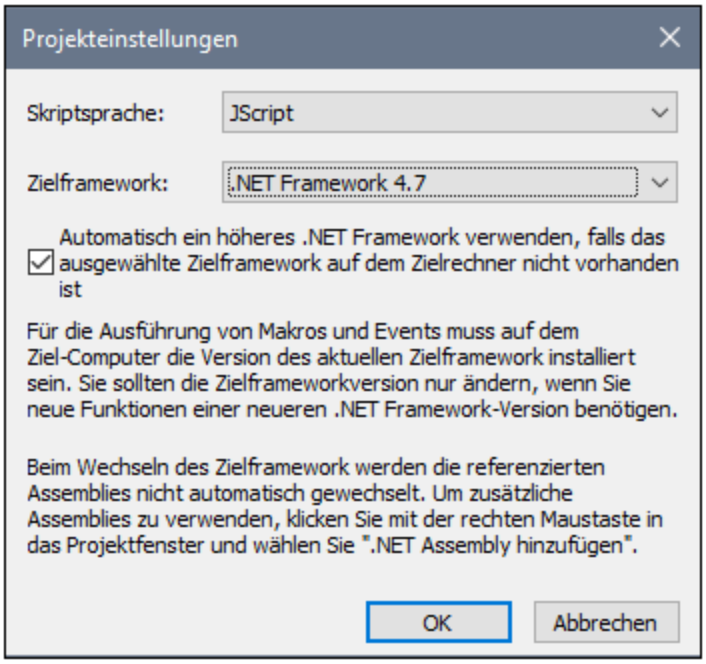
29.1.1 Erstellen eines Skripting-Projekts

Alle im Skript-Editor erstellten Skripting-Informationen werden in Altova Skripting-Projekten (.asprj-Dateien) gespeichert. Ein Skripting-Projekt kann Makros, Applikation Event Handler und Formulare (die wiederum ihre eigenen Event Handler haben können) enthalten. Zusätzlich dazu können Sie globale Variablen und Funktionen zu einem "Global Declarations"-Skript hinzufügen, sodass diese Variablen und Funktionen im gesamten Projekt aufgerufen werden können.

Um ein neues Projekt zu beginnen, klicken Sie auf den Menübefehl **Extras | Skript-Editor**.

Für die Verwendung in einem Skripting-Projekt werden die Sprachen JScript und VBScript unterstützt (nicht zu verwechseln mit Visual Basic, welches nicht unterstützt wird). Diese Skripting-Prozessoren stehen standardmäßig in Windows zur Verfügung. Es müssen keine speziellen Voraussetzungen erfüllt werden, um diese Sprachen auszuführen. Folgendermaßen können Sie eine Skripting-Sprache auswählen:

1. Klicken Sie mit der rechten Maustaste im linken oberen Bereich auf den Eintrag **Projekt** und wählen Sie im Kontextmenü den Befehl **Projekteinstellungen** aus.
2. Wählen Sie eine Sprache (JScript oder VBScript) aus und klicken Sie auf **OK**.



Über das oben gezeigte Dialogfeld "Projekteinstellungen" können Sie auch die .NET-Zielframework-Version ändern. Normalerweise ist dies dann notwendig, wenn für Ihr Skripting-Projekt Funktionen benötigt werden, die in einer neueren .NET Framework-Version zur Verfügung stehen. Beachten Sie, dass dieselbe .NET

Framework-Version (oder eine höhere kompatible Version) auch auf allen Clients, auf denen Ihr Skripting-Projekt verwendet wird, installiert sein muss.

Ein Skripting-Projekt referenziert standardmäßig mehrere .NET Assemblys wie `System`, `System.Data`, `System.Windows.Forms` und andere. Falls nötig, können Sie weitere .NET Assemblys, darunter Assemblys aus .NET Global Assembly Cache (GAC) oder benutzerdefinierte .dll-Dateien importieren. Assemblys können folgendermaßen importiert werden:

1. Statisch durch manuelles Hinzufügen zum Projekt. Klicken Sie mit der rechten Maustaste im linken oberen Bereich auf **Projekt** und wählen Sie im Kontextmenü den Befehl **.NET Assembly hinzufügen**.
2. Dynamisch zur Laufzeit durch Aufruf des Befehls `CLR.LoadAssembly`[1680] über den Code.

Gegebenenfalls können Sie mehrere Skripting-Projekte erstellen. Sie können ein Skripting-Projekt auf der Festplatte speichern und es später wieder über die Windows-Standard-Symbolleisten-Schaltflächen **Neu, Öffnen, Speichern, Speichern unter** in den Skript-Editor laden. Nachdem Sie das Skripting-Projekt getestet haben und es bereitgestellt werden kann, können Sie es in XMLSpy laden und jedes beliebige seiner Makros und Event Handler ausführen. Nähere Informationen dazu finden Sie unter Aktivieren von Skripts und Makros[1687] .

Ein Beispiel für ein Skripting-Projekt finden Sie unter dem folgenden Pfad: **C:\Benutzer\<Benutzer>\Dokumente\Altova\XMLSpy2023\Examples\SampleScripts.asprj**.

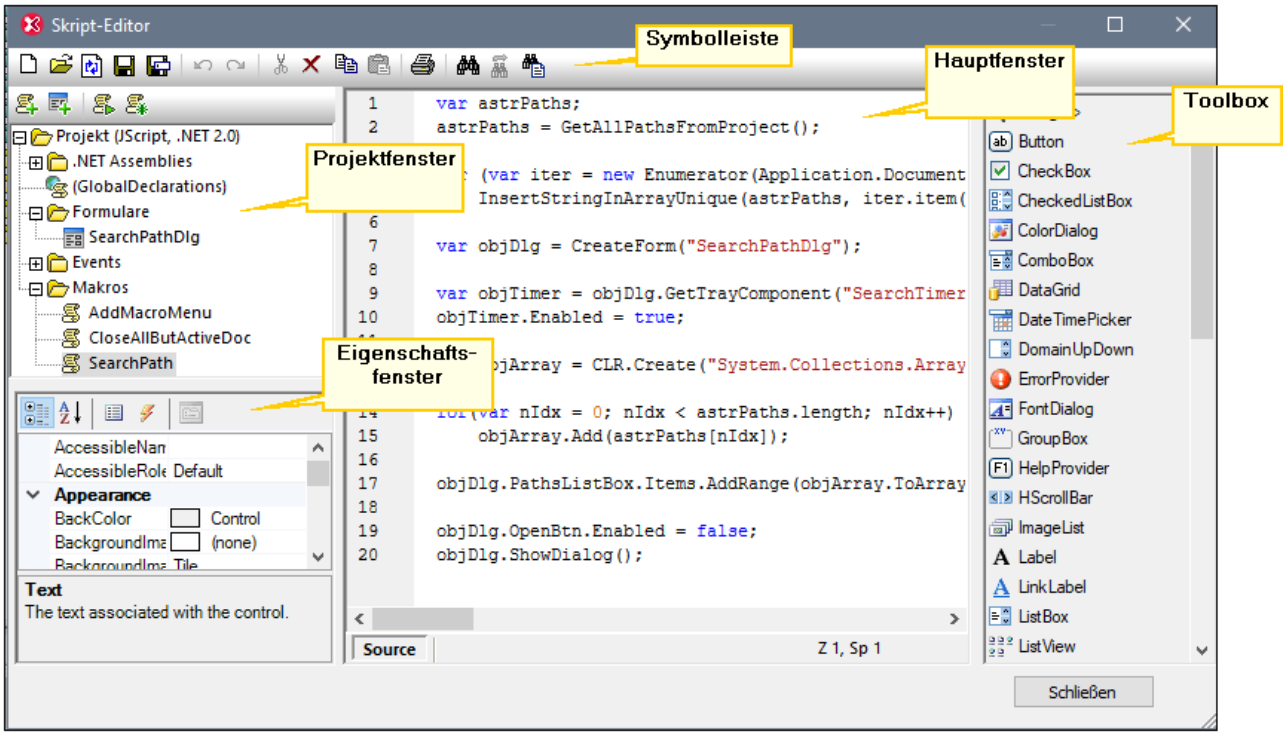
In den nächsten Abschnitten wird beschrieben, welche Teile in Ihrem Skripting-Projekt eventuell benötigt werden: Globale Deklarationen, Makros, Formulare und Events.

29.1.1.1 Übersicht über die Umgebung

Der Skript-Editor besteht aus den folgenden Teilen:

- Symbolleiste
- Projektfenster
- Eigenschaftsfenster
- Hauptfenster
- Toolbox

© 2016-2022 Altova GmbH



Symbolleiste

Die Symbolleiste enthält Windows-Standardbefehle für die Dateiverwaltung (**Neu**, **Öffnen**, **Speichern**, **Speichern unter**) sowie Bearbeitungsbefehle (**Kopieren, Ausschneiden, Löschen, Einfügen**). Bei der Bearbeitung von Quellcode stehen zusätzlich die Befehle **Suchen** und **Ersetzen** sowie **Drucken** zur Verfügung.

Projektfenster

Im Projektfenster können Sie die Struktur des Projekts anzeigen und verwalten. Ein Skripting-Projekt besteht aus mehreren Komponenten, die gemeinsam verwendet und in jeder beliebigen Reihenfolge erstellt werden können.

- Ein Skript *"Global Declarations"*. Wie der Name schon sagt, sind in diesem Skript Informationen gespeichert, die global im gesamten Projekt zur Verfügung stehen. Sie können in diesem Skript alle Variablen oder Funktionen, die in alle Formularen, Event Handlern und Makros zur Verfügung stehen sollen, deklarieren.
- *Formulare.* Formulare dienen normalerweise zum Erfassen von Benutzer-Eingaben oder stellen Informationsdialogfelder zur Verfügung. Ein Formular wird entweder durch einen Aufruf innerhalb einer Funktion (im Globale Deklaration-Skript) oder durch einen Aufruf direkt in einem Makro aufgerufen.
- *Events.* In Ordner "Events" werden XMLSpy Applikations-Events aus der COM API angezeigt. Um ein Skript zu schreiben, das bei Auftreten eines Events ausgeführt werden soll, doppelklicken Sie auf ein beliebiges Event und geben Sie anschließend den behandelnden Code in den Editor ein. Die Applikations-Events sind nicht mit Formular-Events zu verwechseln; letztere werden auf Formularebene behandelt, wie weiter unten beschrieben.
- *Makros*. Ein Makro ist ein Skript, das entweder bei Bedarf über ein Kontextmenü aufgerufen werden kann oder automatisch beim Start von XMLSpy ausgeführt wird. Makros haben keine Parameter oder

Rückgabewerte. Ein Makro kann alle im Global Declarations-Skript deklarierten Variablen und Funktionen aufrufen und auch Formulare anzeigen.

Klicken Sie mit der rechten Maustaste auf eine der Komponenten, um die verfügbaren Kontextmenübefehle und deren Tastenkürzel zu sehen. Doppelklicken Sie auf eine beliebige Datei (wie z.B. ein Formular oder ein Skript), um diese(s) im Hauptfenster zu öffnen.

Über die Symbolleisten-Schaltflächen stehen die folgenden Schnellbefehle zur Verfügung:

	**Neues Makro**	Fügt im Verzeichnis "Makros" ein neues Makro zum Projekt hinzu.
	**Neues Formular**	Fügt im Verzeichnis "Formulare" ein neues Formular zum Projekt hinzu.
	**Makro ausführen**	Führt das ausgewählte Makro aus.
	**Makro debuggen**	Führt das ausgewählte Makro im Debug-Modus aus.

Eigenschaftsfenster

Das Eigenschaftsfenster ist dem Eigenschaftsfenster in Visual Studio sehr ähnlich. Darin werden die folgenden Elemente angezeigt:

- Wenn ein Formular ausgewählt ist, Formulareigenschaften
- Wenn ein Objekt in einem Formular ausgewählt ist, Objekteigenschaften
- Wenn ein Formular ausgewählt ist, Formular-Events
- Wenn ein Objekt in einem Formular ausgewählt ist, Objekt-Events

Um zwischen den Eigenschaften und Events der ausgewählten Komponente hin- und herzuwechseln, klicken Sie auf die Schaltfläche **Properties**  bzw. auf die Schaltfläche **Events**  .

Mit den Schaltflächen **Categorized**  und **Alphabetical**  werden die Eigenschaften oder Events, entweder nach Kategorie oder in aufsteigender alphabetischer Reihenfolge geordnet.

Wird eine Eigenschaft oder ein Event ausgewählt, wird am unteren Rand des Eigenschaftsfensters eine kurze Beschreibung angezeigt.

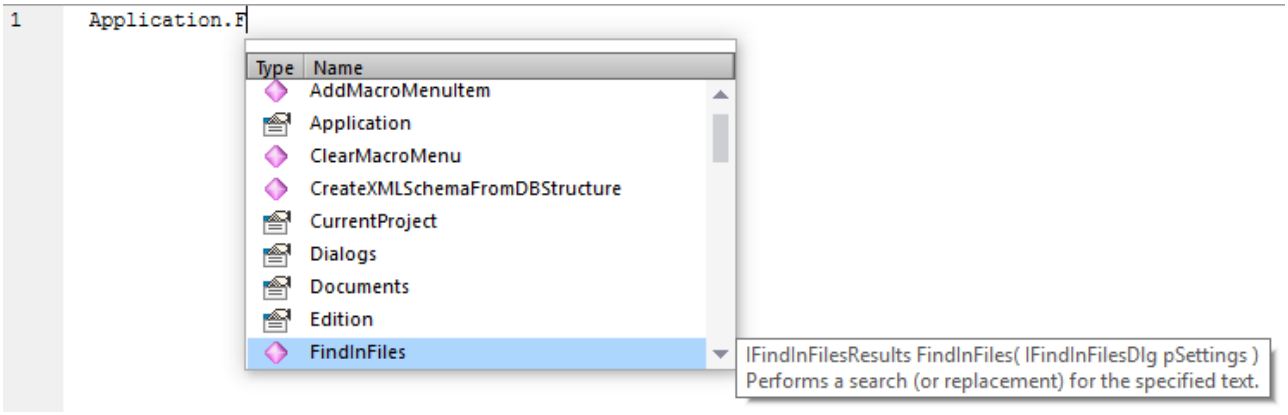
Hauptfenster

Das Hauptfenster ist der Arbeitsbereich, in den Sie Quellcode eingeben oder das Design des Formulars bearbeiten können. Bei der Bearbeitung von Formularen können Sie auf zwei Registern arbeiten: dem Register **Design** und dem Register **Source**. Auf dem Register **Design** sehen Sie das Layout des Formulars, während das Register **Source** den Quellcode wie z.B. Handler-Methoden für die Formular-Events enthält.

Der Quellcode-Editor bietet Codebearbeitungshillfen wie Syntaxfärbung, Quellcode-Klappleiste, Markierung von öffnenden und schließenden geschwungenen Klammern, Zoomen, Autokomplettierungsvorschläge, Lesezeichen.

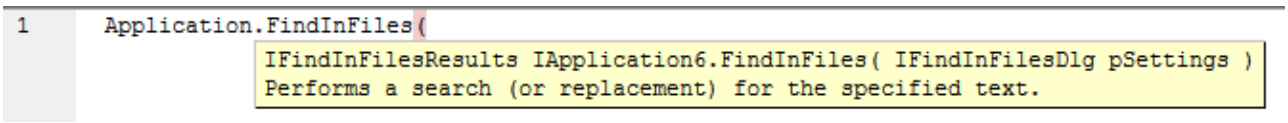
© 2016-2022 Altova GmbH

Autokomplettierungsvorschläge
JScript und VBScript sind typenlose Sprachen, daher ist die Autokomplettierung auf COM API-Namen und vordefinierte XMLSpy-Befehle[1677] beschränkt. Die vollständige Methode oder Eigenschaftssignatur wird neben der Autokomplettierungseingabehilfe angezeigt.



Wenn Namen mit `objDocument`, `objProject`, `objXMLData`, oder `objAuthenticRange` beginnen, werden Mitglieder der entsprechenden Schnittstelle angezeigt.

Wenn Sie den Mauszeiger über eine bekannte Methode oder Eigenschaft platzieren, wird deren Signatur (und, falls vorhanden, Dokumentation) angezeigt, z.B:



Die Autokomplettierungseingabehilfe wird normalerweise automatisch bei der Bearbeitung angezeigt, kann aber auch jederzeit durch Drücken von **Strg+Leerzeichen** angezeigt werden.

Lesezeichen
- Um ein Lesezeichen zu setzen oder zu entfernen, klicken Sie in eine Zeile und drücken Sie **Strg+F2.**
- Um zum nächsten Lesezeichen zu gelangen, drücken Sie **F2.**
- Um zum vorhergehenden Lesezeichen zu gelangen, drücken Sie **Umschalt+F2.**
- Um alle Lesezeichen zu löschen, drücken Sie **Strg+Umschalt+F2.**

Vergrößern/Verkleinern
- Um etwas zu vergrößern oder zu verkleinern, halten Sie die Strg-Taste gedrückt und drücken Sie anschließend die "+" oder "-" Taste oder drehen Sie das Mausrad.

Einstellungen für Textansicht
Um die Texteinstellungen aufzurufen, klicken Sie mit der rechten Maustaste in den Editor und wählen Sie im Kontextmenü den Befehl **Einstellungen für Textansicht.**

Schriftarteinstellungen
Um die Schriftart zu ändern, klicken Sie mit der rechten Maustaste in den Editor und wählen Sie im Kontextmenü den Befehl **Schriftart Textansicht**.

Toolbox

Die Toolbox enthält alle Objekte, die für das Formulardesign zur Verfügung stehen, wie z.B. Schaltflächen, Textfelder, Auswahllisten, usw.

So fügen Sie ein Element aus der Toolbox zu einem Formular hinzu:

1. Erstellen oder öffnen Sie ein Formular und stellen Sie sicher, dass das Register **Design** ausgewählt ist.
2. Klicken Sie auf das Toolbox-Objekt (z.B.**Button**) und anschließend auf die Stelle im Formular, an der Sie es einfügen möchten. Alternativ dazu können Sie das Objekt auch mit der Maus direkt in das Formular ziehen.

Einige Objekte wie z.B. `Timer` werden nicht zum Formular hinzugefügt, sondern in der Ablageunteren Bereich des Hauptfensters angezeigt. Sie können das Objekt dort auswählen und über das Eigenschaftsfenster dafür Eigenschaften und Event Handler definieren. Ein Beispiel für die Behandlung solcher Komponenten über den Code finden Sie unter Behandeln von Formular-Events 1670.

Sie können auch registrierte ActiveX Controls zum Formular hinzufügen. Klicken Sie dazu mit der rechten Maustaste in den Toolbox-Bereich und wählen Sie im Kontextmenü den Befehl **ActiveX Control hinzufügen.**

29.1.1.2 Globale Deklarationen

Das Skript "Global Declarations" ist standardmäßig in jedem Skripting-Projekt vorhanden und muss nicht explizit erstellt werden. Alle Variablen oder Funktionen, die Sie zu diesem Skript hinzufügen, werden als global für das gesamte Projekt behandelt, sodass Sie solche Variablen und Funktionen von jedem beliebigen Makro oder Event des Projekts aus referenzieren können. Im Folgenden sehen Sie ein Beispiel für ein Global Declarations-Skript, das den Namespace `System.Windows.Forms` in das Projekt importiert. Der unten stehende Code ruft zu diesem Zweck den im Skript-Editor vordefinierten Befehl `CLR.Import` auf.

```
// import System.Windows.Forms namespace for all macros, forms and events:
CLR.Import( "System.Windows.Forms" );
```

Anmerkung: Die globalen Deklarationen werden jedes Mal, wenn ein Makro ausgeführt oder ein Event Handler aufgerufen wird, erneut initialisiert.

29.1.1.3 Makros

Makros sind Skripts, die JScript-Anweisungen (oder, je nach Projektsprache, VBScript-Anweisungen) wie z.B. Variablendeklarationen und Funktionen enthalten.

Wenn in Ihren Projekten Makros verwendet werden sollen, können Sie diese folgendermaßen hinzufügen: Klicken Sie mit der rechten Maustaste in das Projektfenster, wählen Sie im Kontextmenü den Befehl Makro hinzufügen aus und geben Sie den Code des Makros in das Hauptformular ein. Beim Makrocode kann es sich z.B. um eine einfache Warnmeldung handeln:

```
alert("Hello, I'm a macro!");
```

 © 2016-2022 Altova GmbH

Komplexere Makros können Variablen und lokale Funktionen enthalten. Makros können auch Code, der Formulare aus dem Projekt aufruft, enthalten. Im Codefragment unten sehen Sie ein Beispiel für ein Makro, das ein Formular aufruft. Es wird davon ausgegangen, dass dieses Formular im Ordner "Formulare" bereits erstellt wurde und den Namen "SampleForm" hat, siehe auch Formulare[1669].

```
// display a form
ShowForm( "SampleForm" );
```

`ShowForm` im obigen Codefragment ist ein im Skript-Editor vordefinierter Befehl. Eine Liste anderer ähnlicher Befehle, die für die Arbeit mit Formularen und .NET-Objekten zur Verfügung stehen, finden Sie unter Vordefinierte Befehle[1677].

Sie können mehrere Makros zum selben Projekt hinzufügen und jedes beliebige Makro als "automatisches Makro" definieren. Wenn ein Makro als automatisches Makro definiert ist, wird es beim Start von XMLSpy automatisch ausgeführt. Um ein Makro als automatisches Makro zu definieren, klicken Sie mit der rechten Maustaste darauf und wählen Sie im Kontextmenü den Befehl **Als automatisches Makro definieren**.

Es kann immer nur ein Makro gleichzeitig ausgeführt werden. Nachdem ein Makro (oder Event) ausgeführt wurde, wird das Skript geschlossen und die globalen Variablen verlieren ihre Werte.

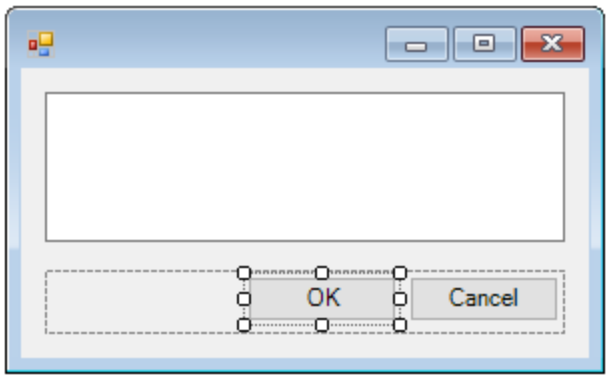
Um ein Makro direkt im Skript-Editor auszuführen, klicken Sie auf **Makro ausführen** . Um ein Makro mit dem Visual Studio Debugger zu debuggen, klicken Sie auf **Makro debuggen** . Informationen darüber, wie Sie in XMLSpy Makros aktivieren und ausführen, finden Sie unter Aktivieren von Skripts und Makros[1687].

29.1.1.4 Formulare

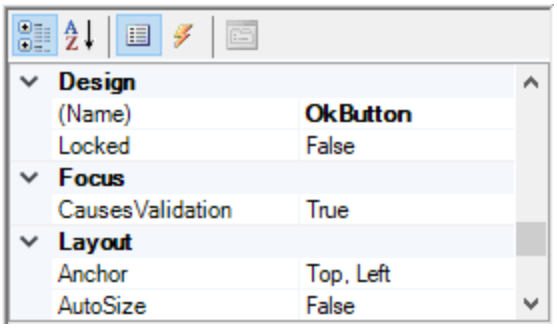
Formulare eigenen sich vor allem, um Eingabedaten von Benutzern zu erfassen oder Benutzern Daten anzuzeigen. Ein Formular kann diverse Steuerelemente wie z.B. Schaltflächen, Kontrollkästchen, Auswahllisten, usw. enthalten.

Um ein Formular hinzuzufügen, klicken Sie im Projektfenster mit der rechten Maustaste auf das Formular und wählen Sie im Kontextmenü den Befehl **Formular hinzufügen**. Um ein Steuerelement zu einem Formular hinzuzufügen, ziehen Sie es aus der Toolbox auf der rechten Seite des Skript-Editors in das Formular.

Sie können die Position und Größe der Steuerelemente direkt im Formular über die Ziehpunkte, die angezeigt werden, wenn Sie auf das Steuerelement klicken, ändern, z.B:



Alle Formularsteuerelemente haben Eigenschaften, die Sie im Eigenschaftsfenster mühelos anpassen können. Wählen Sie dazu das Steuerelement zuerst im Formular aus und bearbeiten Sie anschließend die gewünschten Eigenschaften im Eigenschaftsfenster.

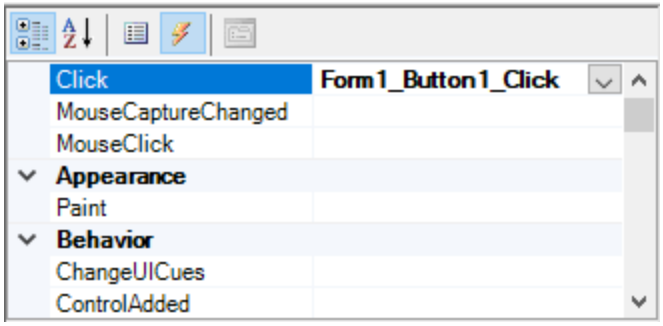


Behandeln von Formular-Events

Jedes Formularsteuerelement stellt außerdem verschiedene Events bereit, an die Ihr Skripting-Projekt gebunden werden kann. So können Sie z.B. jedes Mal, wenn ein Benutzer auf eine Schaltfläche klickt, eine bestimmte XMLSpy COM API-Methode aufrufen lassen. Um eine Funktion zu erstellen, die an ein Formular-Event gebunden ist, gehen Sie folgendermaßen vor:

1. Klicken Sie im Eigenschaftsfenster auf die Schaltfläche **Events** .
2. Doppelklicken Sie in der Spalte **Action** auf das Event, für das Sie die Methode benötigen (in der Abbildung unten ist dies "Click").

© 2016-2022 Altova GmbH



Sie können Handler-Methoden auch durch Doppelklick auf ein Steuerelement im Formular hinzufügen. Wenn Sie z.B. im Formulardesign auf eine Schaltfläche doppelklicken, wird eine Handler-Methode für das "Click"-Event dieser Schaltfläche erstellt.

Nachdem der Rumpf der Handler-Methode generiert wurde, können Sie Code eingeben, der dieses Event behandelt, z.B:

```
//Occurs when the component is clicked.
function MyForm_ButtonClick( objSender, e_EventArgs )
{
   alert("A button was clicked");
}
```

Um ein Formular, an dem Sie gerade arbeiten, getrennt vom Skript-Editor anzuzeigen, klicken Sie mit der rechten Maustaste im Projektfenster auf das Formular und wählen Sie im Kontextmenü den Befehl **Formular testen**. Beachten Sie, dass mit dem Befehl **Formular testen** nur das Formular angezeigt wird; die Events des Formulars (wie z.B. Schaltflächenklicks) sind weiterhin deaktiviert. Damit das Formular auf Events reagiert, rufen Sie es von einem Makro aus auf, z.B:

```
// Instantiate and display a form
ShowForm( "SampleForm" );
```

Aufrufen von Formularsteuerelementen

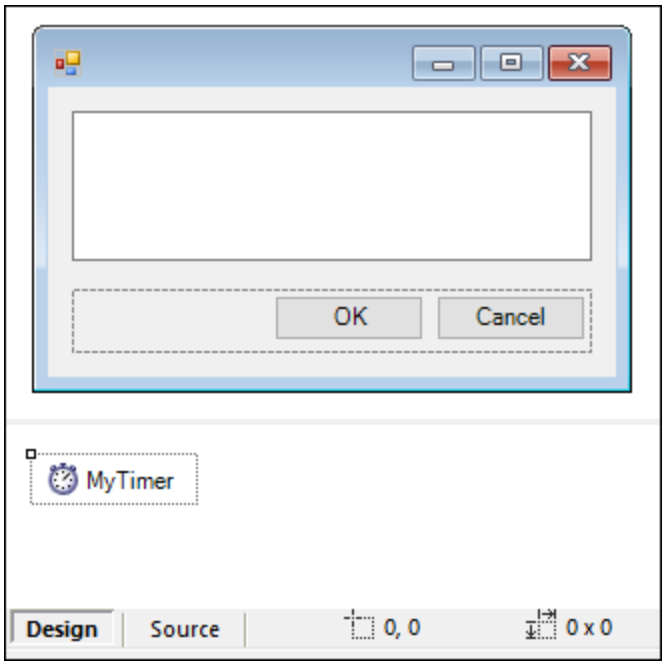
Sie können alle Komponenten eines Formulars von Ihrem Code aus mit Hilfe von Feldaufrufsyntax aufrufen. Angenommen, Sie haben das folgenden Formular:

```
// MyForm
//   ButtonPanel
//      OkButton
//      CancelButton
//   TextEditor
//      AxMediaPlayer1
// TrayComponents
//   MyTimer
```

Im unten stehenden Code sehen Sie, wie das Formular instantiiert wird, einige seiner Steuerelemente mittels Feldaufrufsyntax aufgerufen und das Formular anschließend angezeigt wird.

```
// Instantiate the form
var objForm = CreateForm("MyForm");
// Disable the OK button
objForm.ButtonPanel.OkButton.Enabled = false;
// Change the text of TextEditor
objForm.TextEditor.Text = "Hello";
// Show the form
objForm.ShowDialog();
```

Wenn Sie bestimmte Steuerelemente wie Timer zum Formular hinzufügen, werden diese nicht im Formular, sondern als Komponenten der Ablage am unteren Rand des Formulardesigns angezeigt, z.B:



Mit Hilfe der Methode `GetTrayComponent` können Sie Steuerelemente aus der Ablage aufrufen und den Namen des Steuerelements als Argument bereitstellen. Um in diesem Beispiel eine Referenz zu `MyTimer` zu erhalten und zu aktivieren, verwenden Sie den folgenden Code:

```
var objTimer = objForm.GetTrayComponent("MyTimer");
objTimer.Enabled = true;
```

Bei ActiveX Controls können Sie das zugrunde liegende COM-Objekt über die OCX-Eigenschaft aufrufen:

```
var ocx = lastform.AxMediaPlayer1.OCX; // get underlying COM object
ocx.enableContextMenu = true;
ocx.URL = "mms://apasf.apa.at/fm4_live_worldwide";
```

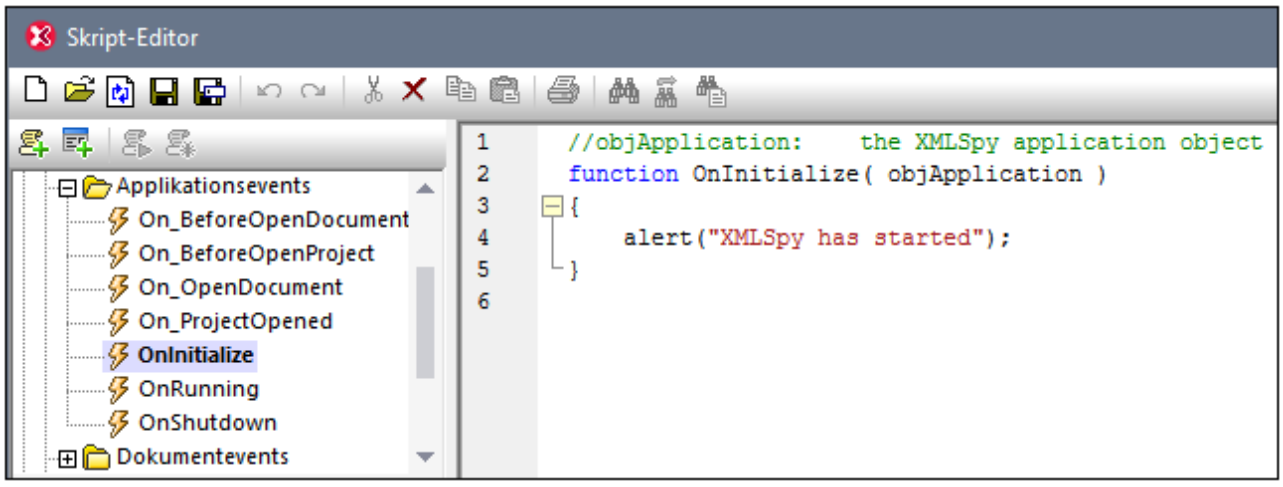

© 2016-2022 Altova GmbH

29.1.1.5 Events

Ihr Skripting-Projekt kann optional Skripts enthalten, die XMLSpy Events wie das Öffnen, Schließen oder Speichern eines Dokuments, das Starten oder Schließen von XMLSpy, das Hinzufügen eines Elements zu einem Diagramm und andere behandeln. Diese Events werden von der XMLSpy COM API bereitgestellt. Sie finden diese Events im Ordner "Ereignisse" Ihres Skripting-Projekts. Beachten Sie, dass diese Events im Gegensatz zu Formular-Events XMLSpy-spezifisch sind. Die Events befinden Sie in den folgenden Ordnern:

- Applikationsevents
- Dokumentevents
- Events der Authentic-Ansicht
- Events der Grid-Ansicht
- Events der Textansicht

Um ein Event Handler-Skript zu erstellen, klicken Sie mit der rechten Maustaste auf ein Event und wählen Sie im Kontextmenü den Befehl **Öffnen** (oder doppelklicken Sie auf das Event). Daraufhin wird das Event Handler-Skript im Hauptfenster, wo Sie es nun bearbeiten können, angezeigt. So zeigt z.B. der unten gezeigte Event Handler bei jedem Start von XMLSpy eine entsprechende Benachrichtigung an:



Beachten Sie die folgenden Punkte:

- Der Befehl `alert` ist auf JScript anwendbar. Das VBScript-Äquivalent dazu ist `MsgBox`. Siehe auch alert[1678].
- Der Name der Event Handler-Funktion darf nicht geändert werden, da das Event Handler-Skript sonst nicht aufgerufen wird.
- Damit Events verarbeitet werden, muss das Kontrollkästchen **Ereignisse abarbeiten** ausgewählt sein, wenn Sie das Skripting-Projekt in XMLSpy aktivieren. Nähere Informationen dazu finden Sie unter Aktivieren von Skripts und Makros[1687].

Sie können in Event Handler-Skripts optional lokale Variablen und Hilfsfunktionen definieren, z.B:

```
var local;

function OnInitialize( objApplication )
```

```
{
   local = "OnInitialize";
   Helper();
}

function Helper()
{
   alert("I'm a helper function for " + local);
}
```

29.1.1.6 Tipps zur Programmierung mit JScript

Im Folgenden finden Sie einige Tipps zum Programmieren in JScript, die sich bei der Erstellung eines Skripting-Projekts im XMLSpy Skript-Editor als nützlich erweisen könnten.

Out-Parameter

Für Out-Parameter von Methoden des .NET Framework werden in JScript spezielle Variablen benötigt, z.B:

```
var dictionary =
CLR.Create("System.Collections.Generic.Dictionary<System.String,System.String>");
dictionary.Add("1", "A");
dictionary.Add("2", "B");

// use JScript method to access out-parameters
var strOut = new Array(1);
if ( dictionary.TryGetValue("1", strOut) ) // TryGetValue will set the out parameter
  alert( strOut[0] ); // use out parameter
```

Ganzzahl-Argumente

.NET-Methoden, für die Ganzzahl-Argumente benötigt werden, sollten nicht direkt mit JScript-Zahlenobjekten, die Gleitkommawerte enthalten, aufgerufen werden. Verwenden Sie z.B. anstelle von:

```
var objCustomColor = CLR.Static("System.Drawing.Color").FromArgb(128,128,128);
```

Folgendes:

```
var objCustomColor =
CLR.Static("System.Drawing.Color").FromArgb(Math.floor(128),Math.floor(128),Math.floor(12
8));
```

Iterieren über .NET Collections

Für die Iteration über .NET Collections können sowohl der JScript Enumerator als auch .NET-Iterierungstechnologien verwendet werden, z.B:

© 2016-2022 Altova GmbH

```
// iterate using the JScript iterator
var itr = new Enumerator( coll );
for ( ; !itr.atEnd(); itr.moveNext() )
  alert( itr.item() );

// iterate using the .NET iterator
var itrNET = coll.GetEnumerator();
while( itrNET.MoveNext() )
  alert( itrNET.Current );
```

.NET Templates

.NET Templates können, wie unten gezeigt, instantiiert werden:

```
var coll = CLR.Create( "System.Collections.Generic.List<System.String>" );
```

oder

```
CLR.Import( "System" );
CLR.Import( "System.Collections.Generic" );
var dictionary = CLR.Create( "Dictionary<String,Dictionary<String,String>>" );
```

.NET-Enumerationswerte

.NET-Enumerationswerte werden, wie unten gezeigt, aufgerufen:

```
var enumValStretch = CLR.Static( "System.Windows.Forms.ImageLayout" ).Stretch;
```

Enumerationsliterale

Die Enumerationsliterale aus der XMLSpy API können, wie unten gezeigt, aufgerufen werden (ihr numerischer Wert muss nicht bekannt sein).

```
objExportXMIFileDlg.XMIType = eXMI21ForUML23;
```

29.1.1.7 Skripting-Beispielprojekt

Unter dem folgenden Pfad steht ein Demo-Projekt, in dem die Skripterstellung mit XMLSpy gezeigt wird, zur Verfügung: **C:\Benutzer\<Benutzer>\Dokumente\Altova\XMLSpy2023\Examples\SampleScripts.asprj**. Dieses Skripting-Projekt besteht aus einem Makro und einem Windows-Formular.

So laden Sie das Skripting-Projekt in den Skript-Editor:

1. Klicken Sie im Menü **Extras** auf **Skript-Editor**.
2. Klicken Sie auf **Öffnen** und navigieren Sie zur Datei **SampleScripts.asprj** unter dem obigen Pfad.

Das Projekt enthält im Verzeichnis "Makros" eine Reihe von Makros.

Makro	Beschreibung
AddMacroMenu	Dieses Makro fügt durch Aufruf der Methode `Application.AddMacroMenuItem` der COM API einen neuen Menüeintrag zu XMLSpy hinzu. Das erste Argument der Methode `AddMacroMenuItem` ist der Name des hinzuzufügenden Makros (in diesem Beispiel "CloseAllButActiveDoc") und das zweite Argument ist der Anzeigetext für den Menüeintrag. Immer, wenn dieses Makro ausgeführt wird, wird ein neuer Menübefehl namens "CloseAllButActiveDoc") zum Menü **Extras** hinzugefügt. Um zuvor erstellte Menüeinträge zu löschen, starten Sie XMLSpy entweder neu oder erstellen Sie ein Makro, das die API-Methode `Application.ClearMacroMenu` aufruft.
CloseAllButActiveDocument	Wenn dieses Makro ausgeführt wird, iteriert es durch die offenen Dokumente in XMLSpy und schließt alle bis auf das aktive Dokument.
SearchPath	Dieses Makro zeigt ein Formular an, über das Benutzer im aktuellen Projekt nach Dateien suchen können. Das Formular steht im Verzeichnis "Formulare", wo Sie sein Design und die damit verknüpften Event Handler anzeigen können, zur Verfügung. Die Methode `GetAllPathsFromProject()` gibt alle Dateipfade, die zum aktuell geöffneten Projekt gehören als Array zurück. Die Definition dieser Methode befindet sich im **GlobalDeclarations**-Skript dieses Projekts. Die `InsertStringInArrayUnique`-Methode stellt sicher, dass nur eindeutige Pfade zum Array hinzugefügt werden. Als nächstes wird das Formular mit CreateForm[1683] initialisiert. Schließlich wird das Array mit Hilfe der CLR.Create[1679]-Methode in einen .NET-Typ konvertiert und das Formular wird mit der erzeugten `ArrayList` Collection befüllt. Die Schaltfläche **Open** des Formulars hat einen Handler, der die API-Methode `Application.Documents.OpenFile` aufruft, um die aktuell ausgewählte Datei zu öffnen.

So aktivieren Sie das Skripting-Projekt als globales XMLSpy-Skripting-Projekt:

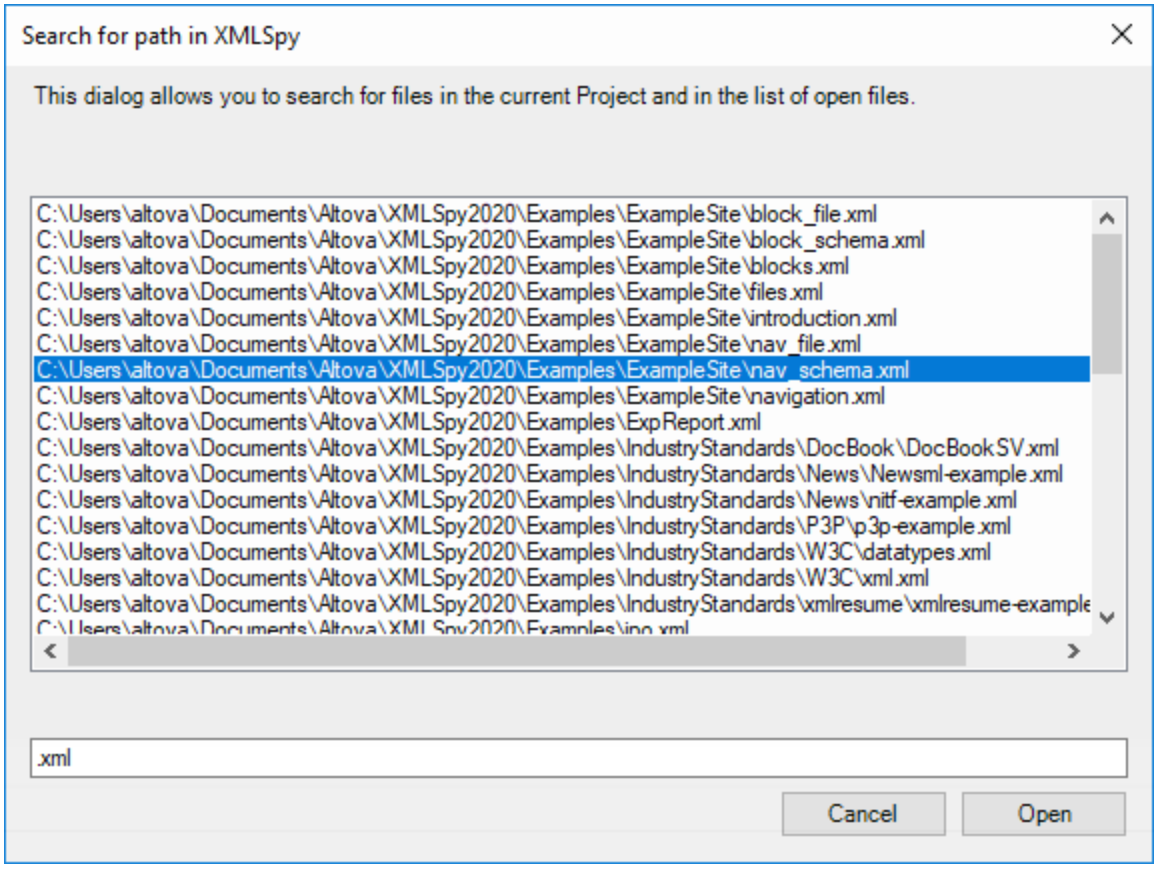
1. Klicken Sie im Menü **Extras** auf **Optionen**.
2. Klicken Sie auf das Register **Skripting-Umgebung**.
3. Klicken Sie unter "Globale Skripting-Projektdatei" auf **Durchsuchen** und wählen Sie die Datei **SampleScripts.asprj** aus dem obigen Pfad aus.
4. Dieses Skripting-Projekt hat keine automatischen Makros und Applikations-Event-Handler, daher müssen die Kontrollkästchen **Automatische Makros...ausführen** und **Ereignisse abarbeiten** nicht aktiviert werden.
5. Klicken Sie auf **Anwenden**.

Daraufhin stehen unter dem Menü **Extras | Makros** einige neue Menüeinträge (für jedes Makro einer) zur Verfügung.

© 2016-2022 Altova GmbH

So führen Sie das Makro "SearchPath" aus:

1. Öffnen Sie ein XMLSpy-Projekt, das mehrere Dateien enthält (in diesem Beispiel **C:\Benutzer\<Benutzer>\Dokumente\Altova\XMLSpy2023\Examples\Examples.spp).**
2. Klicken Sie im Menü **Extras** auf **Makros** und anschließend auf **Search Path**.
3. Geben Sie den Suchbegriff ein (in diesem Beispiel ".xml").



Wie oben gezeigt, werden daraufhin alle Dateinamen, die den Suchbegriff enthalten, aufgelistet. Sie können auf jedes beliebige Element in der Liste und anschließend auf **Open** klicken, um es im Haupt-Editor anzuzeigen.

29.1.2 Vordefinierte Befehle

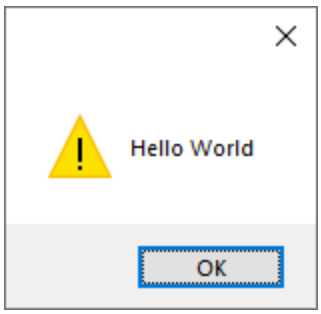
Dieser Abschnitt enthält eine Referenz aller Befehle, die Sie im XMLSpy Skript-Editor verwenden können.

- alert 1678
- confirm 1678
- CLR.Create 1679
- CLR.Import 1680
- CLR.LoadAssembly 1680
- CLR.ShowImports 1681
- CLR.ShowLoadedAssemblies 1682
- CLR.Static 1683

- CreateForm 1683
- doevents 1684
- lastform 1685
- prompt 1685
- ShowForm 1686
- watchdog 1687

29.1.2.1 alert

Zeigt ein Meldungsfeld mit einer angegebenen Meldung und der Schaltfläche "OK" an. Um fortzufahren, muss der Benutzer auf "OK" klicken.



Signatur

Für JScript lautet die Signatur:

```
alert(strMessage : String) -> void
```

Für VBScript lautet die Signatur:

```
MsgBox(strMessage : String) -> void
```

Beispiel

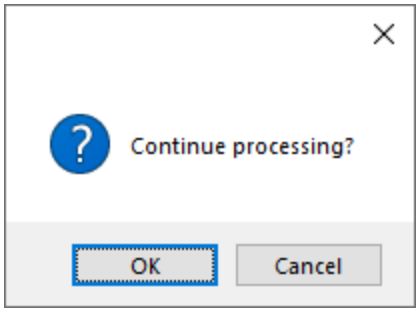
Mit dem folgenden JScript-Code wird ein Meldungsfeld mit dem Text "Hello World" angezeigt.

```
alert("Hello World");
```

29.1.2.2 confirm

Öffnet ein Dialogfeld, in dem eine angegebene Meldung, eine Bestätigungsschaltfläche und eine Abbrechen-Schaltfläche angezeigt werden. Der Benutzer muss entweder auf "OK" oder auf "Cancel" klicken, um fortfahren zu können. Der Rückgabewert ist ein Boolescher Wert, der für die Antwort des Benutzers steht. Wenn der Benutzer auf "OK" klickt, gibt die Funktion **true** zurück, wenn der Benutzer auf "Cancel" klickt, gibt das Dialogfeld **false** zurück.

© 2016-2022 Altova GmbH



Signatur

```
confirm(strMessage : String) -> result : Boolean
```

Beispiel (JScript)

```
if ( confirm( "Continue processing?" ) == false )
  alert("You have cancelled this action");
```

Beispiel (VBScript)

```
If ( confirm( "Continue processing?" ) = false ) Then
  MsgBox("You have cancelled this action")
End If
```

29.1.2.3 CLR.Create

Erstellt eine neue .NET-Objektinstanz für den als Argument angegebenen Typnamen. Wenn mehr als ein Argument übergeben wird, werden die nachfolgenden Argumente als Argumente für den Konstruktor des .NET-Objekts interpretiert. Der Rückgabewert ist eine Referenz auf das erstellte .NET-Objekt.

Signatur

```
CLR.Create(strTypeNameCLR : String, constructor arguments ... ) -> object
```

Beispiel

Im folgenden JScript-Code wird gezeigt, wie Sie Instanzen von verschiedenen .NET-Klassen erstellen.

```
// Create an ArrayList
var objArray = CLR.Create("System.Collections.ArrayList");
// Create a ListViewItem
var newItem = CLR.Create( "System.Windows.Forms.ListViewItem", "NewItemText" );
// Create a List<string>
```

```
var coll = CLR.Create( "System.Collections.Generic.List<System.String>" );
// Import required namespaces and create a Dictionary object
CLR.Import( "System" );
CLR.Import( "System.Collections.Generic" );
var dictionary = CLR.Create( "Dictionary< String, Dictionary< String, String > >" );
```

29.1.2.4 CLR.Import

Importiert einen Namespace. Dies ist das Skripting-Äquivalent zu den C# und VB.Net-Schlüsselwörtern `using` und `imports`. Damit können Sie den Namespace-Teil in aufeinander folgenden Aufrufen wie z.B. CLR.`Create()` und `CLR.Static()` auslassen.

Anmerkung: Durch den Import eines Namespace wird die entsprechende Assembly nicht zum Skripting-Projekt hinzugefügt bzw. geladen. Assemblys können mittels CLR.LoadAssembly[1680] dynamisch (zur Laufzeit) im Quellcode eingefügt werden.

Signatur

```
CLR.Import(strNamespaceCLR : String) -> void
```

Beispiel

Anstatt vollständig qualifizierte Namespaces wie die folgenden verwenden zu müssen:

```
if ( ShowForm( "FormName" ) == CLR.Static( "System.Windows.Forms.DialogResult" ).OK )
{
    var sName = lastform.textboxFirstName.Text + " " + lastform.textboxLastName.Text;
    CLR.Static( "System.Windows.Forms.MessageBox" ).Show( "Hello " + sName );
}
```

können Sie Namespaces zuerst importieren und danach die Kurzform verwenden:

```
CLR.Import( "System.Windows.Forms" );

if ( ShowForm( "FormName" ) == CLR.Static( "DialogResult" ).OK )
{
    var sName = lastform.textboxFirstName.Text + " " + lastform.textboxLastName.Text;
    CLR.Static( "MessageBox" ).Show( "Hello " + sName );
}
```

29.1.2.5 CLR.LoadAssembly

Lädt die .NET Assembly mit dem angegebenen Assembly-Namen oder Dateipfad. Gibt den Booleschen Wert **true** zurück, wenn die Assembly geladen werden konnte, **false**, wenn nicht.

© 2016-2022 Altova GmbH

Signatur

```
CLR.LoadAssembly(strAssemblyNameCLR : String, showLoadErrors : Boolean) -> result :
Boolean
```

Beispiel

Mit dem folgenden JScript-Code wird versucht, den Text in der Zwischenablage durch dynamisches Laden der erforderlichen Assembly zu definieren.

```
// set clipboard text (if possible)
// System.Windows.Clipboard is part of the PresentationCore assembly, so load this
assembly first:
if ( CLR.LoadAssembly( "PresentationCore, Version=3.0.0.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35", true ) )
{
    var clipboard = CLR.Static( "System.Windows.Clipboard" );
    if ( clipboard != null )
        clipboard.SetText( "HelloClipboard" );
}
```

29.1.2.6 CLR.ShowImports

Öffnet ein Meldungsfeld, in dem die aktuell importierten Namespaces angezeigt werden. Der Benutzer muss zum Fortfahren auf OK klicken.

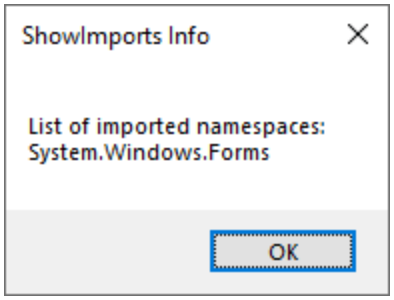
Signatur

```
CLR.ShowImports() -> void
```

Beispiel

Mit dem folgenden JScript-Code wird zuerst ein Namespace importiert und anschließend die Liste der importierten Namespaces angezeigt:

```
CLR.Import( "System.Windows.Forms");
CLR.ShowImports();
```



29.1.2.7 CLR.ShowLoadedAssemblies

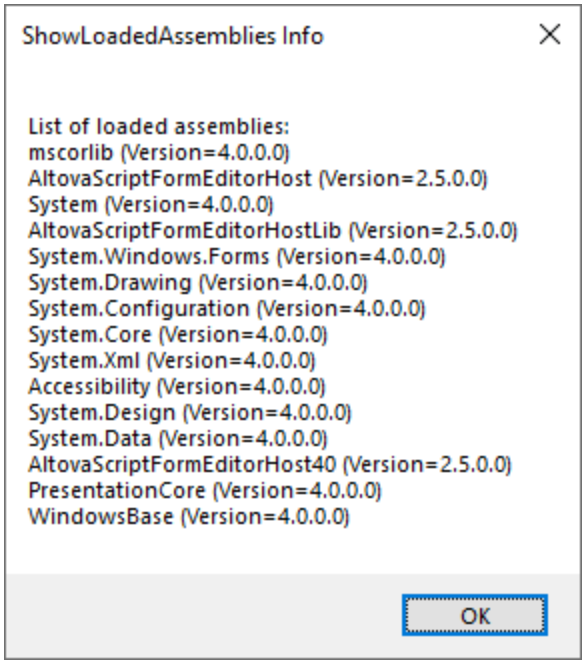
Öffnet ein Meldungsfeld, in dem die aktuell geladenen Assemblys angezeigt werden. Der Benutzer muss zum Fortfahren auf OK klicken.

Signatur

```
CLR.ShowLoadedAssemblies() -> void
```

Beispiel

```
CLR.ShowLoadedAssemblies();
```



© 2016-2022 Altova GmbH

29.1.2.8 CLR.Static

Gibt eine Referenz auf ein statisches .NET-Objekt zurück. Verschafft Zugriff auf .NET-Typen, die keine Instanzen haben und nur statische Mitglieder enthalten.

Signatur

```
CLR.Static(strTypeNameCLR : String) -> object
```

Beispiel (JScript)

```
// Get the value of a .NET Enum into a variable
var enumValStretch = CLR.Static( "System.Windows.Forms.ImageLayout" ).Stretch

// Set the value of the Windows clipboard
var clipboard = CLR.Static( "System.Windows.Clipboard" );
clipboard.SetText( "HelloClipboard" );

// Check the buttons pressed by the user on a dialog box
if ( ShowForm( "FormName" ) == CLR.Static( "System.Windows.Forms.DialogResult" ).OK )
    alert( "ok" );
else
    alert( "cancel" );
```

29.1.2.9 CreateForm

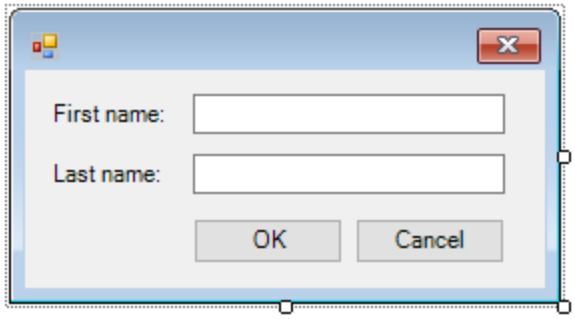
Instantiiert das `Form`-Objekt, das durch den als Argument bereitgestellten Namen identifiziert wird. Das Formular muss im Ordner "Formulare" des Skripting-Projekts vorhanden sein. Gibt das Formularobjekt (`System.Windows.Forms.Form`) des angegebenen Namens oder `null` zurück, wenn kein Formular dieses Namens vorhanden ist.

Signatur

```
CreateForm (strFormName : String) -> System.Windows.Forms.Form | null
```

Beispiel

Angenommen, es gibt im Skripting-Projekt ein Formular namens "FormName".



Der folgende JScript-Code instantiiert das Formular mit einige Standardwerten und zeigt es dem Benutzer an.

```
var myForm = CreateForm( "FormName" );
if ( myForm != null )
{
   myForm.textboxFirstName.Text = "Daniela";
   myForm.textboxLastName.Text = "Heidegger";
   var dialogResult = myForm.ShowDialog();
}
```

Das `dialogResult` kann anschließend folgendermaßen ausgewertet werden:

```
if ( dialogResult == CLR.Static( "System.Windows.Forms.DialogResult" ).OK )
    alert( "ok" );
else
    alert( "cancel" );
```

Anmerkung: Der oben gezeigte Code funktioniert nur, wenn die Eigenschaft **DialogResult** der Schaltflächen "OK" und "Cancel" über das Eigenschaftsfenster korrekt definiert wurde (für die Schaltfläche "OK" muss er z.B. **OK** sein).

29.1.2.10 doevents

Verarbeitet alle Windows-Meldungen, die sich derzeit in der Meldungswarteschlange befinden.

Signatur

```
doevents() -> void
```

Beispiel (JScript)

```
for ( i=0; i < nLongLastingProcess; ++i )
{
    // do long lasting process
```

 © 2016-2022 Altova GmbH

```
    doevents(); // process Windows messages; give UI a chance to update
}
```

29.1.2.11 lastform

Dies ist ein globales Feld, das eine Referenz auf das zuletzt mit `CreateForm()` oder `ShowForm()` erstellte Formularobjekt zurückgibt.

Signatur

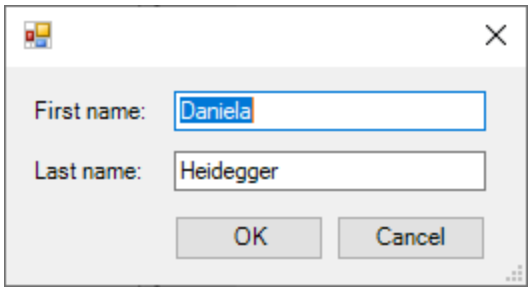
```
lastform -> formObj : System.Windows.Forms.Form
```

Beispiel

Mit dem folgenden JScript-Code wird das Formular "FormName" als Dialogfeld angezeigt.

```
CreateForm( "FormName" );
if ( lastform != null )
{
    lastform.textboxFirstName.Text = "Daniela";
    lastform.textboxLastName.Text = "Heidegger";
    var dialogResult = lastform.ShowDialog();
}
```

Die Werte beider Textfeld-Steuerelemente werden mit Hilfe von `lastform` initialisiert.



29.1.2.12 prompt

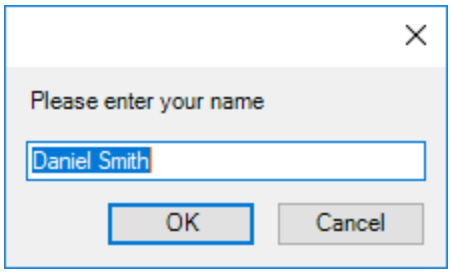
Öffnet ein Dialogfeld, in dem einem Meldung sowie ein Textfeld-Steuerelement mit einer Standardantwort angezeigt wird. Damit kann der Benutzer einen einfachen String-Wert eingeben. Der Rückgabewert ist ein String, der den Textfeldwert enthält oder Null, wenn der Benutzer auf "Cancel" geklickt hat.

Signatur

```
prompt(strMessage : String, strDefault : String) -> val : String
```

Beispiel

```
var name = prompt( "Please enter your name", "Daniel Smith" );
if ( name != null )
    alert( "Hello " + name + "!" );
```



29.1.2.13 ShowForm

Instantiiert ein neues Formularobjekt anhand des angegebenen Formularnamens und zeigt es sofort als Dialogfeld an. Der Rückgabewert ist eine Ganzzahl, die für das generierte Dialogfeldergebnis `DialogResult` (`System.Windows.Forms.DialogResult`) steht. Eine Liste der möglichen Werte finden Sie in der Dokumentation zur `DialogResult` Enum (https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.dialogresult?view=netframework-4.8).

Signatur

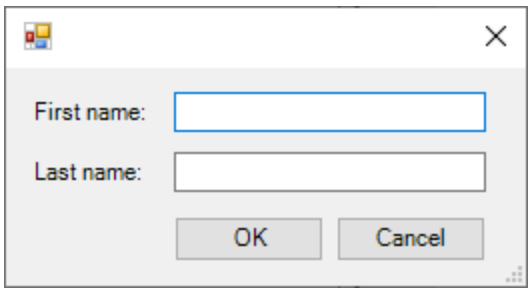
```
ShowForm(strFormName : String) -> result : Integer
```

Beispiel

Der folgende JScript-Code

```
var dialogResult = ShowForm( "FormName" );
```

zeigt das Formular "FormName" als Dialogfeld an:



Das Ergebnis `DialogResult` kann anschließend ausgewertet werden, z.B:

© 2016-2022 Altova GmbH

```
if ( dialogResult == CLR.Static( "System.Windows.Forms.DialogResult" ).OK )
    alert( "ok" );
else
    alert( "cancel" );
```

Anmerkung: Der oben gezeigte Code funktioniert nur, wenn die Eigenschaft **DialogResult** der Schaltflächen "OK" und "Cancel" über das Eigenschaftsfenster korrekt definiert wurde (für die Schaltfläche "OK" muss er z.B. **OK** sein).

29.1.2.14 watchdog

Bei CPU-intensiven Skripts, deren Verarbeitung lange dauert, fragt der Watchdog den Benutzer, ob das Skript abgebrochen werden soll. Mit Hilfe der `watchdog()` Methode können Sie dieses Verhalten aktivieren bzw. deaktivieren. Standardmäßig ist der Watchdog aktiviert.

Durch Aufruf von `watchdog(true)` können Sie den Watchdog auch zurücksetzen. Dies empfiehlt sich manchmal vor der Ausführung langer speicherintensiver Aufgaben, um sicherzustellen, dass diesen der maximale vom Skript her zulässige Verarbeitungsspeicher zur Verfügung steht.

Signatur

```
watchdog(bEnable : boolean) -> void
```

Beispiel

```
watchdog( false );  // disable watchdog - we know the next statement is CPU intensive but
it will terminate for sure
doCPUIntensiveScript();
watchdog( true );   // re-enable watchdog
```

29.1.3 Aktivieren von Skripts und Makros

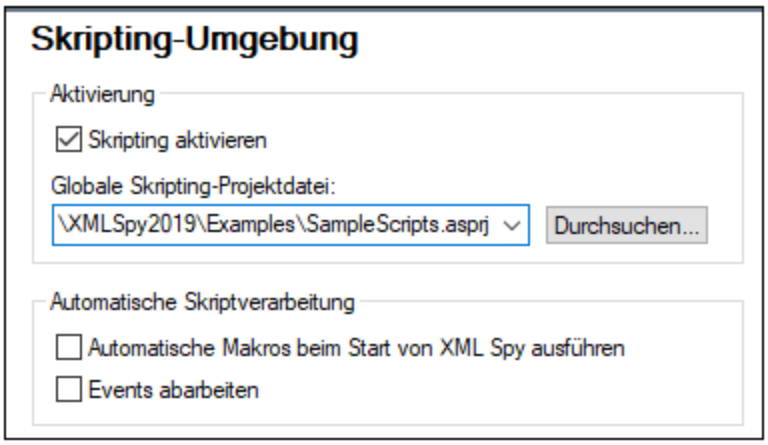
Nachdem ein Skripting-Projekt fertig gestellt und getestet wurde, können Sie es auf folgende Arten verwenden:

1. Als das globale Skripting-Projekt für XMLSpy. Das bedeutet, dass alle Skripts und Makros aus dem Skripting-Projekt XMLSpy zur Verfügung stehen.
2. Auf XMLSpy-Projektebene. Das bedeutet, es wird eine Referenz auf die .asprj-Datei zusammen mit dem XMLSpy-Projekt gespeichert. Wenn das XMLSpy-Projekt geöffnet wird, können die damit verknüpften Skripts und Makros aufgerufen werden.

So definieren Sie ein Skripting-Projekt als globales Skripting-Projekt:

1. Klicken Sie im Menü **Extras** auf **Optionen**.
2. Klicken Sie auf das Register **Skripting-Umgebung**.

3. Aktivieren Sie das Kontrollkästchen **Skripting aktivieren** und navigieren Sie zu der .asprj-Datei, die als globales Skripting-Projekt verwendet werden soll.



Sie können optional die folgenden zusätzlichen Skriptverarbeitungsoptionen aktivieren:

Automatische Makros beim Start von XMLSpy ausführen	Wenn Sie dieses Kontrollkästchen aktivieren, werden alle im Projekt als automatische Makros definierten Makros beim Start von XMLSpy automatisch ausgelöst.
Ereignisse abarbeiten	Aktivieren Sie dieses Kontrollkästchen, wenn Ihre Skripts an Applikations-Events gebunden sind. Deaktivieren Sie dieses Kontrollkästchen, damit die Skripts nicht auf Events reagieren.

So aktivieren Sie ein Skripting-Projekt auf Projektebene:

1. Öffnen Sie das Projekt.
2. Klicken Sie im Menü **Projekt** auf **Skript-Einstellungen**.
3. Aktivieren Sie das Kontrollkästchen **Projektskripts aktivieren** und navigieren Sie zur .asprj-Datei.

Das Kontrollkästchen **Beim Laden des... automatische Makros ausführen** hat dieselbe Bedeutung wie oben beschrieben.

29.1.3.1 Ausführen von Makros

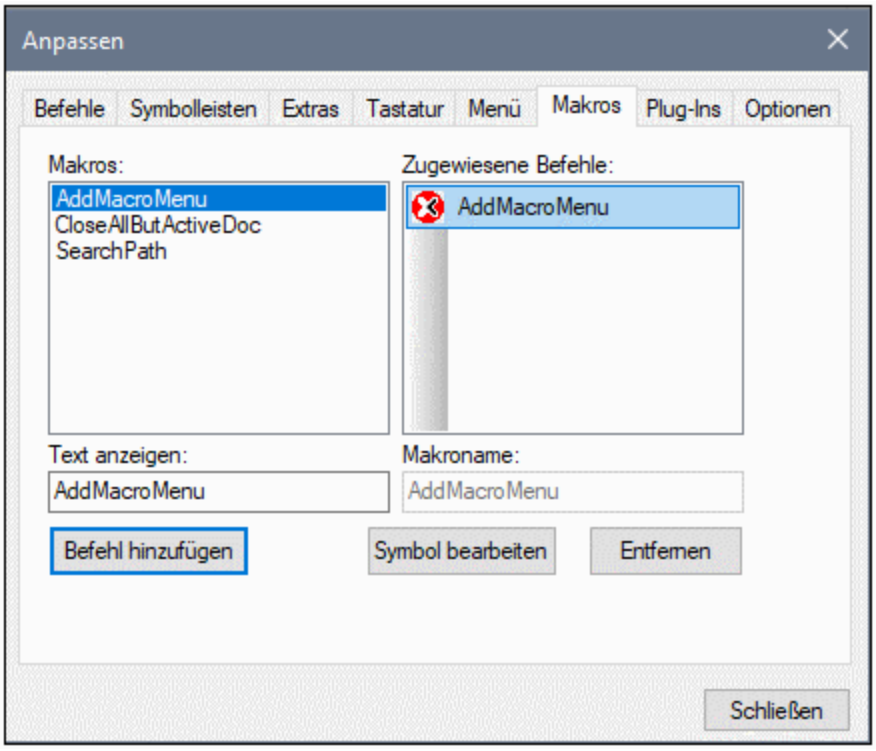
Wenn ein Skripting-Projekt in XMLSpy aktiv ist, werden alle in diesem Projekt verfügbaren Makros im Menü **Extras | Makros** angezeigt. Sie können ein Makro daher durch Ausführen des entsprechenden Menübefehls wie z.B. **Extras | Makros | <Makroname>** jederzeit ausführen.

Makros die als automatische Makros konfiguriert wurden, werden bei jedem Start von XMLSpy automatisch ausgeführt, vorausgesetzt dieses Verhalten wurde in den Optionen, wie unter Aktivieren von Skripts und Makros 1687 beschrieben, aktiviert.

Sie können auch Symbolleisten-Schaltflächen für Makros erstellen. Gehen Sie dazu folgendermaßen vor:

 © 2016-2022 Altova GmbH

1. Klicken Sie im Menü **Extras** auf **Anpassen**.
2. Klicken Sie auf das Register **Makros**. Alle auf Applikationsebene (im *globalen* Skripting-Projekt) verfügbaren Makros werden aufgelistet.
3. Klicken sie auf **Befehl hinzufügen**.



4. Klicken Sie optional auf **Symbol bearbeiten** und erstellen Sie ein neues Symbol für das neue Makro. Sie können dem Makro über das Register **Tastatur** auch ein Tastaturkürzel zuweisen.
5. Ziehen Sie das Makro aus dem Bereich **Zugewiesene Befehle** an die gewünschte Stelle in der Symbolleiste.

So entfernen Sie ein Makro aus einer Symbolleiste:

1. Klicken Sie im Menü **Extras** auf **Anpassen**.
2. Klicken Sie auf das Register **Makros**.
3. Ziehen Sie das Makro aus der Symbolleiste, in der es angezeigt wird, zurück in den Bereich **Zugewiesene Befehle.**

29.2 IDE Plugins

Sie können in XMLSpy Ihre eigenen IDE-Plug-Ins erstellen und in XMLSpy integrieren.

Plug-Ins können zu folgenden Zwecken verwendet werden:

- zum Konfigurieren Ihrer Version von XMLSpy, zum Hinzufügen von Befehlen in Form von Menübefehlen, Symbolleisten-Schaltflächen, Schaltflächen usw.
- zum Reagieren auf Events aus XMLSpy.
- zum Ausführen Ihres eigenen Codes in XMLSpy, wobei Sie Zugriff auf die komplette XMLSpy API haben.

XMLSpy erwartet von Ihrem Plug-In, dass es die IXMLSpyPlugIn[1700] Schnittstelle implementiert. C#- und C++ sind die derzeit unterstützten Sprachen und Ihr Installationspaket enthält im Ordner `XMLSpy2023\Examples\IDEPlugin` Ihrer XMLSpy Beispiele zu diesen Sprachen.

Windows 7, 8, 10, 11	`C:/Benutzer/<Benutzername>/Dokumente`

Ein Beispiel zur Verwendung von C++ finden Sie in den ATL-Beispieldateien[1694].

29.2.1 Registrierung von IDE Plug-Ins

XMLSpy enthält einen speziellen Schlüssel in der Registrierungsdatei, wo alle registrierten IDE-Plug-Ins gespeichert sind:

```
HKEY_CURRENT_USER\Software\Altova\XML Spy\PlugIns
```

Alle Werte dieses Schlüssels werden als Referenzen auf registrierte Plug-Ins behandelt und müssen dem folgenden Format entsprechen:

Wertname:	ProgID des Plug-in
Werttyp:	muss REG_SZ sein
Wertdaten:	CLSID der Komponente

Bei jedem Start der Applikation werden die Werte des **`PlugIns`**-Schlüssels gelesen und die registrierten Plug-ins werden geladen.

Manuelle Registrierung von Plug-Ins

Um ein Plug-In manuell zu registrieren, verwenden Sie in XMLSpy im Menü **Extras** das Dialogfeld "Anpassen". Klicken Sie auf die Schaltfläche **Plug-in hinzufügen...** um die DLL anzugeben, die Ihr Plug-In implementiert. XMLSpy registriert die DLL als einen COM Server und fügt den entsprechenden Eintrag in seinem **`PlugIns`** Schlüssel hinzu.

Wenn es bei der manuellen Registrierung zu Problemen kommt, überprüfen Sie, ob die CLSID Ihres Plug-in korrekt im **`PlugIns`** Schlüssel registriert wurde. Wenn dies nicht der Fall ist, war der Name Ihrer Plug-In DLL

© 2016-2022 Altova GmbH

wahrscheinlich nicht eindeutig genug. Verwenden Sie einen anderen Namen oder führen Sie eine direkte Registrierung durch.

Direkte Registrierung von Plug-Ins

Ein Plug-In kann direkt als IDE Plug-In registriert werden. Dazu muss zuerst die DLL registriert und anschließend müssen z.B. beim Einrichten des Plug-In die entsprechenden Werte zum **`PlugIns`** Schlüssel von XMLSpy hinzugefügt werden. Das neue Plug-In wird beim nächsten Start von XMLSpy aktiviert.

Erstellen von Plug-Ins

Im Ordner (Eigene) Dateien[34]: `Examples\IDEPlugin` der Applikation finden Sie Quellcode für Beispiel-Plug-Ins. Um ein Plug-in anhand von solchem Quellcode zu erstellen, gehen Sie folgendermaßen vor:

1. Öffnen Sie die Lösung, die Sie als Plug-in in Visual Studio erstellen möchten.
2. Erstellen Sie das Plug-in mit dem Befehl im Menü "Build".
3. Die DLL-Datei des Plug-in wird im Ordner `Bin` oder `Debug` erstellt. Diese DLL ist die Datei, die als Plug-in hinzugefügt werden muss (*siehe oben*).

Anmerkung: C# und C++ sind die derzeit unterstützten Sprachen.

29.2.2 ActiveX Controls

ActiveX Controls werden unterstützt. Jedes IDE PlugIn, das auch ein ActiveX Control ist, wird in einer Dialogfeldsteuerleiste angezeigt. Im Ordner `Examples` Ihres Ordners `XMLSpyPlugInActiveX` des Applikationsordners finden Sie auch ein Beispiel-Plug-In, das auch ein ActiveX Control ist.

29.2.3 Konfigurations-XML

Mit Hilfe des IDE-Plug-In können Sie die Benutzeroberfläche von XMLSpy ändern. Dazu wird jede einzelne Änderung mit Hilfe eines XML-Datenstroms beschrieben. Die XML-Konfiguration wird über die Methode GetUIModifications[1704] der IXMLSpyPlugIn-Schnittstelle an XMLSpy übergeben.

Die XML-Datei für das IDE Plug-In, die die Änderungen an der Benutzeroberfläche enthält, muss die folgende Struktur haben:

```
<ConfigurationData>
   <ImageFile>Path to image file</ImageFile>
   <Modifications>
      <Modification>
         ...
      </Modification>
         ...
    </Modifications>
</ConfigurationData>
```

Sie können Symbole oder Symbolleisten-Schaltflächen für die neuen Menüeinträge, die über das Plug-In zur Benutzeroberfläche von XMLSpy hinzugefügt werden, definieren. Der Pfad zur Datei, die das Bild enthält, wird

über das Element `ImageFile` definiert. Jedes Bild muss 16x16 Pixel haben und darf maximal 256 Farben enthalten. Die Bildreferenzen müssen in einem einzigen **<ImageFile>**-Element von links nach rechts angeordnet werden. Der am weitesten rechts gelegene Bild-Indexwert ist Null.

Das Element **Modification** kann beliebig viele **Modification** Child-Elemente enthalten. Jedes **Modification**-Element definiert eine bestimmte Änderung an der Standardbenutzeroberfläche von XMLSpy. Welche Änderungen Sie vornehmen können, ist im nächsten Abschnitt weiter unten beschrieben.

Struktur von Modification-Elementen

Ein `Modification`-Element besteht aus den folgenden beiden Child-Elementen:

```
<Modification>
   <Action>Type of action</Action>
   <UIElement Type="Type of UI element" />
</Modification>
```

Gültige Werte für das Element **Action** sind:

Add: um das darauf folgende Benutzeroberflächenelement zu XMLSpy hinzuzufügen
Hide: um das darauf folgende Benutzeroberflächenelement in XMLSpy auszublenden
Remove: um das Benutzeroberflächenelement im Dialogfeld "Anpassen" aus dem Listenfeld "Befehle" zu entfernen.

Mehrere Änderungen können folgendermaßen in einem **Action**-Element kombiniert werden: **"Add Hide"**

Das Element **UIElement** definiert alle neuen oder bestehenden Elemente der Benutzeroberfläche und kann eines der folgenden Typen sein: Symbolleisten, Schaltflächen, Menüs oder Menübefehle. Das Attribut **type** definiert, zu welchem dieser Typen das Benutzeroberflächen-Element gehört. Die Struktur von **UIElement** wird im folgenden Abschnitt beschrieben.

Gemeinsame Child-Elemente von UIElement

Die Elemente `ID` und `Name` sind für alle verschiedenen Typen von UI-Elementen definiert. Bei einigen Typen wird jedoch eines dieser Elemente ignoriert. So wird z.B. **Name** für ein Trennzeichen ignoriert.

```
<ID></ID>
<Name></Name>
```

Wenn **UIElement** ein bestehendes Element der Benutzeroberfläche beschreibt, wird der Wert des Elements **ID** von XMLSpy vorgegeben. Normalerweise sind diese ID-Werte dem User nicht bekannt. Wenn das XML-Fragment einen neuen Teil der Benutzeroberfläche beschreibt, kann die ID beliebig gewählt werden und der Wert sollte kleiner als **1000** sein. Das Element **Name** definiert den Textwert. Bestehende UI-Elemente können nur nach ihrem Namen identifiziert werden, z.B. Menüs und Menübefehle, die Untermenüs haben. Bei neuen Benutzeroberflächenelementen definiert das Element **Name** die Überschrift, z.B. den Titel einer Symbolleiste oder den Text eines Menübefehls.

Symbolleisten und Menüs

Um eine Symbolleiste zu definieren, muss die ID und/oder der Name der Symbolleiste definiert werden. Eine bestehende Symbolleiste kann nur durch den Namen oder die ID, unter der sie bekannt ist, definiert werden.

© 2016-2022 Altova GmbH

Um eine **neue** Symbolleiste zu erstellen, müssen beide Werte definiert werden. Das Attribut **type** muss **ToolBar** sein.

```
<UIElement Type="ToolBar">
   <ID>1</ID>
   <Name>TestPlugIn</Name>
</UIElement>
```

Um ein XMLSpy Menü zu definieren, benötigen Sie zwei Parameter:

- die ID der Menüleiste, die das Menü enthält. Wenn im Hauptfenster keine XML-Dokumente geöffnet sind, ist die ID der Menüleiste `128`. Wenn ein oder mehrere Dokumente geöffnet sind, ist die ID der Menüleiste `129`.
- den Menünamen. Menüs haben keinen dazugehörigen ID-Wert. Im folgenden Beispiel wird das Menü "Edit" (Bearbeiten) der Menüleiste definiert, die aktiv ist, wenn mindestens ein XML-Dokument geöffnet ist:

```
<UIElement Type="Menu">
   <ID>129</ID>
   <Name>Edit</Name>
</UIElement>
```

Wenn Sie ein neues Menü erstellen möchten, wird ein zusätzliches Element verwendet. Das Element **Place** definiert die Position des neuen Menüs in der Menüleiste:

```
<UIElement Type="Menu">
   <ID>129</ID>
   <Name>PlugIn Menu</Name>
   <Place>12</Place>
</UIElement>
```

Ein Wert von `-1` für das Element `Place` definiert die neue Schaltfläche oder den neuen Menübefehl am Ende des Menüs oder der Symbolleiste.

Befehle

Wenn Sie einen neuen Befehl in Form einer Symbolleisten-Schaltfläche oder eines Menübefehls hinzufügen, kann das **UIElement**-Fragment jedes beliebige dieser Subelemente enthalten:

```
<MacroName></MacroName>
<Info></Info>
<ImageID></ImageID>
```

Wenn **MacroName** definiert ist, sucht XMLSpy in der Skripting-Umgebung nach einem Makro dieses Namens und führt es jedes Mal, wenn dieser Befehl aufgerufen wird, aus. Das Element **Info** enthält einen Beschreibungsstring, der in der Statusleiste angezeigt wird, wenn sich der Mauszeiger über dem entsprechenden Befehl (Schaltfläche oder Menübefehl) befindet. **ImageID** definiert den Index des Symbols in der Bilddatei. Bitte beachten Sie, dass alle Symbole in einer Bilddatei gespeichert sind.

Um einen Symbolleistenbefehl zu definieren, erstellen Sie ein `UIElement` mit der folgenden Struktur:

```
<UIElement Type="ToolBarItem">
   <!--don't reuse local IDs even the commands do the same-->
```

```
   <ID>5</ID>
   <Name>Open file from repository...</Name>
   <!--Set Place To -1 If this is the first button To be inserted-->
   <Place>-1</Place>
   <ImageID>0</ImageID>
   <ToolBarID>1</ToolBarID>
   <!--instead of the toolbar ID the toolbar name could be used-->
   <ToolBarName>TestPlugIn</ToolBarName>
</UIElement>
```

Zusätzliche Elemente zum Deklarieren einer Symbolleisten-Schaltfläche sind **`Place`**, **`ToolBarID`** und **`ToolBarName`**. `ToolBarID` und `ToolBarName` dienen dazu, die Symbolleiste zu identifizieren, die die neue oder bestehende Schaltfläche enthält. Die Groß- und Kleinschreibung des Textwerts von **`ToolBarName`** spielt eine Rolle. Das Attribut **`type`** von UIElement muss **`ToolBarItem`** sein.

Um einen Menübefehl zu definieren, stehen zusätzlich zu den Standardelementen, die zum Deklarieren eines Befehls verwendet werden, die Elemente **`MenuID`**, **`Place`** und **`Parent`** zur Verfügung. Der Inhalt des **`MenuID`**-Elements kann **`128`** oder **`129`** sein. *Nähere Informationen zu diesen Werten finden Sie unter "Symbolleisten und Menüs".*

Das Element **`Parent`** dient dazu, das **Menü** anzugeben, in das der neue Menübefehl eingefügt werden soll. Da die Befehle von Untermenüs keine eindeutige Windows ID haben, benötigen wir eine andere Methode, um das übergeordnete Element des Menübefehls zu definieren. Wir tun dies, indem wir als Inhalt des **`Parent`**-Elements den Pfad zu Menübefehl definieren. Die Schritte im Pfad werden durch einen Doppelpunkt gekennzeichnet. Das Muster wäre **`ParentMenü:Submenü`**. Wenn das Menü kein übergeordnetes Menü hat (da es kein Untermenü ist), stellen Sie dem Namen am Anfang einen Doppelpunkt voran (siehe Beispiel unten). Das Attribut **`type`** muss den Wert **`MenuItem`** haben.

Im Beispiel unten wird ein Menübefehl definiert, dessen übergeordnetes Menü kein Untermenü ist:

```
<UIElement Type="MenuItem">
   <!--the following element is a Local command ID-->
   <ID>3</ID>
   <Name>Open file from repository...</Name>
   <Place>-1</Place>
   <MenuID>129</MenuID>
   <Parent>:PlugIn Menu</Parent>
   <ImageID>0</ImageID>
</UIElement>
```

Sie können Symbolleistentrennzeichen und Menüs hinzufügen, wenn der Wert des **`ID`** Elements auf **`0`** gesetzt wird.

29.2.4 ATL-Beispieldateien

In diesem Abschnitt wird beschrieben, wie Sie mit Hilfe von ATL eine einfache XMLSpy IDE Plug-In DLL erstellen. Sie müssen mit der Arbeit mit MSVisualStudio, ATL und den Assistenten zur Generierung von ATL-Objekten vertraut sein. Zum Aufrufen der API importiert die Implementierung die Typbibliothek von XMLSpy. Der Code liest verschiedene Eigenschaften aus und ruft mit Hilfe der über die **`#import`**-Anweisung zur Verfügung

© 2016-2022 Altova GmbH

gestellten Smart Pointer Methoden auf. Außerdem werden im Beispielcode der MFC Class `CString` und die ATL-Konvertierungsmakros wie z.B. W2T verwendet.

Die grundlegenden Schritte zum Erstellen einer ATL DLL lauten wie folgt:

1. Öffnen Sie VisualStudio und wählen Sie **File | New**.
2. Wählen Sie das Register *Projects*.
3. Wählen Sie *ATL COM AppWizard* und geben Sie einen Projektnamen ein.
4. Aktivieren Sie *Support for MFC*, wenn Sie MFC-Klassen verwenden oder ein Projekt für den Beispielcode erstellen möchten.

Nach Erstellung der Projektdateien können Sie ein ATL-Objekt hinzufügen, um die `IXMLSpyPlugIn`-Schnittstelle zu implementieren:

1. Wählen Sie **Insert | New ATL Object**.
2. Wählen Sie im Assistenten die Option *Simple Object* und klicken Sie auf **Next**.
3. Geben Sie einen Namen für das Objekt ein.
4. Wählen Sie auf dem Register *Attributes* als Schnittstellentyp *Custom* aus und deaktivieren Sie die *Aggregation*.

Damit wird der grundlegende Code für die Implementierung der IDE Plug-In Schnittstelle erzeugt. Wie man den Code ändert und einige grundlegende Funktionalitäten definiert, wird auf den nächsten Seiten beschrieben.

29.2.4.1 Schnittstellenbeschreibung (IDL)

Die IDL des neu erstellten ATL-Objekts enthält eine Deklaration für eine COM-Schnittstelle.

- Diese Schnittstellenbeschreibung muss, wie unten gezeigt, durch die Deklaration von IXMLSpyPlugIn ersetzt werden.
- Die IDL muss außerdem die Definition der SPYUpdateAction Enumeration enthalten.
- Ersetzen Sie in der coclass-Deklaration den (vom Assistenten) generierten Standard-Schnittstellennamen durch "IXMLSpyPlugIn".

Die IDL sollte anschließend in etwa so aussehen wie im Beispielcode unten: Nach Erstellung des ATL-Objekts müssen Sie nun die IDE Plug-In-Schnittstelle von XMLSpy implementieren.

```
import "oaidl.idl";
import "ocidl.idl";

// ----- please insert the following block into your IDL file -----
    typedef enum {
       spyEnable = 1,
       spyDisable = 2,
       spyCheck = 4,
       spyUncheck = 8
    } SPYUpdateAction;

// ----- end insert block ----


// ----- E.g. Interface entry automatically generated by the ATL wizard -----
```

```
//     [
//           object,
//           uuid(AB7CD86A-8145-429A-A1F3-270692EO8AFC),

//           helpstring("IXMLSpyPlugIn Interface")
//           pointer_default(unique)
//     ]
//     interface IXMLSpyPlugIn : IUnknown
//     {
//     };
// ----- end automatically generated Interface Entry


// ----- replace the Interface Entry (shown above) generated for you by the ATL wizard,
with the following block -----

       [
          odl,
          uuid(88F2A622-4B7E-42CD-8D04-3C0E5389DD85),
          helpstring("IXMLSpyPlugIn Interface")
       ]
       interface IXMLSpyPlugIn : IUnknown
          {
             HRESULT _stdcall OnCommand([in] long nID, [in] IDispatch* pXMLSpy);
             HRESULT _stdcall OnUpdateCommand([in] long nID, [in] IDispatch* pXMLSpy, [out,
retval] SPYUpdateAction* pAction);
             HRESULT _stdcall OnEvent([in] long nEventID, [in] SAFEARRAY(VARIANT)*
arrayParameters, [in] IDispatch* pXMLSpy, [out, retval] VARIANT* pReturnValue);
             HRESULT _stdcall GetUIModifications([out, retval] BSTR* pModificationsXML);
             HRESULT _stdcall GetDescription([out, retval] BSTR* pDescription);
          };

// ----- end replace block -----


// ----- The code below is automatically generated by the ATL wizard and will look slightly
different in your case -----

       [
          uuid(24FE0D1B-3FC0-494E-B36E-1D4CE412B014),
          version(1.0),
          helpstring("XMLSpyIDEPlugInDLL 1.0 Type Library")
       ]
       library XMLSPYIDEPLUGINDLLLib
          {
       importlib("stdole32.tlb");
       importlib("stdole2.tlb");

       [
          uuid(3800E791-7F6B-4ACD-9E32-2AC184444501),
          helpstring("XMLSpyIDEPlugIn Class")
       ]
       coclass XMLSpyIDEPlugIn
       {
          [default] interface IXMLSpyPlugIn;   // ----- define IXMLSpyPlugIn as the default
interface -----
```


© 2016-2022 Altova GmbH

```
        };
};
```

29.2.4.2 Klassendefinition

In der Klassendefinition des ATL-Objekts müssen die folgenden Änderungen vorgenommen werden.

- Die Klasse muss von `IXMLSpyPlugIn` abgeleitet sein.
- Die "Interface Map" benötigt einen Eintrag für `IXMLSpyPlugIn`.
- Die Methoden der IDE Plug-In-Schnittstelle müssen deklariert werden.

Diese Änderungen können, wie unten gezeigt, gemacht werden:

```
#ifndef __XMLSPYIDEPLUGIN_H_
#define __XMLSPYIDEPLUGIN_H_

#include "resource.h"       // main symbols

/////////////////////////////////////////////////////////////////////////////
// CXMLSpyIDEPlugIn
class ATL_NO_VTABLE CXMLSpyIDEPlugIn :
   public CComObjectRootEx<CComSingleThreadModel>,
   public CComCoClass<CXMLSpyIDEPlugIn, &CLSID_XMLSpyIDEPlugIn>,
   public IXMLSpyPlugIn
{
public:
   CXMLSpyIDEPlugIn()
   {
   }

DECLARE_REGISTRY_RESOURCEID(IDR_XMLSPYIDEPLUGIN)
DECLARE_NOT_AGGREGATABLE(CXMLSpyIDEPlugIn)

DECLARE_PROTECT_FINAL_CONSTRUCT()

BEGIN_COM_MAP(CXMLSpyIDEPlugIn)
   COM_INTERFACE_ENTRY(IXMLSpyPlugIn)
END_COM_MAP()

// IXMLSpyIDEPlugIn
public:
   virtual HRESULT _stdcall OnCommand(long nID, IDispatch* pXMLSpy);
   virtual HRESULT _stdcall OnUpdateCommand(long nID, IDispatch* pXMLSpy, SPYUpdateAction*
pAction);
   virtual HRESULT _stdcall OnEvent(long nEventID, SAFEARRAY **arrayParameters, IDispatch*
pXMLSpy, VARIANT* pReturnValue);
   virtual HRESULT _stdcall GetUIModifications(BSTR* pModificationsXML);
   virtual HRESULT _stdcall GetDescription(BSTR* pDescription);
};

#endif //__XMLSPYIDEPLUGIN_H_
```

29.2.4.3 Implementierung

Im Code unten sehen Sie eine einfache Implementierung eines XMLSpy IDE Plug-In. Damit wird ein Menübefehl und eine Trennlinie (verfügbar in XMLSpy) zum Menü "Extras" hinzugefügt. Der neue Befehl innerhalb der OnUpdateCommand() Methode ist nur aktiv, wenn das aktive Dokument über die Grid-Ansicht angezeigt wird. Der Befehl sucht nach dem XML-Element, das sich gerade im Fokus befindet, und öffnet jede URL, die mit "http://" beginnt am Textwert des Elements.

```
////////////////////////////////////////////////////////////////////////////
// CXMLSpyIDEPlugIn

#import "XMLSpy.tlb"
using namespace XMLSpyLib;

HRESULT CXMLSpyIDEPlugIn::OnCommand(long nID, IDispatch* pXMLSpy)
{
   USES_CONVERSION;

   if(nID == 1)      {
      IApplicationPtr       ipSpyApp;

      if(pXMLSpy)    {
         if(SUCCEEDED(pXMLSpy->QueryInterface(__uuidof(IApplication),(void **)&ipSpyApp)))
{
            IDocumentPtr  ipDocPtr = ipSpyApp->ActiveDocument;

            // we assume that grid view is active
            if(ipDocPtr) {
               IGridViewPtr   ipGridPtr = ipDocPtr->GridView;

               if(ipGridPtr)       {
                  IXMLDataPtr     ipXMLData = ipGridPtr->CurrentFocus;

                  CString   strValue = W2T(ipXMLData->TextValue);

                  if(!strValue.IsEmpty() && (strValue.Left(7) == _T("http://")))
                     ::ShellExecute(NULL,_T("open"),W2T(ipXMLData-
>TextValue),NULL,NULL,SW_SHOWNORMAL);
               }
            }
         }
      }
   }

   return S_OK;
}


HRESULT CXMLSpyIDEPlugIn::OnUpdateCommand(long nID, IDispatch* pXMLSpy, SPYUpdateAction*
pAction)
{
   *pAction = spyDisable;
```


© 2016-2022 Altova GmbH

```
   if(nID == 1)      {
      IApplicationPtr       ipSpyApp;

      if(pXMLSpy)    {
         if(SUCCEEDED(pXMLSpy->QueryInterface(__uuidof(IApplication),(void **)&ipSpyApp)))
       {
            IDocumentPtr    ipDocPtr = ipSpyApp->ActiveDocument;

            // only enable if grid view is active
            if((ipDocPtr != NULL) && (ipDocPtr->CurrentViewMode == spyViewGrid))
               *pAction = spyEnable;
         }
      }
   }

   return S_OK;
}


HRESULT CXMLSpyIDEPlugIn::OnEvent(long nEventID, SAFEARRAY **arrayParameters, IDispatch*
pXMLSpy, VARIANT* pReturnValue)
{
   return S_OK;
}

HRESULT CXMLSpyIDEPlugIn::GetUIModifications(BSTR* pModificationsXML)
{
   CComBSTR   bstrMods = _T(" \
            <ConfigurationData>    \
               <Modifications>    ");
   // add "Open URL..." to Tools menu
   bstrMods.Append (_T(" \
                  <Modification> \
                     <Action>Add</Action> \
                     <UIElement type=\"MenuItem\"> \
                        <ID>1</ID> \
                        <Name>Open URL...</Name> \
                        <Place>0</Place> \
                        <MenuID>129</MenuID> \
                        <Parent>:Tools</Parent> \
                     </UIElement> \
                  </Modification> "));
   // add Seperator to Tools menu
   bstrMods.Append (_T(" \
                  <Modification> \
                     <Action>Add</Action> \
                     <UIElement type=\"MenuItem\"> \
                        <ID>0</ID> \
                        <Place>1</Place> \
                        <MenuID>129</MenuID> \
                        <Parent>:Tools</Parent> \
                     </UIElement> \
                  </Modification> "));
   // finish modification description
   bstrMods.Append (_T(" \
```

```
            </Modifications> \
          </ConfigurationData>"));

   return bstrMods.CopyTo(pModificationsXML);
}


HRESULT CXMLSpyIDEPlugIn::GetDescription(BSTR* pDescription)
{
   CComBSTR   bstrDescr = _T("ATL C++ XMLSpy IDE PlugIn;This PlugIn demonstrates the
implementation of a simple ATL DLL as a IDE PlugIn for XMLSpy.");
   return bstrDescr.CopyTo(pDescription);
}
```

29.2.5 IXMLSpyPlugIn

Methoden

OnCommand 1700
OnUpdateCommand 1701
OnEvent 1702
GetUIModifications 1704
GetDescription 1704

Beschreibung

Wenn eine DLL als IDE Plug-In zu XMLSpy hinzugefügt wird, muss sie als COM-Komponente registriert werden, die auf eine `IXMLSpyPlugIn` Schnittstelle mit der reservierten `UUID(88F2A622-4B7E-42CD-8D04-3C0E5389DD85)` antwortet, damit sie als Plug-In erkannt wird.

29.2.5.1 OnCommand

Deklaration

```
OnCommand(nID als long, pXMLSpy als IDispatch)
```

Beschreibung

Die `OnCommand()` Methode der Schnittstellenimplementierung wird jedes Mal aufgerufen, wenn ein zum IDE Plug-In hinzugefügter Befehl (Menübefehl oder Symbolleisten-Schaltfläche) verarbeitet wird. `nID` speichert die durch das `ID`-Element des entsprechenden `UIElement` definierte Befehls-ID. `pXMLSpy` enthält eine Referenz auf das Dispatch Interface des `Application`-Objekts von XMLSpy.

Beispiel

```
Public Sub IXMLSpyPlugIn_OnCommand(ByVal nID As Long, ByVal pXMLSpy As Object)
   If (Not (pXMLSpy Is Nothing)) Then
      Dim objDlg
      Dim objDoc As XMLSpyLib.Document
      Dim objSpy As XMLSpyLib.Application
      Set objSpy = pXMLSpy

      If nID = 3 Or nID = 5 Then
         Set objDlg = CreateObject("MSComDlg.CommonDialog")
         objDlg.Filter = "XML Files (*.xml)|*.xml|All Files (*.*)|*.*||"
```


© 2016-2022 Altova GmbH

```
         objDlg.FilterIndex = 1
         objDlg.ShowOpen

         If Len(objDlg.FileName) > 0 Then
            Set objDoc = objSpy.Documents.OpenFile(objDlg.FileName, False)
            Set objDoc = Nothing
         End If
      End If

      If nID = 4 Or nID = 6 Then
         Set objDlg = CreateObject("MSComDlg.CommonDialog")
         objDlg.Filter = "All Files (*.*)|*.*||"
         objDlg.Flags = cdlOFNPathMustExist
         objDlg.ShowSave

         If Len(objDlg.FileName) > 0 Then
            Set objDoc = objSpy.ActiveDocument

            If Not (objDoc Is Nothing) Then
               objDoc.SetPathName objDlg.FileName
               objDoc.Save
               Set objDoc = Nothing
            End If
         End If
      End If

      Set objSpy = Nothing
   End If
End Sub
```

29.2.5.2 OnUpdateCommand

Deklaration

```
OnUpdateCommand(nID als long, pXMLSpy als IDispatch) als SPYUpdateAction
```

Beschreibung

Die `OnUpdateCommand()` Methode wird jedes Mal, wenn der Sichtbarkeitsstatus einer Schaltfläche oder eines Menübefehls definiert werden muss, aufgerufen. `nID` speichert die durch das `ID`-Element des entsprechenden `UIElement` definierte Befehls-ID. `pXMLSpy` enthält eine Referenz auf das Dispatch Interface des `Application`-Objekts.

Mögliche Rückgabewerte zum Definieren des Aktualisierungsstatus sind:

```
spyEnable= 1
spyDisable   = 2
spyCheck     = 4
spyUncheck   = 8
```

Beispiel

```
Public Function IXMLSpyPlugIn_OnUpdateCommand(ByVal nID As Long, ByVal pXMLSpy As Object)
As SPYUpdateAction
   IXMLSpyPlugIn_OnUpdateCommand = spyDisable

   If (Not (pXMLSpy Is Nothing)) Then
```

```
      Dim objSpy As XMLSpyLib.Application
      Set objSpy = pXMLSpy

      If nID = 3 Or nID = 5 Then
         IXMLSpyPlugIn_OnUpdateCommand = spyEnable
      End If
      If nID = 4 Or nID = 6 Then
         If objSpy.Documents.Count > 0 Then
            IXMLSpyPlugIn_OnUpdateCommand = spyEnable
         Else
            IXMLSpyPlugIn_OnUpdateCommand = spyDisable
         End If
      End If
   End If
End Function
```

29.2.5.3 OnEvent

Deklaration

```
OnEvent(nEventID als long, arrayParameters als SAFEARRAY(VARIANT), pXMLSpy als IDispatch)
als VARIANT
```

Beschreibung

`OnEvent()` wird jedes Mal, wenn ein Event von XMLSpy ausgelöst wird, aufgerufen.

Mögliche Werte für `nEventID` sind:

```
      On_BeforeStartEditing              = 1
      On_EditingFinished                 = 2
      On_FocusChanged                    = 3
      On_Beforedrag                      = 4
      On_BeforeDrop                      = 5
      On_OpenProject                     = 6
      On_OpenDocument                    = 7
      On_CloseDocument                   = 8
      On_SaveDocument                    = 9
      On_DocEditDragOver                 = 10
      On_DocEditDrop                     = 11
      On_DocEditKeyDown                  = 12
      On_DocEditKeyUp                    = 13
      On_DocEditKeyPressed               = 14
      On_DocEditMouseMove                = 15
      On_DocEditButtonUp                 = 16
      On_DocEditButtonDown               = 17
      On_DocEditContextMenu              = 18
```


© 2016-2022 Altova GmbH

```
On_DocEditPaste                    = 19
On_DocEditCut                      = 20
On_DocEditCopy                     = 21
On_DocEditClear                    = 22
On_DocEditSelectionChanged         = 23
On_DocEditDragOver                 = 10
On_BeforeOpenProject               = 25
On_BeforeOpenDocument              = 26
On_BeforeSaveDocument              = 27
On_BeforeCloseDocument             = 28
On_ViewActivation                  = 29
On_DocEditKeyboardEvent            = 30
On_DocEditMouseEvent               = 31
On_BeforeValidate                  = 32
On_BeforeShowSuggestions           = 33
On_ProjectOpened                   = 34
On_Char                            = 35
On_Initialize                      = 36
On_Running                         = 37
On_Shutdown                        = 38
On_AuthenticBeforeSave             = 39
On_AuthenticContextMenuActivated   = 40
On_AuthenticLoad                   = 41
On_AuthenticToolbarButtonClicked   = 42
On_AuthenticToolbarButtonExecuted  = 43
On_AuthenticUserAddedXMLNode       = 44
```

Die Namen der Events sind dieselben wie in der Skripting-Umgebung von XMLSpy. Bei IDE Plug-Ins spielt es keine Rolle, welche Namen verwendet werden. Die Events werden über den ID Wert definiert.

`arrayParameters` ist ein Array, das mit den Parametern des aktuell ausgelösten Event gefüllt ist. Die Reihenfolge, Art und Bedeutung der einzelnen Parameter stehen über die Skripting-Umgebung von XMLSpy zur Verfügung. Das Event-Modul eines Skripting-Projekts enthält vordefinierte Funktionen für alle Events vor Version 4.4. Die an die vordefinierten Funktionen übergebenen Parameter sind mit den Array-Elementen des `arrayParameters`-Parameters identisch.

Events, die über die Authentic-Ansicht von XMLSpy ausgelöst werden, übergeben Parameter nicht direkt. Statt dessen wird ein "event" Objekt verwendet. Das Event-Objekt kann über das `Document`-Objekt des aktiven Dokuments aufgerufen werden.

`pXMLSpy` enthält eine Referenz auf das Dispatch Interface des `Application`-Objekts von XMLSpy.

Wenn der Rückgabewert von `OnEvent()` definiert ist, so ruft weder das IDE Plug-In noch ein Event Handler innerhalb der Skripting-Umgebung dieses Event später auf. Bitte beachten Sie, dass alle IDE Plug-Ins das Event abrufen/verarbeiten, bevor dies die Skripting-Umgebung tut.

29.2.5.4 GetUIModifications

Deklaration

```
GetUIModifications() als String
```

Beschreibung

Die `GetUIModifications()` Methode wird während der Initialisierung des Plug-In aufgerufen, um die XML-Konfigurationsdaten abzurufen, die die Änderungen an der Benutzeroberfläche von XMLSpy definieren. Die Methode wird beim ersten Laden des Plug-In und bei jedem Start von XMLSpy aufgerufen. *Eine ausführliche Anleitung zum Ändern der Benutzeroberfläche finden Sie auch unter Konfigurations-XML* 1691.

Beispiel

```
Public Function IXMLSpyPlugIn_GetUIModifications() As String
   ' GetUIModifications() gets the XML file with the specified modifications of
   ' the UI from the config.xml file in the plug-in folder
   Dim strPath As String
   strPath = App.Path

   If Len(strPath) > 0 Then
      Dim fso As New FileSystemObject
      Dim file As file
      Set file = fso.GetFile(strPath & "\config.xml")

      If (Not (file Is Nothing)) Then
         Dim stream As TextStream
         Set stream = file.OpenAsTextStream(ForReading)

         ' this replaces the token '**path**' from the XML file with
         ' the actual installation path of the plug-in to get the image file
         Dim strMods As String
         strMods = stream.ReadAll
         strMods = Replace(strMods, "**path**", strPath)

         IXMLSpyPlugIn_GetUIModifications = strMods
      Else
         IXMLSpyPlugIn_GetUIModifications = ""
      End If
   End If
End Function
```

29.2.5.5 GetDescription

Deklaration

```
GetDescription() as String
```


© 2016-2022 Altova GmbH

Beschreibung

`GetDescription()` dient zur Definition des Beschreibungsstring für die Plug-In-Einträge, die im Dialogfeld "Anpassen" angezeigt werden.

Beispiel

```
Public Function IXMLSpyPlugIn_GetDescription() As String
    IXMLSpyPlugIn_GetDescription = "Sample Plug-in for XMLSpy;This Plug-in demonstrates the
implementation of a simple VisualBasic DLL as a Plug-in for XMLSpy."
End Function
```

29.3 Applikations-API

Mit Hilfe der COM-basierten API von XMLSpy können andere Applikationen die Funktionalitäten von XMLSpy verwenden. Dadurch können die verschiedensten Aufgaben vom Validieren einer XML-Datei bis zum Ändern von komplexem XML-Inhalt (mit der XMLData 1970 Schnittstelle) automatisiert werden.

XMLSpy und die Applikations-API entsprechen den von Microsoft vorgegebenen allgemeinen Spezifikationen für Automation Server. Die Methoden und Eigenschaften der Applikations-API können von gängigen Entwicklungsumgebungen wie z.B. solchen, die C#, C++, VisualBasic und Delphi verwenden, und von von Skript-Sprachen wie JScript und VBScript aufgerufen werden.

Ausführungsumgebungen für die Applikations-API

Die Applikations-API kann über die folgenden Ausführungsumgebungen aufgerufen werden:

- externe Programme (Beschreibung siehe unten 1706 und in diesem Abschnitt im Kapitel Übersicht 1707)
- vom integrierten Skript-Editor von XMLSpy aus. Eine Beschreibung zur Skripting-Umgebung finden Sie im Abschnitt Skript-Editor 1662.
- Sie haben in XMLSpy die Möglichkeit, Ihre eigenen Plug-ins zu erstellen und über eine spezielle Schnittstelle für Plug-ins in die Applikation zu integrieren. Eine Beschreibung zur Erstellung von Plug-ins finden Sie im Abschnitt IDE Plug-ins 1690.
- über ein ActiveX Control, das verfügbar ist, wenn das Integration Package 2073 installiert wurde. Nähere Informationen dazu finden Sie im Abschnitt ActiveX Integration 2073.

Externe Programme

Im Abschnitt Übersicht 1707 dieses Abschnitts wird beschrieben, wie die Funktionalitäten von XMLSpy von externen Programmen aus aufgerufen und automatisiert werden können.

Um die Applikations-API außerhalb von XMLSpy verwenden zu können, muss zuerst eine Instanz von XMLSpy gestartet werden. Wie dies erfolgt, hängt von der verwendeten Programmiersprache ab. Nähere Informationen zu den einzelnen Sprachen finden Sie unter Programmiersprachen 1708.

Im Wesentlichen wird XMLSpy über seine COM Registrierung gestartet. Anschließend wird das mit der XMLSpy-Instanz verknüpfte `Application` Objekt zurückgegeben. Je nach COM-Einstellungen kann ein mit einer bereits laufenden XMLSpy-Instanz verknüpftes Objekt zurückgegeben werden. Es kann jede Programmiersprache verwendet werden, die die Erstellung und den Aufruf von COM-Objekten unterstützt. Die am häufigsten dazu verwendeten Sprachen sind unten aufgelistet.

- JScript und VBScript 1719 Skript-Dateien haben eine einfache Syntax und dienen dazu COM-Objekte aufzurufen. Sie können direkt über eine DOS-Befehlszeile oder durch Doppelklick im Windows Explorer ausgeführt werden. Am besten eignen sich diese Sprachen für einfache Automationsaufgaben.
- C# 1722 ist eine umfangreiche Programmiersprache mit einer breiten Palette an Funktionalitäten. Der Zugriff auf COM-Objekte kann in C# automatisch über Wrapper erfolgen.
- C++ ermöglicht die direkte Steuerung des COM-Zugriffs. Allerdings muss dafür im Vergleich zu anderen Sprachen relativ viel Code geschrieben werden.
- Java 1732: Im Lieferumfang von Altova-Produkten sind native Java-Klassen inkludiert, die als Wrapper für die Applikations-API verwendet werden und dadurch eine Java-Umgebung ermöglichen.
- Weitere nützliche Alternativen sind Visual Basic für Applikationen, Perl und Python.

© 2016-2022 Altova GmbH

Anmerkungen zur Programmierung

Berücksichtigen Sie bitte in Ihrem Client-Code die folgenden Einschränkungen:

- Denken Sie daran, dass bei einem Crash Ihres Client-Code eventuell noch Instanzen von XMLSpyim System laufen.
- Behalten Sie Referenzen auf Objekte - vor allem jene von der XMLData-Schnittstelle - nicht länger als nötig im Arbeitsspeicher. Wenn der Benutzer zwischen zwei Aufrufen Ihres Client Eingaben macht, ist nicht gewährleistet, dass diese Referenzen noch gültig sind.
- Vergessen Sie nicht, Dialogfelder zu deaktivieren, wenn die Benutzeroberfläche nicht sichtbar ist.
- Nähere Informationen, wie Sie lästige Fehlermeldungen vermeiden, finden Sie unter Fehlerbehandlung in JScript[1713] (und in C#[1730] and Java[1739]).
- Geben Sie Referenzen explizit frei, wenn Sie C oder C++ verwenden.

Diese Dokumentation

Die Dokumentation zur Applikations-API ist in zwei Hauptteile gegliedert.

- Der erste Teile besteht aus einer Übersicht[1707], in der das Objektmodell für die API beschrieben wird. Weiters wird beschrieben, wie Sie die API über verschiedene Programmiersprachen[1708] aufrufen.
- Der zweite Teil ist ein Referenzteil (Schnittstellen[1745] und Enumerationen[2020]), der Beschreibungen der Schnittstellenobjekte der Applikations-APIenthält

29.3.1 Übersicht

Diese Übersicht über die Applikations-API ist in die folgenden Abschnitte gegliedert:

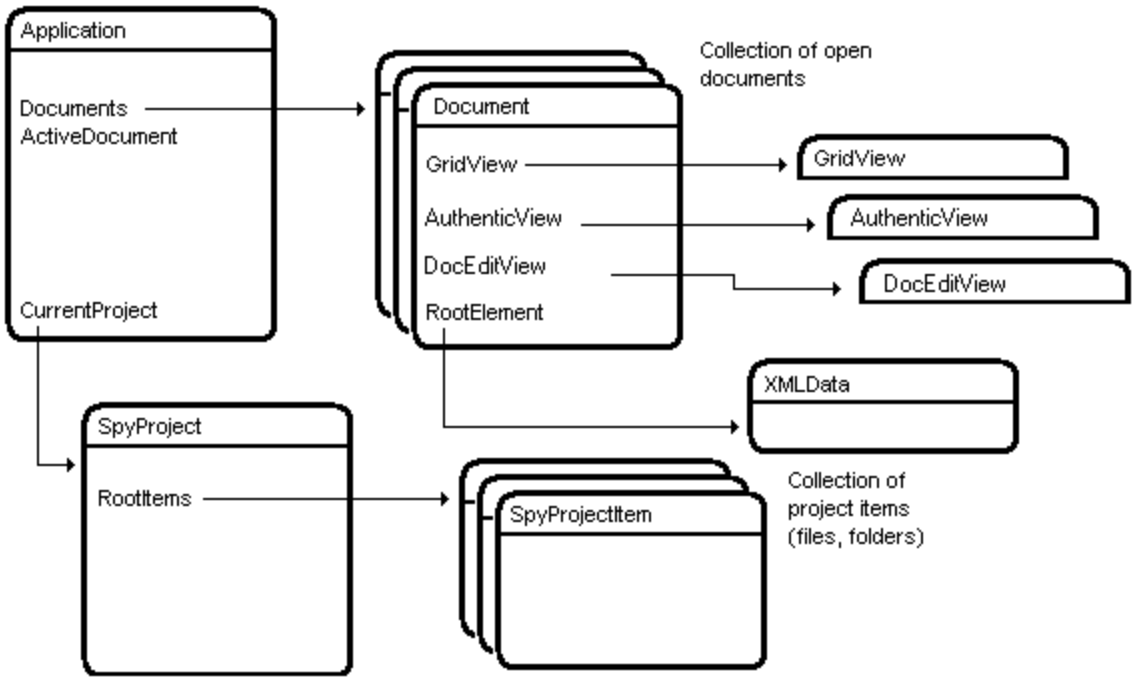
- Das Objektmodell[1707]: Hier werden die Beziehungen zwischen den Objekten der Applikations-API beschrieben.
- Programmiersprachen[1708]: Hier wird beschrieben, wie Sie die Funktionalitäten der Applikations-API über die gebräuchlichsten Programmiersprachen (JScript, VBScript, C# und Java) aufrufen können. Die Codefragmente aus den mitinstallierten Beispieldateien dienen zum Veranschaulichen einiger grundlegender Mechanismen.
- DOM und XMLData[1740]: Hier wird die Beziehung zwischen der `XMLData`-Schnittstelle der Applikations-API und der DOM erläutert.
- Nicht mehr verwendete Zeilenoperationen in der Authentic-Ansicht[1743]: Hier finden Sie Informationen zu nicht mehr verwendeten Objekten für Tabellenzeilenoperationen in der Authentic-Ansicht.
- Nicht mehr verwendete Bearbeitungsoptionen in der Authentic-Ansicht[1744]: Hier finden Sie Informationen zu nicht mehr verwendeten Objekten für Bearbeitungsoperationen in der Authentic-Ansicht.

29.3.1.1 Objektmodell

Der Ausgangspunkt für alle Applikationen, die die Applikations-API verwenden, ist das `Application`[1746] Objekt. Dieses Objekt enthält allgemeine Methoden wie Import/Export-Funktionen und Referenzen auf offene Dokumente und das offene Projekt, falls vorhanden.

Das `Application`-Objekt wird in verschiedenen Programmiersprachen unterschiedlich erstellt. In Skriptsprachen wie JScript oder VBScript wird dabei eine Funktion aufgerufen, die das COM-Objekt der Applikation initialisiert. Beispiele dazu finden Sie im Abschnitt Programmiersprachen[1708].

In der Abbildung unten sehen Sie die Verknüpfungen zwischen den Hauptobjekten der Applikations-API:



Das Application-Objekt besteht aus den folgenden Teilen:

1. Dokumentsammlung und Referenz auf das aktive Dokument.
2. Referenz auf das aktuelle Projekt und Methoden zum Erstellen und Öffnen von Projekten.
3. Methoden zur Unterstützung des Exports und Imports aus/in Datenbanken, Textdateien und Word-Dokumente.
4. URL-Verwaltung.
5. Methoden für Makro-Menübefehle.

Nach Erstellung eines Application-Objekts können Sie beginnen, die Funktionalitäten von XMLSpy zu nutzen. In den meisten Fällen werden Sie entweder ein Projekt öffnen und die Dokumente von dort aufrufen oder ein Dokument direkt über die Documents[1870] Schnittstelle öffnen.

29.3.1.2 Programmiersprachen

Der Zugriff auf die COM-Schnittstelle erfolgt in den verschiedenen Programmiersprachen auf unterschiedliche Weise. Eine Beschreibung dazu finden Sie anhand einiger Beispiele für die gebräuchlichsten Programmiersprachen (*Links siehe unten*). Anhand der Codefragmente in diesem Abschnitt wird gezeigt, wie grundlegende Funktionalitäten aufgerufen werden. Sie können diese Funktionalitäten anhand der Dateien im Unterordner "API Examples" des Ordners `Examples` testen:

© 2016-2022 Altova GmbH

Windows 7, Windows 8, Windows 10, Windows 11	`C:\Benutzer\<Benutzername>\Dokumente\Altova\XMLSpy\2023\Examples`

JScript

Im JScript-Code werden die folgenden grundlegenden Funktionen verwendet:

- Starten der Applikation oder Anhängen einer laufenden Instanz[1710]
- Einfacher Dokumentzugriff[1712]
- Iteration[1712]
- Fehlerbehandlung[1713]
- Events[1714]
- Import und Export von Daten[1715]

VBScript

VBScript unterscheidet sich nur syntaktisch von JScript und funktioniert ansonsten auf genau die gleiche Art. Über den nachstehenden Link gelangen Sie zu einem Beispiel für die Verwendung von VBScript. Nähere Informationen finden Sie in den JScript-Beispielen[1710].

- Events[1719]: Zeigt, wie Events mit VBScript verwendet werden.

C#

Sie können mit Hilfe von C# auf die Funktionalitäten der Application API zugreifen. Der unten aufgelistete Code enthält ein Beispiel dafür, wie Sie die API für bestimmte grundlegende Funktionalitäten aufrufen können.

- Start XMLSpy[1727]: Startet XMLSpy, das als Automation Server registriert ist, oder aktiviert das Programm, wenn XMLSpy bereits ausgeführt wird.
- Open OrgChart.pxf[1728]: Navigiert zu einem der mit XMLSpy installierten Beispieldokumente und öffnet es. Wenn das Dokument bereits offen ist, wird es zum aktiven Dokument.
- OnDocumentOpened Event On/Off[1731]: Zeigt, wie die Applikation auf XMLSpy Events hört. Ist die Funktion aktiv, erscheint ein Meldungsfeld, nachdem ein Dokument geöffnet wurde.
- Open ExpReport.xml[1728]: Öffnet ein weiteres Beispieldokument.
- Toggle View Mode[1729]: Wechselt in der Ansicht aller offenen Dokumente zwischen der Text- und der Authentic-Ansicht. Im Code wird gezeigt, wie Sie durch offene Dokumente iterieren können.
- Validate[1730]: Validiert das aktive Dokument und zeigt das Ergebnis in einem Meldungsfeld an. Im Code wird gezeigt, wie Fehler und COM-Ausgabeparameter behandelt werden.
- Shutdown XMLSpy[1727]: Beendet XMLSpy.

Java

Sie können mit Hilfe von Java auf die Funktionalitäten der Application API zugreifen. Im Unterabschnitt "Java" dieses Abschnitts[1732] wird erklärt, wie einige grundlegende XMLSpy-Funktionen über Java-Code aufgerufen werden können. Der Abschnitt ist in die folgenden Unterabschnitte gegliedert:

- Mapping-Regeln für den Java Wrapper[1732]
- Java-Beispielprojekt[1733]

- Starten und Beenden der Applikation 1737
- Einfacher Dokumentaufruf 1738
- Iterationen 1738
- Verwendung von Out-Parametern 1739
- Event Handler 1740

29.3.1.2.1 JScript

Dieser Abschnitt enthält eine Liste von JScript-Code, anhand dessen die folgenden grundlegenden Funktionalitäten erläutert werden:

- Starten der Applikation oder Anhängen einer laufenden Instanz 1710
- Einfacher Dokumentzugriff 1712
- Iteration 1712
- Fehlerbehandlung 1713
- Events 1714
- Import und Export von Daten 1715

Beispieldateien

Sie finden den Code in diesem Abschnitt in den Beispieldateien, die Sie je nach Bedarf testen und modifizieren können. Die JScript-Beispieldateien befinden sich im Ordner `JScript` des Ordners API Examples:

Windows 7, Windows 8, Windows 10, Windows 11	`C:\Benutzer\<Benutzername>\Dokumente\Altova\XMLSpy\2023\Examples`

Die Beispieldateien können auf zwei Arten ausgeführt werden:

- *Über die Befehlszeile:* Öffnen Sie ein Eingabeaufforderungsfenster, wechseln Sie zum obigen Verzeichnis und geben Sie den Namen eines der Beispiel-Skripts ein (z.B. `Start.js`).
- *Über den Windows Explorer:* Navigieren Sie im Windows Explorer zur JScript-Datei und doppelklicken Sie darauf.

Der mit dem Windows-Betriebssystem verfügbare Windows Script Host führt das Skript aus. Nähere Informationen zum Windows Script Host finden Sie in der MSDN-Dokumentation (https://msdn.microsoft.com).

29.3.1.2.1.1 Applikation starten

Mit dem unten aufgeführten JScript-Code wird die Applikation gestartet und beendet. Wenn das COM-Objekt der 32-Bit-Version von XMLSpy nicht gefunden wird, wird nach dem COM-Objekt der 64-Bit-Applikation gesucht; andernfalls wird ein Fehler ausgegeben. Wenn bereits eine Instanz der Applikation ausgeführt wird, so wird diese Instanz aufgerufen.

Anmerkung: Für die 32-Bit-Version von XMLSpy ist der registrierte Name oder der programmatische Identifier (ProgId) des COM-Objekts `XMLSpy.Application`. Für die 64-Bit-Version von XMLSpy ist der

 © 2016-2022 Altova GmbH

Name `XMLSpy_x64.Application`. Beachten Sie jedoch, dass das aufrufende Programm die CLASSES Registry-Einträge in seiner eigenen Registry Hive oder -Gruppe (32-Bit oder 64-Bit) aufruft. Wenn Sie daher Skripts über die Standardbefehlszeileneingabe und mit Windows Explorer auf einem 64-Bit-Windows-System ausführen, werden die 64-Bit-Registry-Einträge, welche auf die 64-Bit-Version von XMLSpy verweisen, aufgerufen. Wenn daher sowohl XMLSpy 32-Bit als auch die 64-Bit-Version installiert ist, ist eine spezielle Behandlung erforderlich, damit die 32-Bit-Version von XMLSpy aufgerufen wird. Angenommen, der Windows Skripting Host ist das aufrufende Programm, so gehen Sie folgendermaßen vor:

1. Wechseln Sie in das Verzeichnis **C:\Windows\SysWOW64**.
2. Geben Sie in der Befehlszeile **wscript.exe** gefolgt vom Pfad zum gewünschten Skript ein, z.B:

```
wscript.exe "C:\Users\...
\Documents\Altova\StyleVision2023\StyleVisionExamples\API\JScript\start.js"wscript.exe
"C:\Users\...\Documents\Altova\XMLSpy2023\Examples\API\JScript\start.js"
```

```
// Initialize application's COM object. This will start a new instance of the application
and
// return its main COM object. Depending on COM settings, the main COM object of an
already
// running application might be returned.

try{   objSpy = WScript.GetObject("", "XMLSpy.Application");   }
catch(err) {}

if( typeof( objSpy ) == "undefined")
{
  try  {   objSpy = WScript.GetObject("", "XMLSpy_x64.Application")   }
  catch(err)
  {
     WScript.Echo( "Can't access or create XMLSpy.Application");
     WScript.Quit();
  }
}

// if newly started, the application will start without its UI visible. Set it to
visible.
objSpy.Visible = true;

WScript.Echo(objSpy.Edition + " has successfully started. ");

objSpy.Visible = false;      // will shutdown application if it has no more COM
connections
//objSpy.Visible = true;   // will keep application running with UI visible
```

Der oben angeführte JScript-Code steht in der Datei `Start.js` zur Verfügung (siehe Beispieldateien[1710]).

29.3.1.2.1.2 Einfacher Dokumentaufruf

Nachdem Sie die Applikation, wie unter Applikation starten[1710] beschrieben, gestartet haben, werden Sie wahrscheinlich ein Dokument programmatisch öffnen wollen, um damit zu arbeiten. Im nachstehenden JScript-Codefragment sehen Sie, wie Sie zwei Dokumente aus dem XMLSpy-Beispielordner öffnen und eines davon als aktives Dokument definieren.

```
// Locate examples via USERPROFILE shell variable. The path needs to be adapted to major
release versions.
objWshShell = WScript.CreateObject("WScript.Shell");
majorVersionYear = objSpy.MajorVersion + 1998
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\My
Documents\\Altova\\XMLSpy"+ majorVersionYear + "\\Examples\\";

// Tell XMLSpy to open two documents. No dialogs
objDoc1 = objSpy.Documents.OpenFile(strExampleFolder + "OrgChart.pxf", false);
objSpy.Documents.OpenFile(strExampleFolder + "ExpReport.xml", false);

// The document currently active can be easily located.
objDoc2 = objSpy.ActiveDocument;

// Let us make sure that the document is shown in grid view.
objDoc2.SwitchViewMode(0);   // SPYViewModes.spyViewGrid = 0

// Now switch back to the document opened first
objDoc1.SetActiveDocument();
```

Sie finden den oben aufgelisteten JScript-Code in der Beispieldatei `DocumentAccess.js` (siehe Beispieldateien[1710]).

29.3.1.2.1.3 Iteration

Im unten aufgelisteten JScript-Code wird gezeigt, wie man durch offene Dokumente iteriert. Es wird davon ausgegangen, dass Sie die Applikation bereits gestartet und einige Dokumente, wie in den vorherigen Abschnitten beschrieben, geöffnet haben.

```
// go through all open documents using a JScript Enumerator
bRequiresSaving = false;
for(variterDocs = newEnumerator(objSpy.Documents); !iterDocs.atEnd();
iterDocs.moveNext())
{
  if(iterDocs.item().IsModified)
     bRequiresSaving = true;

  varstrErrorText = newArray(1);
  varnErrorNumber = newArray(1);
  varerrorData = newArray(1);
```


© 2016-2022 Altova GmbH

```
  if(!iterDocs.item().IsValid(strErrorText, nErrorNumber, errorData))
  {
      vartext = strErrorText;
     // access that XMLData object only if filled in
      if(errorData[0] != null)
        text += "("+ errorData[0].Name + "/"+ errorData[0].TextValue + ")";

     WScript.Echo("Document \""+ iterDocs.item().Name +"\" validation error["+
nErrorNumber + "]: "+ text);
  }
  else
  {
     // The COM call succeeded and the document is valid.
     WScript.Echo("Document \""+ iterDocs.item().Name + "\" is valid.");
  }
}

// go through all open documents using index-based access to the document collection
for(i = objSpy.Documents.Count; i > 0; i--)
  objSpy.Documents.Item(i).Close(false);
```

Sie finden den oben aufgelisteten JScript-Code in der Beispieldatei `DocumentAccess.js` (siehe Beispieldateien[1710]).

29.3.1.2.1.4 Fehlerbehandlung

Die Applikations-API gibt Fehler auf zwei verschiedene Arten zurück:

- das von jeder API-Methode zurückgegebene `HRESULT`
- die `IErrorInfo`-Schnittstelle der Applikations-API

Jede API-Methode gibt ein `HRESULT` zurück. Diese Rückgabewert enthält Informationen über Fehler, die während der Ausführung der Methode aufgetreten sind. Wenn der Aufruf erfolgreich war, so ist der Rückgabewert `S_OK`. Die Option `HRESULT` wird häufig in C/C++ Programmen verwendet.

Programmiersprachen wie VisualBasic sowie Skripting-Sprachen und andere Entwicklungsumgebungen gestatten dem Programmierer jedoch keinen Zugriff auf das zurückgegebene HRESULT eines COM-Aufrufs. Sie verwenden die IErrorInfo Schnittstelle, die ebenfalls von der Applikations-API unterstützt wird. Bei Auftreten eines Fehlers erstellt die API ein neues Objekt, das die IErrorInfo Schnittstelle implementiert. Die aus der IErrorInfo Schnittstelle stammenden Informationen werden in die Entwicklungsumgebung in ihren eigenen Fehlerbehandlungsmechanismus importiert.

So wird etwa mit dem nachstehenden JScript-Code ein Fehler zurückgegeben, weil ein Array falsch deklariert wurde. Weitere Informationen zum Fehler-Objekt stehen in seinen Eigenschaften `number` und `description` zur Verfügung.

```
try{
  vararr = newArray(-1);
}
```

```
catch(err) {
  WScript.Echo("Error : ("+ (err.number & 0xffff) + ")"+ err.description);
}
```

29.3.1.2.1.5 Events

COM definiert einen Client, der sich über den Connection Point-Mechanismus als Server für Callbacks registrieren muss. Das Automation Interface für XMLSpy definiert die dafür erforderlichen Event Interfaces. Wie eine Verbindung zu diesen Events hergestellt wird, hängt von der Programmiersprache ab, die Sie auf Ihrem Client verwenden. Im folgenden Codefragment wird gezeigt, wie dies in JScript erfolgt.

Die Methode `WScript.ConnectObject` dient zum Empfang von Events.

```
// The event-handler function
functionDocEvent_OnBeforeCloseDocument(objDocument)
{
  WScript.Echo("Received event - before closing document");
}

// Create or connect to XMLSpy (or Authentic Desktop)
try
{
  // Create the environment and XMLSpy (or Authentic Desktop)
  objWshShell = WScript.CreateObject("WScript.Shell");
  objFSO = WScript.CreateObject("Scripting.FileSystemObject");
  objSpy = WScript.GetObject("", "XMLSpy.Application");

// If only Authentic Desktop is installed (and XMLSpy is not installed) use:
// objSpy = WScript.GetObject("", "AuthenticDesktop.Application")

}
catch(err)
  { WScript.Echo ("Can't create WScript.Shell object or XMLSpy"); }

// Create document object and connect to its events
objSpy.Visible = true;
majorVersionYear = objSpy.MajorVersion + 1998
docPath = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\Documents\\Altova\
\XMLSpy"+ majorVersionYear + "\\Examples\\ExpReport.xml";
objDoc = objSpy.Documents.OpenFile (docPath, false);
WScript.ConnectObject(objDoc, "DocEvent_");

// Keep running while waiting for the event
// In the meanwhile close this document in XMLSpy (or Authentic Desktop) manually
WScript.Echo ("Sleeping for 10 seconds ...");
WScript.Sleep (10000);

objDoc = null;
WScript.Echo ("Stopped listening for event");
objSpy.Quit();
```


© 2016-2022 Altova GmbH

29.3.1.2.1.6 Import und Export von Daten

Bevor Sie Ihre Import- und Exportaufgaben mit der Applikations-API implementieren, ist es sinnvoll, die Verbindungen, Parameter und SQL-Abfragen usw. in XMLSpy zu testen. Auf diese Art können Sie die Ergebnisse überprüfen und eventuell Anpassungen an allen Import- und Export-Parametern vornehmen. Die meisten der Methoden für das Importieren und Exportieren von Daten werden in das `Application`[1746] Objekt platziert. Die restlichen Funktionen können über die `Document`[1838] Schnittstelle aufgerufen werden.

Bevor Sie mit dem eigentlichen Import und Export beginnen können, müssen zuerst einige Vorbereitungen getroffen werden. Jede Import/Export-Aufgabe besteht aus zwei Teilen. Sie müssen eine Verbindung zu Ihren Daten und das entsprechende Verhalten für den Import/Export-Vorgang definieren. Bei einem Import wird eine Verbindung zu einer Datenbank, einer Textdatei oder einem Word-Dokument hergestellt. Definiert wird im Grunde, welche Daten (Spalten) in XMLSpy importiert werden sollen. Bei einem Export wird eine Verbindung zu einer Datenbank oder einer Textdatei hergestellt. Geben Sie an, welche Daten (Elemente der XML-Datei) und welche zusätzlichen Parameter (z.B. automatische Schlüsselgenerierung oder Anzahl der untergeordneten Ebenen) aus der XML-Struktur verwendet werden sollen.

Die Eigenschaften in den Schnittstellen `DatabaseConnection`, `TextImportExportSettings` und `ExportSettings` haben Standardwerte. Nähere Informationen dazu finden Sie in den entsprechenden Beschreibungen im Kapitel `Schnittstellen`[1745].

In den Unterabschnitten dieses Abschnitts werden die einzelnen Operationen ausführlich beschrieben.

- Import aus einer Datenbank[1715]
- Export in eine Datenbank[1716]
- Import aus einer Textdatei[1717]
- Export in eine Textdatei[1718]

Dies sind die Schritte, um für den Import eine Verbindung zu einer vorhandenen Datenbank herzustellen:

1. Verwenden Sie ein `DatabaseConnection`[1828] Objekt und definieren Sie die Eigenschaften:
   - Die Methode `Application.GetDatabaseSettings`[1753] gibt ein neues Objekt für eine Datenbankverbindung zurück: `objImpSettings = objSpy.GetDatabaseSettings();`
   - Sie müssen entweder einen ADO Connection String definieren, `objImpSettings.ADOConnection = strADOConnection` oder den **Pfad** zu einer bestehenden Datenbankdatei: `objImpSettings.File = strExampleFolder + "Tutorial\\Company.mdb";`
   - Abschließend erstellen Sie eine SQL SELECT-Anweisung, um festzulegen, welche Daten abgefragt werden sollen: `objImpSettings.SQLSelect = "SELECT * FROM Address";`
2. Rufen Sie `Application.GetDatabaseImportElementList`[1753] auf, um eine Sammlung der Ergebnisspalten der SQL-Abfrage zu erhalten: `objElementList = objSpy.GetDatabaseImportElementList(objImpSettings);` Mit Hilfe dieser Sammlung können Sie festlegen, welche Spalten importiert werden sollen und welchen Typ die neuen Elemente haben sollen. Jedes Element der Sammlung steht für eine zu importierende Spalte. Wenn Sie ein Element entfernen, wird die entsprechenden Spalte nicht importiert. Zusätzlich dazu können Sie die Eigenschaft `ElementListItem.ElementKind`[1879] ändern, um den Typ der erstellten XML-Elemente für die einzelnen Spalten zu definieren.Bitte beachten Sie, dass `GetDatabaseImportElementList()` die SQL-Abfrage ausführt und einen zeitaufwändigen Aufruf initiieren könnte. Um dies zu vermeiden, können Sie einen null-pointer als zweiten Parameter an `ImportFromDatabase()` übergeben, um alle Spalten als einfache XML-Elemente zu importieren.
3. Starten Sie den Import mit `Application.ImportFromDatabase`[1757]: `objImpDocFromDB = objSpy.ImportFromDatabase(objImpSettings,objElementList);`

```
// Locate examples via USERPROFILE shell variable.
objWshShell = WScript.CreateObject("WScript.Shell");
majorVersionYear = objSpy.MajorVersion + 1998
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\My
Documents\\Altova\\XMLSpy"+ majorVersionYear + "\\Examples\\";

try
{
  // specify the source of data import
  objImpSettings = objSpy.GetDatabaseSettings();
  objImpSettings.File = strExampleFolder + "Tutorial\\Company.mdb";
  objImpSettings.SQLSelect = "SELECT * FROM Address";

  // column filter
  objElementList = objSpy.GetDatabaseImportElementList(objImpSettings);

  // import into a new XML file
  objImpDocFromDB = objSpy.ImportFromDatabase(objImpSettings,objElementList);
}
catch(err)
{
  WScript.Echo("Error importing from database.\n\n"+
                    "Error: "+ (err.number & 0xffff) + "\n"+
                    "Description: "+ err.description);
}
```

Sie finden den oben aufgelisteten JScript-Code in der Beispieldatei `ImportExport.js` (siehe Beispieldateien[1710]).

Um Daten in eine Datenbank zu exportieren, gehen Sie folgendermaßen vor:

1. Verwenden Sie ein `DatabaseConnection`[1828] Objekt und definieren Sie die erforderlichen Eigenschaften. Alle Eigenschaften mit Ausnahme von SQLSelect sind für den Export relevant. `ADOConnection` oder `File` definiert das Ziel für die Ausgabe. Sie müssen nur eine davon definieren.
2. Füllen Sie ein `ExportSettings`[1880] Objekt mit den erforderlichen Werten. Diese Eigenschaften sind dieselben Optionen wie die im Exportdialogfeld von XMLSpy. Wählen Sie die Menüoption **Konvertieren | Exportieren in Textdatei/Datenbank** um die Optionen zu sehen und eine Kombination von Exporteinstellungen auszuprobieren. Anschließend können Sie diese Einstellungen jederzeit auf die Eigenschaften der Schnittstelle übertragen. Rufen Sie "`Application.GetExportSettings`[1754]" auf, um ein `ExportSettings` Objekt aufzurufen: `objExpSettings = objSpy.GetExportSettings()`
3. Erstellen Sie mit `Document.GetExportElementList`[1858] eine Elementliste. Mit Hilfe der Elementliste können Sie XML-Elemente aus dem Exportprozess ausnehmen. Außerdem enthält diese Liste in den Eigenschaften `RecordCount` und `FieldCountInformationen` über die Anzahl der Datensätze und Felder. Setzen Sie die `ExportSettings.ElementList`[1881] Eigenschaft auf diese Sammlung. Sie können die Elementliste auf `null/Nothing` (Standardeinstellung) setzen, um alle Elemente zu exportieren.
4. Rufen Sie `Document.ExportToDatabase`[1851] auf, um den Export auszuführen. Die Beschreibung der ExportToDatabaseMethode enthält auch ein Codebeispiel für einen Datenbankexport.

© 2016-2022 Altova GmbH

```
  // set the behaviour of the export with ExportSettings
  objExpSettings = objSpy.GetExportSettings()

  //set the destination with DatabaseConnection
  objDB = objSpy.GetDatabaseSettings();
  objDB.CreateMissingTables = true;
  objDB.CreateNew = true;
  objDB.File = "C:\\Temp\\Export.mdb";

try
{
  objImpDocFromDB.ExportToDatabase(objImpDocFromDB.RootElement, objExpSettings, objDB);
}
catch(err)
{
  WScript.Echo("Error exporting to database.\n\n"+
                   "Error: "+ (err.number & 0xffff) + "\n"+
                   "Description: "+ err.description);
}
```

Sie finden den oben aufgelisteten JScript-Code in der Beispieldatei `ImportExport.js` (siehe Beispieldateien[1710]).

Der Import von Daten aus einer Textdatei erfolgt ähnlich wie der Import aus einer Datenbank. Sie müssen andere Schnittstellen (Beschreibung siehe Schritt 1-3 unten) mit anderen Methoden und Eigenschaften verwenden:

1. Verwenden Sie ein `TextImportExportSettings`[1925] Objekt und definieren Sie die Eigenschaften: Die Methode `Application.GetTextImportExportSettings`[1756] gibt ein neues Objekt zurück, das eine Textdatei für den Import definiert. `objImpSettings =objSpy.GetTextImportExportSettings();` Sie müssen zumindestens für die Eigenschaft ImportFileden Pfad zu der zu importierenden Datei definieren. Eine weitere wichtige Eigenschaft ist HeaderRow. Setzen Sie sie auf "False", wenn die Textdatei keine Überschrift als Kopfzeile enthält. `objImpSettings.ImportFile = strExampleFolder + "Tutorial\\Shapes.txt";`
2. Rufen Sie `Application.GetTextImportElementList`[1755] auf, um eine Sammlung aller Spalten in der Textdatei zu erhalten: `objElementList =objSpy.GetTextImportElementList(objImpSettings);`
3. Starten Sie den Import mit `Application.ImportFromText`[1758]. `objImpDocFromText =objSpy.ImportFromText(objImpSettings,objElementList);`

```
try
{
  // specify the source of data import
  objImpSettings = objSpy.GetTextImportExportSettings();
  objImpSettings.ImportFile = strExampleFolder + "Tutorial\\Shapes.txt";
  objImpSettings.HeaderRow = false;

  // column filter
  objElementList = objSpy.GetTextImportElementList(objImpSettings);
```

```
  // import into a new XML file
  objImpDocFromText = objSpy.ImportFromText(objImpSettings,objElementList);
}
catch(err)
{
  WScript.Echo("Error importing from text file.\n\n"+
                    "Error: "+ (err.number & 0xffff) + "\n"+
                    "Description: "+ err.description);
}
```

Sie finden den oben aufgelisteten JScript-Code in der Beispieldatei `ImportExport.js` (siehe Beispieldateien[1710]).

Um Daten in eine Textdatei zu exportieren, gehen Sie folgendermaßen vor:

1. Verwenden Sie ein `TextImportExportSettings`[1925] Objekt und definieren Sie die erforderlichen Eigenschaften.
2. Füllen Sie ein `ExportSettings`[1880] Objekt mit den erforderlichen Werten. Siehe Punkt 2 aus Export in eine Datenbank[1716].
3. Erstellen Sie eine Elementliste mit `Document.GetExportElementList`[1858]. Siehe Punkt 3 aus Export in eine Datenbank[1716].
4. Rufen Sie `Document.ExportToText`[1852] auf, um den Export auszuführen.

```
objExpSettings = objSpy.GetExportSettings();
objExpSettings.ElementList =
objImpDocFromText.GetExportElementList(objImpDocFromText.RootElement, objExpSettings);

objTextExp = objSpy.GetTextImportExportSettings();
objTextExp.HeaderRow = true;
objTextExp.DestinationFolder = "C:\\Temp";

try
{

      objImpDocFromText.ExportToText(objImpDocFromText.RootElement, objExpSettings,
objTextExp);
}
catch(err)
{
  WScript.Echo("Error exporting to text.\n\n"+
                    "Error: "+ (err.number & 0xffff) + "\n"+
                    "Description: "+ err.description);
}
```

Sie finden den oben aufgelisteten JScript-Code in der Beispieldatei `ImportExport.js` (siehe Beispieldateien[1710]).

© 2016-2022 Altova GmbH

29.3.1.2.2 VBScript

VBScript unterscheidet sich syntaktisch von JScript, funktioniert ansonsten aber auf genau die gleiche Art. Dieser Abschnitt enthält Beispielcode um zu zeigen, wie Events mit VBScript verwendet werden[1719] sowie ein Beispiel[1720].

Informationen über andere Funktionalitäten finden Sie in den unten aufgeführten JScript-Beispielen:

- Starten der Applikation oder Anhängen einer laufenden Instanz[1710]
- Einfacher Dokumentaufruf[1712]
- Iteration[1712]
- Fehlerbehandlung[1713]
- Events[1714]
- Import und Export von Daten[1715]

29.3.1.2.2.1 Events

COM definiert einen Client, der sich über den Connection Point-Mechanismus als Server für Callbacks registrieren muss. Das Automation Interface für XMLSpy definiert die dafür erforderlichen Event Interfaces. Wie eine Verbindung zu diesen Events hergestellt wird, hängt von der Programmiersprache ab, die Sie auf Ihrem Client verwenden. Im folgenden Codefragment wird gezeigt, wie dies im VBScript erfolgt.

Die Methode `WScript.ConnectObject` dient zum Empfangen von Events.

Um diesen Code auszuführen, fügen Sie ihn in eine Datei mit der Erweiterung .vbs ein und doppelklicken Sie entweder in Windows Explorer darauf oder führen Sie die Datei über die Befehlszeile aus.

```
' the event handler function
Function DocEvent_OnBeforeCloseDocument(objDocument)
  Call WScript.Echo("received event - before closing document")
EndFunction

' create or connect to XmlSpy
Set objWshShell = WScript.CreateObject("WScript.Shell")
Set objFSO = WScript.CreateObject("Scripting.FileSystemObject")
Set objSpy = WScript.GetObject("", "XMLSpy.Application")
' If only Authentic is installed (and XMLSpy is not installed) use:
' Set objSpy = WScript.GetObject("", "AuthenticDesktop.Application")
' If only XMLSpy 64-bit is intalled, use:
' Set objSpy = WScript.GetObject("", "XMLSpy_x64.Application")

' create document object and connect to its events
objSpy.Visible = True

' Find out user's personal folder and locate one of the installed examples.
personalFolder = objWshShell.ExpandEnvironmentStrings("%UserProfile%")
majorVersionYear = objSpy.MajorVersion + 1998
xmlspyExamplesFolder = personalFolder & "\Documents\Altova\XMLSpy" & majorVersionYear  &
"\Examples\"
docPath = xmlspyExamplesFolder & "ExpReport.xml"
```

```
' open a document
Setobjdoc = objSpy.Documents.OpenFile (docPath, False)
CallWScript.ConnectObject(objDoc, "DocEvent_")

' keep running while waiting on the event
' in the meantime close the document in XMLSPY manually
CallWScript.Echo ("sleeping for 10 seconds ...")
CallWScript.Sleep (10000)

SetobjDoc = Nothing
CallWScript.Echo ("stopped listening for event")
CallobjSpy.Quit
```

Anmerkung: Für die 32-Bit-Version von XMLSpy ist der registrierte Name oder der programmatische Identifier (ProgId) des COM-Objekts `XMLSpy.Application`. Für die 64-Bit-Version von XMLSpy ist der Name `XMLSpy_x64.Application`. Beachten Sie jedoch, dass das aufrufende Programm die CLASSES Registry-Einträge in seiner eigenen Registry Hive oder -Gruppe (32-Bit oder 64-Bit) aufruft. Wenn Sie daher Skripts über die Standardbefehlszeileneingabe und mit Windows Explorer auf einem 64-Bit-Windows-System ausführen, werden die 64-Bit-Registry-Einträge, welche auf die 64-Bit-Version von XMLSpy verweisen, aufgerufen. Wenn daher sowohl XMLSpy 32-Bit als auch die 64-Bit-Version installiert ist, ist eine spezielle Behandlung erforderlich, damit die 32-Bit-Version von XMLSpy aufgerufen wird. Angenommen, der Windows Skripting Host ist das aufrufende Programm, so gehen Sie folgendermaßen vor:

1. Wechseln Sie in das Verzeichnis **C:\Windows\SysWOW64**.
2. Geben Sie in der Befehlszeile **wscript.exe** gefolgt vom Pfad zum gewünschten Skript ein, z.B:

```
wscript.exe "C:\Users\...
\Documents\Altova\StyleVision2023\StyleVisionExamples\API\JScript\start.js"wscript.exe
"C:\Users\...\Documents\Altova\XMLSpy2023\Examples\API\JScript\start.js"
```

29.3.1.2.2.2 Beispiel: Verwendung von Events

Die Authentic-Ansicht unterstützt die Verbindung zu Events pro Objekt. Die Implementierung dieser Funktion basiert auf COM Connection Points und steht in Umgebungen zur Verfügung, die diesen Mechanismus unterstützen.

Das folgende Beispiel ist ein Beispiel für VBScript-Code, in dem gezeigt wird, wie man von einem VBScript-Projekt aus Events verwendet.

```
' ------------------------------------------------------------------------
' VBScript example that demonstrates how to use events.
' ------------------------------------------------------------------------

' Event handler for OnSelectionChanged event of AuthenticView
FunctionAuthenticViewEvent_OnSelectionChanged(objAuthenticRange)
```


© 2016-2022 Altova GmbH

```
   Ifobjauthenticrange.firsttextposition <> objauthenticrange.lasttextposition Then
      Callwscript.echo("selection: " & objauthenticrange.text & vbnewline & vbnewline &
"close this dialog.")
   Else
      Callwscript.echo("cursor position: " & objauthenticrange.firsttextposition &
vbnewline & vbnewline & "close this dialog.")
   EndIf
EndFunction

' Start/access XMLSpy and connect to its automation interface.
Setwshshell = wscript.CreateObject("wscript.shell")
SetobjSpy = GetObject("", "XMLSpy.Application")
' Make the UI of XMLSpy visible.
objSpy.Visible = True

' Find out user's personal folder and locate one of the installed XMLSpy examples.
personalFolder = WshShell.ExpandEnvironmentStrings("%UserProfile%")
majorVersionYear = objSpy.MajorVersion + 1998
xmlspyExamplesFolder = personalFolder & "\Documents\Altova\XMLSpy" & majorVersionYear  &
"\Examples\"
docPath = xmlspyExamplesFolder & "ExpReport.xml"

' Create object to access windows file system and test if the our document exists.
Setfso = CreateObject("Scripting.FileSystemObject")
Iffso.FileExists(docPath) Then
   ' open the document
   CallobjSpy.Documents.OpenFile(docPath, False)
   setobjDoc = objSpy.ActiveDocument

   ' switch active document to authentic view
   objDoc.SwitchViewMode 4 ' spyViewAuthentic

   ' Register for connection point events on the authentic view of the active document.
   ' Any function with a valid event name prefixed with "AuthenticViewEvent_" will
   ' be called when the corresponding event gets triggered on the specified object.
   setobjView = objDoc.AuthenticView
   CallwScript.ConnectObject(objView, "AuthenticViewEvent_")
   CallwScript.Echo("Events are connected." & vbNewLine & vbNewLine & "Now set or move
the cursor in XMLSpy." & vbNewLine & vbNewLine & "Close this dialog to shut down
XMLSpy.")

   ' To disconnect from the events delete the reference to the object.
   setobjView = Nothing
Else
   CallwScript.Echo("The file " & docPath & " does not exist.")
EndIf

' shut down XMLSpy when this script ends
objSpy.Visible = False
```

29.3.1.2.3 C#

Sie können mit Hilfe von C# auf die Funktionalitäten der Application API zugreifen. Zur Erstellung des C#-Code können Sie Visual Studio 2012/2013/2015/2017/2019/2022 verwenden und den Code in einem Visual Studio-Projekt speichern. Erstellen Sie das Projekt folgendermaßen:

1. Fügen Sie in Microsoft Visual Studio mit dem Befehl **Datei | Neu | Projekt** ein neues Projekt hinzu.
2. Fügen Sie durch Auswahl des Befehls **Projekt | Verweis hinzufügen** hinzufügen eine Referenz zur XMLSpy-Typbibliothek hinzu. Daraufhin wird das Dialogfeld "Verweis hinzufügen" angezeigt. Navigieren Sie zur XMLSpy-Typbibliothekkomponente, die sich im XMLSpy-Applikationsordner befindet und fügen Sie sie hinzu.
3. Geben Sie den gewünschten Code ein.
4. Kompilieren Sie den Code und führen Sie ihn aus.

C#-Beispielprojekt

Das XMLSpy-Paket enthält ein C#-Beispielprojekt, das Sie im Unterordner `API\C#` des Ordners `Examples` finden:

Windows 7, Windows 8, Windows 10, Windows 11	`C:\Benutzer\<Benutzername>\Dokumente\Altova\XMLSpy\2023\Examples`

Sie können das Projekt innerhalb von Visual Studio 2012/2013/2015/2017/2019/2022 kompilieren und ausführen.

Anhand der unten stehenden Codefragmente wird gezeigt, wie Sie Grundfunktionen der Applikation verwenden können. Der Code ist dem Code im C#-Beispielprojekt im Ordner API Examples Ihres Applikationsordners ähnlich, kann sich aber geringfügig davon unterscheiden.

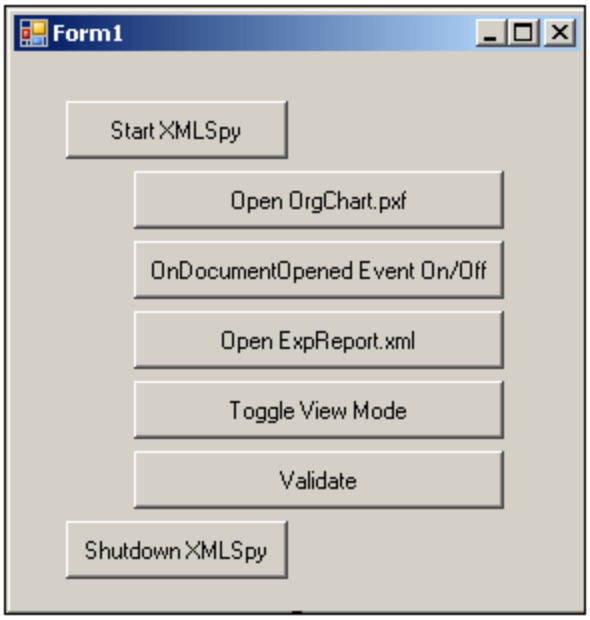
Platformkonfiguration

Wenn Sie ein 64-Bit-Betriebssystem haben und eine 32-Bit-Installation von XMLSpy verwenden, müssen Sie die x86-Plattform im Konfigurationsmanager der Projektmappe hinzufügen und das Beispiel mit dieser Konfiguration erstellen. Im Dialogfeld "Neue Projektmappenplattform" kann eine neue x86-Plattform (für die aktive Projektmappe in Visual Studio) erstellt werden (**Build | Konfigurations-Manager | Aktive Projektmappenplattform: | <Neu...>**).

Funktionen im Codebeispiel unten

In diesem Beispiel sehen Sie eine einfache Benutzeroberfläche (*Abbildung unten*) mit Schaltflächen, über die grundlegende XMLSpy-Operationen aufgerufen werden:

© 2016-2022 Altova GmbH



- Start XMLSpy[1727]: Startet XMLSpy, das als Automation Server registriert ist, oder aktiviert das Programm, wenn XMLSpy bereits ausgeführt wird.
- Open OrgChart.pxf[1728]: Navigiert zu einem der mit XMLSpy installierten Beispieldokumente und öffnet es. Wenn das Dokument bereits offen ist, wird es zum aktiven Dokument.
- OnDocumentOpened Event On/Off[1731]: Zeigt, wie die Applikation auf XMLSpy Events hört. Ist die Funktion aktiv, erscheint ein Meldungsfeld, nachdem ein Dokument geöffnet wurde.
- Open ExpReport.xml[1728]: Öffnet ein weiteres Beispieldokument.
- Toggle View Mode[1729]: Wechselt in der Ansicht aller offenen Dokumente zwischen der Text- und der Authentic-Ansicht. Im Code wird gezeigt, wie Sie durch offene Dokumente iterieren können.
- Validate[1730]: Validiert das aktive Dokument und zeigt das Ergebnis in einem Meldungsfeld an. Im Code wird gezeigt, wie Fehler und COM-Ausgabeparameter behandelt werden.
- Shutdown XMLSpy[1727]: Beendet XMLSpy.

Sie können den Code (des Codefragments unten oder des C#-Beispiels aus dem Ordner API Examples) beliebig modifizieren und ausführen.

Kompilieren und Ausführen des Beispiels

Doppelklicken Sie im Ordner API Examples auf die Datei `AutomateXMLSpy_VS2008.sln` bzw. auf die Datei `AutomateXMLSpy_VS2010.sln` (um sie in Visual Studio 2012/2013/2015/2017/2019/2022 zu öffnen). Alternativ dazu können Sie die Datei auch von Visual Studio aus öffnen (mit dem Befehl **Datei | Öffnen | Projekt/Projektmappe**). Um das Beispiel zu kompilieren und auszuführen wählen Sie **Debuggen | Debuggen starten** bzw. **Debuggen | Starten ohne Debugging**.

Code des Beispiels

Nachstehend sehen Sie den C#-Code der Grundfunktionen des im `AutomateXMLSpy` Beispiel erstellten Formulars (`Form1.cs`). Beachten Sie, dass ich der unten aufgelistete Code geringfügig von dem Code im Formular unter API Examples unterscheiden kann. Der Code ist zum besseren Verständnis mit Kommentaren

versehen. Abhängig davon, welche Application API-Funktionen sie aufrufen, sind Teile dieses Codes auch separat in Unterabschnitten dieses Abschnitts beschrieben.

Der Code besteht im Wesentlichen aus einer Reihe von Handlern für die Schaltflächen der oben gezeigten Benutzeroberfläche.

```
namespace WindowsFormsApplication2
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        // An instance of XMLSpy accessed via its automation interface
        XMLSpyLib.Application XMLSpy;

        // Location of examples installed with XMLSpy
        String strExamplesFolder;

        private void Form1_Load(object sender, EventArgs e)
        {
            // Locate examples installed with XMLSpy
            // REMARK: You might need to adapt this if you have a different major version
of the product
            strExamplesFolder = Environment.GetEnvironmentVariable("USERPROFILE") + "\\My
Documents\\Altova\\XMLSpy2012\\Examples\\";
        }

        // Handler for the "Start XMLSpy" button
        private void StartXMLSpy_Click(object sender, EventArgs e)
        {
            if (XMLSpy == null)
            {
                Cursor.Current = Cursors.WaitCursor;

                // If no XMLSpy instance is open, create one and make it visible
                XMLSpy = new XMLSpyLib.Application();
                XMLSpy.Visible = true;

                Cursor.Current = Cursors.Default;
            }
            else
            {
                // If an instance of XMLSpy is already running, make sure it's visible
                if (!XMLSpy.Visible)
                    XMLSpy.Visible = true;
            }
        }

        // Handler for the "Open OrgChart.pxf" button
        private void openOrgChart_Click(object sender, EventArgs e)
        {
            // Make sure there's a running XMLSpy instance, and that it's visible
```

 © 2016-2022 Altova GmbH

```
            StartXMLSpy_Click(null, null);

            // Open one of the example files installed with the product
            XMLSpy.Documents.OpenFile(strExamplesFolder + "OrgChart.pxf", false);
        }

        // Handler for the "Open ExpReport.xml" button
        private void openExpReport_Click(object sender, EventArgs e)
        {
            // Make sure there's a running XMLSpy instance, and that it's visible
            StartXMLSpy_Click(null, null);

            // Open one of the sample files installed with the product.
            XMLSpy.Documents.OpenFile(strExamplesFolder + "ExpReport.xml", false);
        }

        // Handler for the "Toggle View Mode" button
        private void toggleView_Click(object sender, EventArgs e)
        {
            // Make sure there's a running XMLSpy instance, and that it's visible
            StartXMLSpy_Click(null, null);

            // Iterate through all open documents and toggle view between Text View and
Authentic View
            foreach (XMLSpyLib.Document doc in XMLSpy.Documents)
                if (doc.CurrentViewMode == XMLSpyLib.SPYViewModes.spyViewText)
                    doc.SwitchViewMode(XMLSpyLib.SPYViewModes.spyViewAuthentic);
                else
                    doc.SwitchViewMode(XMLSpyLib.SPYViewModes.spyViewText);
        }

        // Handler for the "Shutdown XMLSpy" button
        // Shut down the application instance by explicitly releasing the COM object
        private void shutdownXMLSpy_Click(object sender, EventArgs e)
        {
            if (XMLSpy != null)
            {
                // Allow shutdown of XMLSpy by releasing the UI
                XMLSpy.Visible = false;

                // Explicitly release the COM object
                try
                {
                    while (System.Runtime.InteropServices.Marshal.ReleaseComObject(XMLSpy)
> 0) ;
                }
                finally
                {
                    // Disallow subsequent access to this object
                    XMLSpy = null;
                }
            }
        }

        // Handler for button "Validate"
        private void validate_Click(object sender, EventArgs e)
```

```
        {
            // COM errors are returned to C# as exceptions. We use a try/catch block to
handle them.
            try
            {
                // Method 'IsValid' is one of the few functions that uses output parameters
                // Use 'object' type for these parameters
                object strErrorText = "";
                object nErrorNumber = 0;
                object errorData = null;

                if (!XMLSpy.ActiveDocument.IsValid(ref strErrorText, ref nErrorNumber, ref
errorData))
                {
                    // The COM call succeeded but the document is not valid
                    // A detailed description of the problem is returned in strErrorText,
nErrorNumber and errorData
                    listBoxMessages.Items.Add("Document " + XMLSpy.ActiveDocument.Name + "
is not valid.");
                    listBoxMessages.Items.Add("\tErrorText  : " + strErrorText);
                    listBoxMessages.Items.Add("\tErrorNumber: " + nErrorNumber);
                    listBoxMessages.Items.Add("\tElement    : " + (errorData != null ?
((XMLSpyLib.XMLData)errorData).TextValue : "null"));
                }
                else
                {
                    // The COM call succeeded and the document is valid
                    listBoxMessages.Items.Add("Document " + XMLSpy.ActiveDocument.Name + "
is valid.");
                }
            }
            catch (Exception ex)
            {
                // The COM call was not successful
                // Probably no application instance has been started or no document is
open.
                listBoxMessages.Items.Add("Error validating active document: " +
ex.Message);
            }
        }



        // Event handler for OnDocumentOpened event
        private void handleOnDocumentOpened(XMLSpyLib.Document i_ipDocument)
        {
            MessageBox.Show("Document " + i_ipDocument.Name + " was opened!");
        }

        // Remember if the event handler is currently registered.
        private bool bEventHandlerIsRegistered = false;

        // Handler for button 'OnDocuemntOpened Event On/Off
        private void toggleOnDocumentOpenedEvent_Click(object sender, EventArgs e)
        {
            if (XMLSpy != null)
```

```
            {
                if (bEventHandlerIsRegistered)
                    XMLSpy.OnDocumentOpened -= new
XMLSpyLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpened);
                else
                    XMLSpy.OnDocumentOpened += new
XMLSpyLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpened);

                bEventHandlerIsRegistered = !bEventHandlerIsRegistered;
            }
        }
    }
}
```

29.3.1.2.3.1 Hinzufügen einer Referenz zur XMLSpy API

Um die Typbibliothek der Applikations-API als Referenz in einem .NET-Projekt hinzuzufügen, gehen Sie folgendermaßen vor: Klicken Sie, während das .NET-Projekt geöffnet ist, auf den Befehl **Projekt | Verweis hinzufügen**. Navigieren Sie zur Typbibliothek `XMLSpy.tlb` im XMLSpy Applikationsordner.

Deklarieren Sie eine Variable zum Aufrufen der XMLSpy API

```
// An instance of XMLSpyis accessed via its automation interface.
XMLSpyLib.ApplicationXMLSpy;
```

29.3.1.2.3.2 Starten und Beenden der Applikation

In den nachstehenden Codefragmenten wurden den Schaltflächen im Beispiel AutomateXMLSpy[1722] die Methoden `StartXMLSpy_Click` und `ShutdownXMLSpy_Click` zugewiesen. Sie starten bzw. beenden die Applikation. Dieses Beispiel befindet sich im Unterordner C# des Ordners API Examples (*siehe Datei Form1.cs)*:

Windows 7, Windows 8, Windows 10, Windows 11	`C:\Benutzer\<Benutzername>\Dokumente\Altova\XMLSpy\2023\Examples`

Sie können das Projekt von Visual Studio 2012/2013/2015/2017/2019/2022 aus kompilieren und ausführen.

Starten von XMLSpy

Im folgenden Codefragment aus dem Beispiel AutomateXMLSpy[1722] wird gezeigt, wie man die Applikation startet.

```
// Handler for the "Start XMLSpy" button
private void StartXMLSpy_Click(object sender, EventArgs e)
{
    if (XMLSpy == null)
    {
```

```
            Cursor.Current = Cursors.WaitCursor;

            // If no XMLSpy instance is running, we create one and make it visible
            XMLSpy = new XMLSpyLib.Application();
            XMLSpy.Visible = true;

            Cursor.Current = Cursors.Default;
        }
        else
        {
            // If an instance of XMLSpy is already running, make sure it's visible
            if (!XMLSpy.Visible)
                XMLSpy.Visible = true;
        }
    }
```

Beenden von XMLSpy

Im folgenden Codefragment aus dem Beispiel AutomateXMLSpy[1722] wird gezeigt, wie man die Applikation beendet.

```
    // Handler for the "Shutdown XMLSpy" button
    // Shut down the application instance by explicitly releasing the COM object
     private void shutdownXMLSpy_Click(object sender, EventArgs e)
     {
         if (XMLSpy != null)
         {
             // Allow shutdown of XMLSpy by releasing the UI
             XMLSpy.Visible = false;

             // Explicitly release COM object
             try
             {
                 while (System.Runtime.InteropServices.Marshal.ReleaseComObject(XMLSpy)
> 0) ;
             }
             finally
             {
                 // Disallow subsequent access to this object
                 XMLSpy = null;
             }
         }
     }
```

29.3.1.2.3.3 Öffnen von Dokumenten

In den unten stehenden Codefragmenten (aus dem Beispiel AutomateXMLSpy[1722]) wird gezeigt, wie zwei Dateien über zwei separate Methoden, die zwei Schaltflächen der Benutzeroberfläche zugewiesen wurden, geöffnet werden. Beide Methoden bedienen sich desselben Zugriffsmechanismus auf die Application API: XMLSpy.Documents.OpenFile(string, boolean)[1872].

Das AutomateXMLSpy Beispiel[1722] (*siehe Datei form1.cs)* befindet sich im Unterordner C# des Ordners API Examples:

Windows 7, Windows 8, Windows 10, Windows 11	`C:\Benutzer\<Benutzername>\Dokumente\Altova\XMLSpy\2023\Examples`

Sie können das Projekt von Visual Studio 2012/2013/2015/2017/2019/2022 aus kompilieren und ausführen.

Codefragment

```
// Handler for the "Open OrgChart.pxf" button
 private void openOrgChart_Click(object sender, EventArgs e)
 {
     // Make sure there's a running XMLSpy instance, and that it's visible
     StartXMLSpy_Click(null, null);

     // Open a file from the Examples folder installed with the product
     XMLSpy.Documents.OpenFile(strExamplesFolder + "OrgChart.pxf", false);
 }

 // Handler for the "Open ExpReport.xml" button
 private void openExpReport_Click(object sender, EventArgs e)
 {
     // Make sure there's a running XMLSpy instance, and that it's visible
     StartXMLSpy_Click(null, null);

     // Open a file from the Examples folder installed with the product
     XMLSpy.Documents.OpenFile(strExamplesFolder + "ExpReport.xml", false);
 }
```

Die zuletzt geöffnete Datei ist die aktive Datei.

29.3.1.2.3.4 Iterieren durch offene Dokumente

Im unten gezeigten Codefragment (aus dem Beispiel AutomateXMLSpy[1722]; *siehe Datei form1.cs*) wird gezeigt, wie man durch offene Dokumente iteriert. Anschließend wird in der Iterationsschleife eine Bedingung überprüft und es wird zwischen der Text- und der Authentic-Ansicht gewechselt.

```
// Handler for the "Toggle view mode" button
 private void toggleView_Click(object sender, EventArgs e)
 {
     // Make sure there's a running XMLSpy instance, and that it's visible
     StartXMLSpy_Click(null, null);

     // Iterate through open documents and toggle current view between text and
authentic view.
     foreach (XMLSpyLib.Document doc in XMLSpy.Documents)
         if (doc.CurrentViewMode == XMLSpyLib.SPYViewModes.spyViewText)
             doc.SwitchViewMode(XMLSpyLib.SPYViewModes.spyViewAuthentic);
         else
             doc.SwitchViewMode(XMLSpyLib.SPYViewModes.spyViewText);
```

```
    }
```

Das Beispiel AutomateXMLSpy[1722] befindet sich im Unterordner C# des Ordners API Examples:

Windows 7, Windows 8, Windows 10, Windows 11	C:\Benutzer\<Benutzername>\Dokumente\Altova\XMLSpy\2023\Examples

Sie können das Projekt von Visual Studio 2012/2013/2015/2017/2019/2022 aus kompilieren und ausführen.

29.3.1.2.3.5 Fehler und COM-Ausgabeparameter

Im unten gezeigten Codefragment (aus dem Beispiel AutomateXMLSpy[1722]) wird gezeigt, wie man Fehler und COM-Ausgabeparameter behandelt. In der Methode XMLSpy.ActiveDocument.IsValid(ref strErrorText, ref nErrorNumber, ref errorData)[1859] werden Ausgabeparameter verwendet, mit Hilfe derer im unten gezeigten Codefragment der Text einer Fehlermeldung generiert wird.

Das AutomateXMLSpy Beispiel[1722] (*siehe Datei form1.cs)* befindet sich im Unterordner C# des Ordners API Examples:

Windows 7, Windows 8, Windows 10, Windows 11	C:\Benutzer\<Benutzername>\Dokumente\Altova\XMLSpy\2023\Examples

Sie können das Projekt von Visual Studio 2012/2013/2015/2017/2019/2022 aus kompilieren und ausführen.

Codefragment

```
     // Handler for button "Validate"
        private void validate_Click(object sender, EventArgs e)
        {
            // COM errors are returned to C# as exceptions. We use a try/catch block to
handle them.
            try
            {
                // Method 'IsValid' is one of the few functions that uses output parameters
                // Use 'object' type for these parameters
                object strErrorText = "";
                object nErrorNumber = 0;
                object errorData = null;

                if (!XMLSpy.ActiveDocument.IsValid(ref strErrorText, ref nErrorNumber, ref
errorData))
                {
                    // The COM call succeeded but the document is not valid
                    // A detailed description of the problem is returned in strErrorText,
nErrorNumber and errorData
```

 © 2016-2022 Altova GmbH

```
                    listBoxMessages.Items.Add("Document " + XMLSpy.ActiveDocument.Name + "
is not valid.");
                    listBoxMessages.Items.Add("\tErrorText  : " + strErrorText);
                    listBoxMessages.Items.Add("\tErrorNumber: " + nErrorNumber);
                    listBoxMessages.Items.Add("\tElement    : " + (errorData != null ?
((XMLSpyLib.XMLData)errorData).TextValue : "null"));
                }
                else
                {
                    // The COM call succeeded and the document is valid
                    listBoxMessages.Items.Add("Document " + XMLSpy.ActiveDocument.Name + "
is valid.");
                }
            }
            catch (Exception ex)
            {
                // The COM call was not successful
                // Probably no application instance has been started or no document is
open.
                listBoxMessages.Items.Add("Error validating active document: " +
ex.Message);
            }
        }
```

29.3.1.2.3.6 Events

Im unten gezeigten Codefragment (aus dem Beispiel AutomateXMLSpy[1722]) wird der Code für zwei Event Handler aufgelistet. Das AutomateXMLSpy Beispiel[1722] (*siehe Datei form1.cs)* befindet sich im Unterordner C# des Ordners API Examples:

Windows 7, Windows 8, Windows 10, Windows 11	C:\Benutzer\<Benutzername>\Dokumente\Altova\XMLSpy\2023\Examples

Sie können das Projekt von Visual Studio 2012/2013/2015/2017/2019/2022 aus kompilieren und ausführen.

Codefragment

```
    // Event handler for OnDocumentOpened event
        private void handleOnDocumentOpened(XMLSpyLib.Document i_ipDocument)
        {
            MessageBox.Show("Document " + i_ipDocument.Name + " was opened!");
        }

        // Remember if the event handler is currently registered.
        private bool bEventHandlerIsRegistered = false;

        // Handler for button 'OnDocuemntOpened Event On/Off
        private void toggleOnDocumentOpenedEvent_Click(object sender, EventArgs e)
        {
            if (XMLSpy != null)
```

```
            {
                if (bEventHandlerIsRegistered)
                    XMLSpy.OnDocumentOpened -= new
XMLSpyLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpened);
                else
                    XMLSpy.OnDocumentOpened += new
XMLSpyLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpened);

                bEventHandlerIsRegistered = !bEventHandlerIsRegistered;
            }
        }
```

29.3.1.2.4 Java

Sie können mit Hilfe von Java-Code auf die Applikations-API zugreifen. Damit Sie direkt vom Java-Code aus Zugriff auf den APPNAME%> Automation Server haben, müssen sich die unten aufgelisteten Bibliotheken im `classpath` befinden. Sie sind im XMLSpy Applikationsordner im Ordner `JavaAPI` installiert.

- `AltovaAutomation.dll`: ein JNI Wrapper für Altova Automation Server (Im Fall von 64-Bit-Versionen `AltovaAutomation_x64.dll`)
- `AltovaAutomation.jar`: Java-Klassen zum Aufrufen von Altova Automation Servern
- `XMLSpyAPI.jar`: Java-Klassen, die als Wrapper für die XMLSpy Automation-Schnittstelle dienen
- `XMLSpyAPI_JavaDoc.zip`: eine Javadoc Datei, die die Hilfedokumentation zur Java API enthält

Anmerkung: Um die Java API verwenden zu können, müssen sich die DLL und Jar-Dateien im Java Classpath befinden.

Java-Beispielprojekt

Im Lieferumfang Ihres Produkts ist ein Java-Beispielprojekt enthalten. Sie können das Java-Projekt nach Belieben testen und verwenden. Nähere Informationen zum Java-Beispielprojekt finden Sie im Abschnitt Java-Beispielprojekt 1733.

Regeln für das Mappen der Application API-Namen auf Java

Für das Mapping zwischen der Application API und dem Java Wrapper gelten die folgenden Regeln:

- ***Klassen und Klassennamen***
  Für jedes Interface des XMLSpy Automation Interface gibt es eine Java-Klasse mit dem Namen des Interface.

- ***Methodennamen***
  Die Methodennamen im Java Interface sind dieselben wie die in den COM Interfaces, beginnen aber aufgrund der Java-Namenskonventionen mit einem Kleinbuchstaben. Zum Aufrufen von COM-Eigenschaften können dem Eigenschaftsnamen vorangestellte Java-Methoden mit `get` und `set` verwendet werden. Wenn eine Eigenschaft keinen Schreibzugriff ermöglicht, steht keine Setter-Methode zur Verfügung. Beispiel: Für die Eigenschaft `Name` des `Document` Interface stehen die Java-Methoden `getName` und `setName` zur Verfügung.

- ***Enumerationen***
  Für jede im Automation Interface definierte Enumeration ist eine Java-Enumeration desselben Namens und mit denselben Werten definiert.

© 2016-2022 Altova GmbH

- ***Events und Event Handler***

Für jedes Interface im Automation Interface, das Events unterstützt, steht ein Java-Interface desselben Namens plus '`Event`' zur Verfügung. Um das Überladen von Einzel-Events zu vereinfachen, gibt es eine Java-Klasse mit Standardimplementierungen für alle Events. Der Name dieser Java-Klasse ist der Name des Event Interface plus '`DefaultHandler`'. Beispiel:
`Application`: Java-Klasse zum Aufrufen der Applikation
`ApplicationEvents`: Events Interface für die Applikation
`ApplicationEventsDefaultHandler`: Standard-Handler für `ApplicationEvents`

Ausnahmen für Mapping-Regeln

Zu den oben aufgelisteten Regeln gibt es die folgenden Ausnahmen:

Interface	Java-Name
`Document`, Methode `SetEncoding`	`setFileEncoding`
`AuthenticView`, Methode `Goto`	`gotoElement`
`AuthenticRange`, Methode `Goto`	`gotoElement`
`AuthenticRange`, Methode `Clone`	`cloneRange`

Dieser Abschnitt

In diesem Abschnitt wird erklärt, wie einige grundlegende XMLSpy-Funktionen über Java-Code aufgerufen werden können. Der Abschnitt ist in die folgenden Unterabschnitte gegliedert:

- Java-Beispielprojekt 1733
- Starten und Beenden der Applikation 1737
- Einfacher Dokumentaufruf 1738
- Iterationen 1738
- Verwendung von Out-Parametern 1739
- Event Handler 1740

29.3.1.2.4.1 Java-Beispielprojekt

Das XMLSpy-Installationspaket enthält ein Java-Beispielprojekt, das Sie im Unterordner `API\Java` des Ordners `Examples` finden:

Windows 7, Windows 8, Windows 10, Windows 11	`C:\Benutzer\<Benutzername>\Dokumente\Altova\XMLSpy\2023\Examples`

Dieser Ordner enthält Java-Beispiele für die XMLSpy API. Sie können das Beispielprojekt mit Hilfe der Batch-Datei `BuildAndRun.bat`, direkt über die Befehlszeile testen oder Sie können es in Eclipse kompilieren und ausführen. Anleitungen dafür finden Sie weiter unten.

Dateiliste

Der Ordner für die Java-Beispiele enthält alle zum Ausführen des Beispielprojekts erforderlichen Dateien. Diese Dateien sind unten aufgelistet. Wenn Sie eine 64-Bit-Version der Applikation verwenden, enthalten einige Dateinamen _x64. Diese Dateinamen sind mit `(_x64)` gekennzeichnet.

`AltovaAutomation(_x64).dll`	Java-COM Bridge: DLL-Teil
`AltovaAutomation.jar`	Java-COM Bridge: Java-Bibliotheksteil
`XMLSpyAPI.jar`	Java-Klassen der XMLSpy API
`RunXMLSpy.java`	Java-Beispielquellcode
`BuildAndRun.bat`	Batch-Datei zum Kompilieren und Ausführen des Beispielcodes über die Befehlszeile. Es wird ein Ordner benötigt, in dem sich die Java Virtual Machine als Parameter befindet.
`.classpath`	Hilfedatei Eclipse-Projekt
`.project`	Eclipse-Projektdatei
`XMLSpyAPI_JavaDoc.zip`	`Javadoc` Datei, die die Hilfedokumentation für die Java API enthält

Funktionen in diesem Beispiel

In diesem Beispielprojekt wird XMLSpy gestartet und einige Operationen wie das Öffnen und Schließen von Dokumenten werden ausgeführt. XMLSpy bleibt danach geöffnet. Sie müssen die Applikation manuell schließen.

- XMLSpy starten[1737]: Startet XMLSpy, das als Automation Server registriert ist, bzw. aktiviert das Programm, wenn XMLSpy bereits ausgeführt wird.
- OrgChart.pxf öffnen[1738]: Navigiert zu einem der mit XMLSpy installierten Beispieldokumente und öffnet es.
- Iterieren durch und Ändern des Ansichtsmodus[1738]: Wechselt von der Ansicht aller offenen Dokumente in die Textansicht. Im Code wird auch gezeigt, wie man durch offene Dokumente iteriert.
- Iteration, Validierung und Ausgabeparameter[1739]: Validiert das aktive Dokument und zeigt das Ergebnis in einem Meldungsfeld an. Im Code wird gezeigt, wie Ausgabeparameter verwendet werden.
- Event-Behandlung[1740]: Zeigt, wie XMLSpy Events behandelt werden.
- XMLSpy beenden[1737]: Beendet XMLSpy.

Sie können das Beispiel beliebig modifizieren und es ausführen.

Ausführen des Beispiels über die Befehlszeile

Um das Beispiel von der Befehlszeile aus auszuführen, öffnen Sie ein Eingabeaufforderungsfester, gehen Sie zum Ordner Java des Ordners API Examples (*Pfad siehe oben*) und geben Sie folgende Zeile ein:

```
buildAndRun.bat "<Path-to-the-Java-bin-folder>"
```

Der Java Binary-Ordner muss von einer JDK 1.5 oder höheren Version auf Ihrem Rechner sein. Drücken Sie die **Eingabetaste.** Der Java-Quellcode in `RunXMLSpy.java` wird kompiliert und anschließend ausgeführt.

© 2016-2022 Altova GmbH

Laden des Beispiels in Eclipse

Öffnen Sie Eclipse und wählen Sie den Befehl **Import | Existing Projects into Workspace** um die Eclipse-Projektdatei (`.project`) im Ordner Java des Ordners API Examples (*Pfad siehe oben*) zu Eclipse hinzuzufügen. Daraufhin wird das Projekt `RunXMLSpy` in Ihrem Package Explorer oder Navigator angezeigt. Wählen Sie das Projekt aus und klicken Sie anschließend auf **Run as | Java Application** um das Beispiel auszuführen.

Anmerkung: Sie können einen Klassennamen oder eine Methode der Java API auswählen und F1 drücken, um Hilfe zu dieser Klasse oder Methode zu erhalten.

Java-Quellcode

Im Folgenden finden Sie den mit Kommentaren versehenen Java-Quellcode aus der Beispieldatei `RunXMLSpy.java`.

```
001 // Access general JAVA-COM bridge classes
002 import com.altova.automation.libs.*;
003
004 // Access XMLSpy Java-COM bridge
005 import com.altova.automation.XMLSpy.*;
006 import com.altova.automation.XMLSpy.Enums.SPYViewModes;
007
008 /**
009  * An example that starts XMLSpy COM server and performs view operations on it
010  * Feel free to extend
011  */
012 public class RunXMLSpy
013 {
014   public static void main(String[] args)
015   {
016     // An instance of the application.
017     Application xmlSpy = null;
018
019     // Instead of COM error handling, use Java exception mechanism
020     try
021       {
022       // Start XMLSpy as COM server
023       xmlSpy = new Application();
024
025       // COM servers start up invisible, so make it visible
026       xmlSpy.setVisible(true);
027
028       // Locate samples installed with the product
029       String strExamplesFolder =
030       System.getenv("USERPROFILE") + "\\My Documents\\Altova\\XMLSpy2012\\Examples\\";
031
032       // Open two example files
033       xmlSpy.getDocuments().openFile(strExamplesFolder + "OrgChart.pxf", false);
034       xmlSpy.getDocuments().openFile(strExamplesFolder + "ExpReport.xml", false);
035
036       // Iterate through open documents and set view mode to 'Text'.
037       for (Document doc:xmlSpy.getDocuments())
```

```
038       if ( doc.getCurrentViewMode() != SPYViewModes.spyViewText)
039           doc.switchViewMode(SPYViewModes.spyViewText);
040
041       // An alternative iteration mode is index-based
042       // COM indices are typically zero-based
043       Documents documents = xmlSpy.getDocuments();
044       for (int i = 1; i <= documents.getCount();
045           i++)
046           {
047             Document doc = documents.getItem(i);
048
049       // Validation is one of the few methods to have output parameters.
050       // The class JVariant is the correct type for parameters in these cases.
051       // To get values back mark them with the by-reference flag.
052       JVariant validationErrorText = new
053
054       JVariant.JStringVariant("");
055
056       validationErrorText.setByRefFlag();
057       JVariant validationErrorCount = new
058
059       JVariant.JIntVariant(0);
060
061       validationErrorCount.setByRefFlag();
062       JVariant validationErrorXMLData = new
063
064       JVariant.JIDispatchVariant(0);
065
066       validationErrorXMLData.setByRefFlag();
067       if (!doc.isValid(validationErrorText, validationErrorCount,
validationErrorXMLData))
068           System.out.println("Document" + doc.getName() + " is not wellformed - " +
validationErrorText.getStringValue());
069         else
070           System.out.println("Document" + doc.getName() + " is wellformed.");
071       }
072
073       // The following lines attach to the document events using a default
implementation
074       // for the events and override one of its methods.
075       // If you want to override all document events it is better to derive your
listener class
076       // from DocumentEvents and implement all methods of this interface.
077       Document doc = xmlSpy.getActiveDocument();
078       doc.addListener(new
079
080 DocumentEventsDefaultHandler()
081       {
082         @Override
083         public boolean
084
085 onBeforeCloseDocument(Document i_ipDoc) throws AutomationException
086         {
087           System.out.println("Document
088
089 " + i_ipDoc.getName() + " requested closing.");
```

 © 2016-2022 Altova GmbH

```
090
091           // Allow closing of document
092           return true;
093         }
094       });
095       doc.close(true);
096       doc = null;
097
098       System.out.println("Watch XMLSpy!");
099     }
100     catch (AutomationException e)
101     {
102       // e.printStackTrace();
103     }
104     finally
105     {
106       // Make sure that XMLSpy can shut down properly.
107       if (xmlSpy != null)
108           xmlSpy.dispose();
109
110       // Since the COM server was made visible and still is visible,
111       // it will keep running, and needs to be closed manually.
112       System.out.println("Now close XMLSpy!");
113     }
114   }
115 }
```

29.3.1.2.4.2 Starten und Beenden der Applikation

Im nachstehenden Code wird gezeigt, wie man die Applikation startet und beendet.

Starten der Applikation

Bevor Sie die Applikation starten müssen die entsprechenden Klassen importiert werden (*siehe unten*).

```
01 // Access general JAVA-COM bridge classes
02 import com.altova.automation.libs.*;
03
04 // Access XMLSpy Java-COM bridge
05 import com.altova.automation.XMLSpy.*;
06 import com.altova.automation.XMLSpy.Enums.SPYViewModes;
07
08 /**
09  * An example that starts XMLSpy COM server and performs view operations on it
10  * Feel free to extend
11  */
12 public class RunXMLSpy
13 {
14   public static void main(String[] args)
15   {
16     // An instance of the application.
17     Application xmlSpy = null;
18
```

```
19     // Instead of COM error handling, use Java exception mechanism
20     try
21     {
22         // Start XMLSpy as COM server
23           xmlSpy = new Application();
24         // COM servers start up invisible, so make it visible
25           xmlSpy.setVisible(true);
26
27 ...
28     }
29   }
30 }
```

Beenden der Applikation

Die Applikation kann wie unten gezeigt beendet werden.

```
01 {
02       // Allow shutdown of XMLSpy by releasing the UI.
03       xmlSpy.setVisible(true);
04
05       // Make sure that XMLSpy can shut down properly.
06       if (xmlSpy != null)
07         xmlSpy.dispose();
08
09       // Since the COM server was made visible and still is visible,
10       // it will keep running, and needs to be closed manually.
11       System.out.println("Now close XMLSpy!");
12 }
```

29.3.1.2.4.3 Einfacher Dokumentzugriff

Im unten stehenden Code wird gezeigt, wie man ein Dokument öffnet.

```
1  // Locate samples installed with the product
2  String strExamplesFolder =
3  System.getenv("USERPROFILE") + "\\My Documents\\Altova\\XMLSpy2012\\Examples\\";
4
5
6  // Open file
7  xmlSpy.getDocuments().openFile(strExamplesFolder + "OrgChart.pxf", false);
```

29.3.1.2.4.4 Iterationen

Im nachstehenden Code wird gezeigt, wie man durch offene Dokumente iteriert.

```
01  // Iterate through open documents and set view mode to 'Text'.
02  for (Document doc:xmlSpy.getDocuments())
```


© 2016-2022 Altova GmbH

```
03    if ( doc.getCurrentViewMode() != SPYViewModes.spyViewText)
04             doc.switchViewMode(SPYViewModes.spyViewText);
05
06  // An alternative iteration mode is index-based
07  // COM indices are typically zero-based
08  Documents documents = xmlSpy.getDocuments();
09     for (int i = 1; i <= documents.getCount();
10         i++)
11       {
12         Document doc = documents.getItem(i);
13               ...
14       }
```

29.3.1.2.4.5 Verwendung von Out-Parametern

Im unten stehenden Code wird durch offene Dokumente iteriert, wobei jedes davon validiert wird. Für jede Validierung wird mit Hilfe der Ausgabeparameter der Methode "Validation" eine Meldung generiert.

```
01 // An alternative iteration mode is index-based
02 // COM indices are typically zero-based
03 Documents documents = xmlSpy.getDocuments();
04 for (int i = 1; i <= documents.getCount();
05       i++)
06     {
07         Document doc = documents.getItem(i);
08
09 // Validation is one of the few methods to have output parameters.
10 // The class JVariant is the correct type for parameters in these cases.
11 // To get values back mark them with the by-reference flag.
12 JVariant validationErrorText = new
13
14 JVariant.JStringVariant("");
15
16 validationErrorText.setByRefFlag();
17       JVariant validationErrorCount = new
18
19 JVariant.JIntVariant(0);
20
21 validationErrorCount.setByRefFlag();
22       JVariant validationErrorXMLData = new
23
24 JVariant.JIDispatchVariant(0);
25
26 validationErrorXMLData.setByRefFlag();
27       if (!doc.isValid(validationErrorText,
28
29         validationErrorCount, validationErrorXMLData))
30         System.out.println("Document
31
32         " + doc.getName() + " is not wellformed - " +
33
34         validationErrorText.getStringValue());
```

```
35       else
36         System.out.println("Document
37
38         " + doc.getName() + " is wellformed.");
39     }
```

29.3.1.2.4.6 Event Handler

Im nachstehenden Code wird gezeigt, wie die Applikation auf XMLSpy Events hört und diese verwendet.

```
01 // The following lines attach to the document events using a default implementation
02 // for the events and override one of its methods.
03 // If you want to override all document events it is better to derive your listener
class
04 // from DocumentEvents and implement all methods of this interface.
05
06 Document doc = xmlSpy.getActiveDocument();
07 doc.addListener(new DocumentEventsDefaultHandler()
08     {
09       @Override
10       public boolean
11       onBeforeCloseDocument(Document i_ipDoc) throws AutomationException
12       {
13         System.out.println("Document " + i_ipDoc.getName() + " requested closing.");
14
15         // Allow closing of document
16         return true;
17       }
18     });
19 doc.close(true);
20 doc = null;
```

29.3.1.3 DOM und XML Data

Über die XMLDataSchnittstelle haben Sie Zugriff auf die gesamte XML-Struktur, die dem aktuellen Dokument zugrunde liegt. Sie benötigen dafür weniger Methoden als DOM und der Zugriff ist einfacher. Die XMLDataSchnittstelle ist eine möglichst einfache Methode, um vorhandene oder neu erstellte XML-Daten zu lesen und zu bearbeiten. Eventuell sollten Sie jedoch eine DOM-Struktur verwenden, weil Sie auf diese von einer externen Quelle aus zugreifen können oder weil Sie die MSXML DOM-Implementierung vorziehen.

Die unten gezeigten Funktionen ProcessDOMNode()und ProcessXMLDataNode() konvertieren alle Segmente einer XML-Struktur zwischen XMLDataund DOM.

Um die ProcessDOMNode() Funktion zu verwenden:
- übergeben Sie das Root-Element des zu konvertierenden DOM-Segments in objNode und
- übergeben Sie das Plugin-Objekt mit der CreateChild() Methode in objCreator

Um die ProcessXMLDataNode() Funktion zu verwenden:
- übergeben Sie das Root-Element des XMLData-Segments in objXMLData und

 © 2016-2022 Altova GmbH

- übergeben Sie das mit MSXML in xmlDoc erstellte DOMDocument

```
////////////////////////////////////////////////////////////
// DOM To XMLData conversion
Function ProcessDOMNode(objNode,objCreator)
{
	var objRoot;
	objRoot = CreateXMLDataFromDOMNode(objNode,objCreator);

	If(objRoot)   {
		If((objNode.nodeValue != Null) && (objNode.nodeValue.length > 0))
			objRoot.TextValue = objNode.nodeValue;
			// add attributes
		If(objNode.attributes) {
			var Attribute;
			var oNodeList = objNode.attributes;

			For(var i = 0;i < oNodeList.length; i++) {
				Attribute = oNodeList.item(i);

				var newNode;
				newNode = ProcessDOMNode(Attribute,objCreator);

				objRoot.AppendChild(newNode);
			}
		}
		If(objNode.hasChildNodes)   {
			try {
				// add children
				var Item;
				oNodeList = objNode.childNodes;

				For(var i = 0;i < oNodeList.length; i++) {
					Item = oNodeList.item(i);

					var newNode;
					newNode = ProcessDOMNode(Item,objCreator);

					objRoot.AppendChild(newNode);
				}
			}
			catch(err) {
			}
		}
	}
	Return objRoot;
}

Function CreateXMLDataFromDOMNode(objNode,objCreator)
{
	var bSetName = True;
	var bSetValue = True;

	var nKind = 4;

	switch(objNode.nodeType)    {
```

```
                Case 2:nKind = 5;break;
                Case 3:nKind = 6;bSetName = False;break;
                Case 4:nKind = 7;bSetName = False;break;
                Case 8:nKind = 8;bSetName = False;break;
                Case 7:nKind = 9;break;
        }
        var objNew = Null;
        objNew = objCreator.CreateChild(nKind);

        If(bSetName)
                objNew.Name = objNode.nodeName;

        If(bSetValue && (objNode.nodeValue != Null))
                objNew.TextValue = objNode.nodeValue;

        Return objNew;
}
////////////////////////////////////////////////////////////
// XMLData To DOM conversion

Function ProcessXMLDataNode(objXMLData,xmlDoc)
{
        var objRoot;
        objRoot = CreateDOMNodeFromXMLData(objXMLData,xmlDoc);

        If(objRoot)   {
                If(IsTextNodeEnabled(objRoot) && (objXMLData.TextValue.length > 0))
                        objRoot.appendChild(xmlDoc.createTextNode(objXMLData.TextValue));

                If(objXMLData.HasChildren)  {
                        try {
                                var objChild;
                                objChild = objXMLData.GetFirstChild(-1);

                                While(True)   {
                                        If(objChild)  {
                                                var newNode;
                                                newNode = ProcessXMLDataNode(objChild,xmlDoc);

                                                If(newNode.nodeType == 2)   {
                                                        // child node is an attribute
                                                        objRoot.attributes.setNamedItem(newNode);
                                                }
                                                Else
                                                        objRoot.appendChild(newNode);
                                        }
                                        objChild = objXMLData.GetNextChild();
                                }
                        }
                        catch(err) {
                        }
                }
        }
        Return objRoot;
}
```

 © 2016-2022 Altova GmbH

```
Function CreateDOMNodeFromXMLData(objXMLData,xmlDoc)
{
	switch(objXMLData.Kind)	{
		Case 4:Return xmlDoc.createElement(objXMLData.Name);
		Case 5:Return xmlDoc.createAttribute(objXMLData.Name);
		Case 6:Return xmlDoc.createTextNode(objXMLData.TextValue);
		Case 7:Return xmlDoc.createCDATASection(objXMLData.TextValue);
		Case 8:Return xmlDoc.createComment(objXMLData.TextValue);
		Case 9:Return
xmlDoc.createProcessingInstruction(objXMLData.Name,objXMLData.TextValue);
	}

	Return xmlDoc.createElement(objXMLData.Name);
}
Function IsTextNodeEnabled(objNode)
{
	switch(objNode.nodeType) {
		Case 1:
		Case 2:
		Case 5:
		Case 6:
		Case 11:Return True;
	}
	Return False;
}
```

29.3.1.4 Nicht mehr verwendete Zeilenoperationen in der Authentic-Ansicht

Wenn im Schema, auf dem das XML-Dokument basiert, festgelegt ist, dass ein Element wiederholbar ist, kann eine derartige Struktur in der Authentic-Ansicht als Tabelle dargestellt werden. Bei Darstellung in einer Tabelle können Zeilen und ihr Inhalt einzeln bearbeitet werden, wodurch Sie jedes der wiederholbaren Elemente einzeln bearbeiten können. Solche Zeilenoperationen würden von einem externen Skript durchgeführt werden.

Wenn ein externes Skript Zeilenoperationen durchführen soll, sind zwei Schritte erforderlich:

- Im ersten Schritt wird überprüft, ob sich der Cursor gerade in einer Zeile befindet, in der eine Eigenschaft verwendet wird. Eine solche Überprüfung könnte z.B. sein IsRowInsertEnabled.Diese Methode gibt das Ergebnis TRUE oder FALSE zurück.
- Wenn der Rückgabewert TRUE ist, kann eine Zeilenmethode wie z.B. RowAppendaufgerufen werden. (RowAppendhat keine Parameter und gibt keinen Wert zurück.)

Im Folgenden sehen Sie eine Liste von Eigenschaften und Methoden, die für Tabellenoperationen zur Verfügung stehen. Jede Eigenschaft gibt einen booleschen Wert zurück und die Methoden haben keine Parameter.

Eigenschaft	Methode	Tabellenoperationen
IsRowInsertEnabled	**RowInsert**2016, wird nun ersetzt durchAuthenticRange.InsertRow1789	Operation "Zeile einfügen"

IsRowAppendEnabled	**RowAppend**[2014], wird nun ersetzt durch AuthenticRange.AppendRow[1775]	Operation "Zeile anhängen"
IsRowDeleteEnabled	**RowDelete**[2015], wird nun ersetzt durch AuthenticRange.DeleteRow[1778]	Operation "Zeile löschen"
IsRowMoveUpEnabled	**RowMoveUp**[2017], wird nun ersetzt durch AuthenticRange.MoveRowUp[1797]	XML-Daten eine Zeile nach oben verschieben
IsRowMoveDownEnabled	**RowMoveDown**[2016], wird nun ersetzt durch AuthenticRange.MoveRowDown[1796]	XML-Daten eine Zeile nach unten verschieben
IsRowDuplicateEnabled	**RowDuplicate**[2015], wird nun ersetzt durch AuthenticRange.DuplicateRow[1779]	Aktuell ausgewählte Zeile duplizieren

29.3.1.5 Nicht mehr verwendete Bearbeitungsoptionen in der Authentic-Ansicht

Wenn XML-Daten als Daten in der Authentic-Ansicht angezeigt werden, können einzelne Elemente mit Hilfe von Standardbearbeitungsoperationen wie Ausschneiden, Kopieren und Einfügen bearbeitet werden. Es können jedoch nicht alle XML-Daten-Nodes bearbeitet werden. Um daher eine Bearbeitungsoperation durchzuführen, muss zuerst mit Hilfe einer Eigenschaft überprüft werden, ob eine Bearbeitung möglich ist. Anschließend wird eine Methode aufgerufen, um die Bearbeitungsoperation durchzuführen.

Die einzige Methode, für die es keinen Test gibt, ist die Methode EditSelectAll, die automatisch alle im Dokument angezeigten Elemente auswählt.

Im Folgenden sehen Sie eine Liste der Eigenschaften und Methoden zur Durchführung von Bearbeitungsoperationen. Jede Methode gibt einen boolschen Wert zurück und die Methoden haben keine Parameter.

Eigenschaft	Methode	Bearbeitungsoperation
IsEditUndoEnabled	**EditUndo**[2002], wird nun ersetzt durch AuthenticView.Undo[1821]	Bearbeitungsoperation rückgängig machen
IsEditRedoEnabled	**EditRedo**[2001], wird nun ersetzt durch AuthenticView.Redo[1820]	Bearbeitungsoperation wiederholen
IsEditCopyEnabled	**EditCopy**[2000], wird nun ersetzt durch AuthenticRange.Copy[1777]	Ausgewählten Text in die Zwischenablage kopieren

© 2016-2022 Altova GmbH

IsEditCutEnabled	**EditCut**[2000], wird nun ersetzt durch `AuthenticRange.Cut`[1778]	Ausgewählten Text ausschneiden und in der Zwischenablage ablegen
IsEditPasteEnabled	**EditPaste**[2001], wird nun ersetzt durch `AuthenticRange.Paste`[1797]	Aus der Zwischenablage an der aktuellen Cursorposition einfügen
IsEditClearEnabled	**EditClear**[1999], wird nun ersetzt durch `AuthenticRange.Delete`[1778]	Ausgewählten Text aus dem XML-Dokument löschen

29.3.2 Schnittstellen

Objekthierarchie
Application[1746]
SpyProject[1920]
SpyProjectItems[1924]
SpyProjectItem[1921]
Documents[1870]
Document[1838]
GridView[1902]
AuthenticView[1803]
AuthenticRange[1773]
AuthenticDataTransfer[1768](vormals DocEditDataTransfer)
OldAuthenticView[1996](vormals DocEditView, **wird nicht mehr verwendet**, jetzt ersetzt durch AuthenticView[1803] und AuthenticRange[1773])
AuthenticSelection[1994](vormals DocEditSelection, **wird nicht mehr verwendet**, jetzt ersetzt durch AuthenticRange[1773])
AuthenticEvent[1981](vormals DocEditEvent, **wird nicht mehr verwendet**)
AuthenticDataTransfer[1768](vormals DocEditDataTransfer)
TextView[1928]
XMLData[1970]
Dialogs[1752]
CodeGeneratorDlg[1822]
FileSelectionDlg[1883]
SchemaDocumentationDlg[1906]
GenerateSampleXMLDlg[1896]
DTDSchemaGeneratorDlg[1874]
FindInFilesDlg[1885]
WSDLDocumentationDlg[1934]
WSDL20DocumentationDlg[1946]
XBRLDocumentationDlg[1958]
DatabaseConnection[1828]
ExportSettings[1880]
TextImportExportSettings[1925]
ElementList[1878]
ElementListItem[1879]

Enumerations[2020]

Beschreibung
Dieses Kapitel enthält die Referenz der XMLSpy1.5 Typbibliothek.

Die meisten der hier beschriebenen Beispiele sind in VisualBasic verfasst. Bei diesen Codefragmenten wird davon ausgegangen, dass eine Variable namens **objSpy vom Typ Application** definiert und gesetzt ist. Einige der Codebeispiele wurden auch in JavaScript geschrieben.

29.3.2.1 Application

Methoden
GetDatabaseImportElementList 1753
GetDatabaseSettings 1753
GetDatabaseTables 1754
ImportFromDatabase 1757
CreateXMLSchemaFromDBStructure 1751

GetTextImportElementList 1755
GetTextImportExportSettings 1756
ImportFromText 1758

ImportFromWord 1759

ImportFromSchema 1758

GetExportSettings 1754

NewProject 1760
OpenProject 1761

AddMacroMenuItem 1750
ClearMacroMenu 1751

ShowForm 1764

ShowApplication 1763

URLDelete 1764
URLMakeDirectory 1765

AddXSLT_XQParameter 1750
GetXSLT_XQParameterCount 1756
GetXSLT_XQParameterName 1756
GetXSLT_XQParameterXPath 1756
RemoveXSLT_XQParameter 1762

FindInFiles 1752

Quit 1761

Eigenschaften

© 2016-2022 Altova GmbH

Application[1750]
Parent[1761]

ActiveDocument[1749]
Documents[1752]

CurrentProject[1751]

Dialogs[1752]

WarningNumber[1765]
WarningText[1766]

Status[1764]
MajorVersion[1760]
MinorVersion[1760]
Edition[1752]
IsAPISupported[1759]
ServicePackVersion[1763]

Beschreibung
Application ist die Root für alle anderen Objekte. Es ist das einzige Objekt, das Sie mit CreateObject(VisualBasic) oder anderen ähnlichen COM-Funktionen erstellen können.

Beispiel

```
DimobjSpy AsApplication
SetobjSpy = CreateObject("XMLSpy.Application")
```

29.3.2.1.1 Events

29.3.2.1.1.1 OnBeforeOpenDocument

Event: OnBeforeOpenDocument(*objDialog* als FileSelectionDlg[1883])

Beschreibung
Dieses Event wird immer dann ausgelöst, wenn ein Dokument mit dem Menübefehl OpenFile oder OpenURL geöffnet wird. Es wird nach dem Auswählen einer Dokumentdatei aber vor dem Öffnen des Dokuments gesendet. Das Dialogfeldobjekt zur Auswahl der Datei wird mit dem Namen der ausgewählten Dokumentdatei initialisiert. Sie können diese Auswahl ändern. Um mit dem Öffnen des Dokuments fortzufahren, belassen Sie die Eigenschaft FileSelectionDlg.DialogAction[1884] von *io_objDialog* auf dem Standardwert spyDialogOK[2023]. Um das Öffnen des Dokuments abzubrechen, setzen Sie die Eigenschaft auf spyDialogCancel[2023].

Beispiele
Im Folgenden finden Sie Beispiele dafür, wie ein Skript für dieses Event erstellt werden kann.

XMLSpy scripting environment - VBScript:

```
Function On_BeforeOpenDocument(objDialog)
End Function
```

XMLSpy scripting environment - JScript:
function On_BeforeOpenDocument(*objDialog*)
{
}

XMLSpy IDE Plugin:
IXMLSpyPlugIn.OnEvent(26, ...) // nEventId = 26

29.3.2.1.1.2 OnBeforeOpenProject

Event: OnBeforeOpenProject(*objDialog* als FileSelectionDlg[1883])

Beschreibung
Dieses Event wird ausgelöst, wenn eine Projektdatei ausgewählt, aber bevor sie geöffnet wird. Das Dialogfeldobjekt zur Auswahl der Datei wird mit dem Namen der ausgewählten Projektdatei initialisiert. Sie können diese Auswahl ändern. Um mit dem Öffnen des Projekts fortzufahren, belassen Sie die Eigenschaft FileSelectionDlg.DialogAction[1884] von *io_objDialog* auf dem Standardwert spyDialogOK[2023]. Um das Öffnen das Projekts abzubrechen, setzen Sie die Eigenschaft auf spyDialogCancel[2023].

Beispiele
Im Folgenden sehen Sie einige Beispiele, wie ein Skript für dieses Event erstellt werden kann.

XMLSpy scripting environment - VBScript:
Function On_BeforeOpenProject(*objDialog*)
End Function

XMLSpy scripting environment - JScript:
function On_BeforeOpenProject(*objDialog*)
{
}

XMLSpy IDE Plugin:
IXMLSpyPlugIn.OnEvent(25, ...) // nEventId = 25

29.3.2.1.1.3 OnDocumentOpened

Event: OnDocumentOpened(*objDocument* als Document[1838])

Beschreibung
Dieses Event wird ausgelöst, wenn ein Dokument in XMLSpygeöffnet wird. Voraussetzung ist, dass eine Datei über das Dialogfeld OpenFile oder OpenURL geöffnet oder eine neu Datei erstellt oder eine Datei in XMLSpygezogen wird. Das neue Dokument wird als Parameter übergeben. Die Operation kann nicht abgebrochen werden.

Beispiele
Im Folgenden finden Sie Beispiele dafür, wie ein Skript für dieses Event erstellt werden kann.

 © 2016-2022 Altova GmbH

XMLSpy Skripting-Umgebung - VBScript:
Function On_OpenDocument(*objDocument*)
End Function

XMLSpy Skripting-Umgebung - JScript:
function On_OpenDocument(*objDocument*)
{
}

XMLSpy IDE Plugin:
IXMLSpyPlugIn.OnEvent(7, ...) // nEventId = 7

29.3.2.1.1.4 OnProjectOpened

Event: OnProjectOpened(*objProject* als SpyProject[1920])

Beschreibung
Dieses Event wird ausgelöst, wenn ein Projekt in XMLSpygeöffnet wird. Das neue Projekt wird als Parameter übergeben.

Beispiele
Im Folgenden finden Sie Beispiele dafür, wie ein Skript für dieses Event erstellt werden kann.

XMLSpy Skripting-Umgebung - VBScript:
Function On_OpenProject(*objProject*)
End Function

XMLSpy Skripting-Umgebung - JScript:
function On_OpenProject(*objProject*)
{
}

XMLSpy IDE Plugin:
IXMLSpyPlugIn.OnEvent(6, ...) // nEventId = 6

29.3.2.1.2 ActiveDocument

Eigenschaft: ActiveDocumentals Document[1838]

Beschreibung
Referenz auf das aktive Dokument. Wenn kein Dokument geöffnet ist, ist ActiveDocumentnull (nothing).

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.3 AddMacroMenuItem

Methode: AddMacroMenuItem(*strMacro*als String,*strDisplayText*als String)

Beschreibung
Fügt einen Menübefehl zum Menü **Extras** hinzu. Dieser neue Menübefehl ruft das von strMacrodefinierte Makro auf. Siehe auch "Beispiel: Skripting-Projekt[1675]".

Fehler

1111	Das application-Objekt ist nicht mehr gültig.
1100	Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.
1108	Die Anzahl der Makroelemente ist auf 16 beschränkt.

29.3.2.1.4 AddXSLT_XQParameter

Methode: AddXSLT_XQParameter(name als String, XPath als String)

Beschreibung
Fügt einen XSLT- oder XQuery-Parameter hinzu. Der Name und Wert des Parameters sind die beiden Argumente der Methode.

Fehler

1111	Das application-Objekt ist nicht mehr gültig.
1100	Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.
1124	Der XPath-Ausdruck wurde nicht definiert.
1125	Kein QName.
1126	Der definierte XPath ist nicht gültig. Der Grund dafür ist angehängt.
1127	Es existiert bereits ein Parameter mit dem angegebenen Namen.

29.3.2.1.5 Application

Eigenschaft: Applicationals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler

1111	Das application-Objekt ist nicht mehr gültig.
1100	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.1.6 ClearMacroMenu

Methode: ClearMacroMenu()

Rückgabewert
Keiner

Beschreibung
Entfernt alle Menübefehle, die durch Aufruf von `AddMacroMenuItem` [1750] hinzugefügt wurden, aus dem Menü **Extras**. Siehe auch Beispiel:Skripting-Projekt [1675].

Fehler
1111 Das application-Objekt ist nicht mehr gültig.

29.3.2.1.7 CreateXMLSchemaFromDBStructure

Methode: CreateXMLSchemaFromDBStructure(pImportSettingsals DatabaseConnection [1828], pTablesals ElementList [1878])

Beschreibung
CreateXMLSchemaFromDBStructureerstellt anhand der in pImportSettingsdefinierten Datenbank für die in pTables definierten Tabellen ein neues/neue Schema-Dokument(e), das/die die Struktur der Datenbanktabellen beschreibt/beschreiben.

Der Parameter pTablesdefiniert, welche Tabellenstrukturen das XML-Schema-Dokument enthalten soll. Dieser Parameter kann NULL sein und definiert in diesem Fall, dass alle Tabellenstrukturen exportiert werden sollen.

Siehe auch GetDataBaseTables [1754].

Fehler
1112 Es wurde eine ungültige Datenbank definiert.
1120 Der Datenbankimport ist fehlgeschlagen.

29.3.2.1.8 CurrentProject

Eigenschaft: CurrentProjectals SpyProject [1920]

Beschreibung
Referenz auf das aktive Dokument. Wenn kein Projekt geöffnet ist, ist CurrentProjectgleich Null (nothing).

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.9 Dialogs

Eigenschaft: Dialogsals Dialogs[1835](schreibgeschützt)

Beschreibung
Ruft die integrierten Dialogfelder von XMLSpy auf.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.10 Documents

Eigenschaft: Documentsals Documents[1870]

Beschreibung
Sammlung aller offenen Dokumente.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.11 Edition

Eigenschaft: Editionals String

Beschreibung
Gibt die Edition der Applikation, also z.B. "`Altova XMLSpy Enterprise Edition`" für die Enterprise Editionzurück.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.12 FindInFiles

Method: FindInFiles(*pSettings*als FindInFilesDlg[1885]) als FindInFilesResults[1895]

Beschreibung
Gibt ein FindInFilesResults[1895]Objekt zurück, das Informationen über die Dateien enthält, die den angegebenen Kriterien entsprechen.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.1.13 GetDatabaseImportElementList

Methode: GetDatabaseImportElementList(*pImportSettings*als DatabaseConnection[1828])als ElementList[1878]

Beschreibung
Die Funktion gibt eine Sammlung von ElementListItems zurück, wobei die ElementListItem.Name[1880] Eigenschaften die Namen der Felder enthalten, die für den Import ausgewählt werden können und die ElementListItem.ElementKind[1879] Eigenschaften, je nachdem, welcher Wert in DatabaseConnection.AsAttributes[1830] übergeben wird, entweder auf *spyXMLDataAttr* oder *spyXMLDataElement* initialisiert werden. Diese Liste dient als Filter dafür, was schließlich durch einen zukünftigen Aufruf von ImportFromDatabase[1757] importiert wird. Verwenden Sie ElementList.RemoveElement[1879] um Felder vom Export auszuschließen.

Für die Herstellung der Verbindung zur Datenbank muss eines der Objekte DatabaseConnection.File[1831], DatabaseConnection.ADOConnection[1829]oder DatabaseConnection.ODBCConnection[1833]sowie DatabaseConnection.SQLSelect[1834]ausgefüllt werden. Verwenden Sie die Eigenschaft DatabaseConnection.AsAttributes[1830]um ElementListItem.ElementKind[1879]der erzeugten Elementliste entweder auf *spyXMLDataAttr* oder *spyXMLDataElement* zu initialisieren.

Beispiel
Siehe Beispiel unter ImportFromDatabase[1757].

Fehler

1111	Das application-Objekt ist nicht mehr gültig.
1100	Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.
1107	Import aus Datenbank ist fehlgeschlagen.
1112	Es wurde eine ungültige Datenbank angegeben.
1114	Select-Anweisung fehlt.
1119	Die Liste der Datenbankelemente konnte nicht importiert werden.

29.3.2.1.14 GetDatabaseSettings

Methode: GetDatabaseSettings() als DatabaseConnection[1828]

Beschreibung
GetDatabaseSettingserstellt ein neues Objekt für Datenbankeinstellungen. Mit Hilfe des Objekts werden die Datenbankverbindungsparameter für die Methoden GetDatabaseTables[1754], GetDatabaseImportElementList[1753], ImportFromDatabase[1757], ImportFromSchema[1758]und ExportToDatabase[1851] definiert.

Beispiel
Siehe Beispiel von ImportFromDatabase[1757].

Fehler

1111	Das application-Objekt ist nicht mehr gültig.
1100	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.15 GetDatabaseTables

Methode: GetDatabaseTables(*pImportSettings*als DatabaseConnection[1746])als ElementList[1878]

Beschreibung
GetDatabaseTablesliest die Tabellennamen aus der in *pImportSettings*definierten Datenbank. Für die Datenbankverbindung muss eine der folgenden Eigenschaften ausgefüllt werden DatabaseConnection.File[1831], DatabaseConnection.ADOConnection[1829]oder DatabaseConnection.ODBCConnection[1833]. Alle anderen Eigenschaften werden ignoriert.
Die Funktion gibt eine Sammlung von ElementListItems zurück, wobei die ElementListItem.Name[1880] Eigenschaften die Namen der in der angegebenen Datenbank gespeicherten Tabellen enthält. Die restlichen Eigenschaften von ElementListItem[1879] werden nicht verwendet.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.
1112 Es wurde eine ungültige Datenbank angegeben
1113 Fehler beim Lesen der Datenbanktabellendaten.
1118 Die Abfrage der Datenbanktabelle ist fehlgeschlagen.

Beispiel

```
    DimobjImpSettings AsDatabaseConnection
SetobjImpSettings = objSpy.GetDatabaseSettings
objImpSettings.ADOConnection = TxtADO.Text

'store table names in list box
ListTables.Clear

DimobjList AsElementList
DimobjItem AsElementListItem
OnErrorGoToErrorHandler
SetobjList = objSpy.GetDatabaseTables(objImpSettings)

    ForEachobjItem InobjList
        ListTables.AddItem objItem.Name
    Next
```

29.3.2.1.16 GetExportSettings

Methode: GetExportSettings()als ExportSettings[1880](schreibgeschützt)

Beschreibung
GetExportSettingserstellt ein neues Objekt allgemeiner Exporteinstellungen. Mit Hilfe dieses Objekts werden die Parameter an die Exportfunktionen übergeben und das Verhalten der Exportaufrufe wird definiert. Siehe auch die Exportfunktionen aus Document[1838]und die Beispiele unter Import und Export[1715].

Fehler
1111 Das application-Objekt ist nicht mehr gültig.

© 2016-2022 Altova GmbH

1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.17 GetTextImportElementList

Methode: GetTextImportElementList(pImportSettingsals TextImportExportSettings[1925])als ElementList[1878]

Beschreibung
GetTextImportElementListruft Importinformationen über die in pImportSettings definierte Textdatei ab. Die Funktion gibt eine Sammlung von ElementListItems zurück, wobei die ElementListItem.Name[1880] Eigenschaften die Namen der in der Datei gefundenen Felder enthält. Die Werte der restlichen Eigenschaften werden nicht definiert.

Wenn die Textdatei keine Spaltenüberschriften enthält, setzen Sie pImportSettings.HeaderRow[1927] auf false. Die erzeugte Elementliste enthält allgemeine Spaltennamen wie 'Field1' usw.

Siehe auch Import und Export von Daten[1715].

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.
1107 Import aus Datenbank ist fehlgeschlagen.
1115 Fehler beim Import der Textelementliste. Für die Importdatei kann kein Parser erstellt werden.
1116 Fehler beim Import der Textelementliste.

Beispiel

```
' ----------------------------------------------------------
' VBA client code fragment - import selected fields from text file
' ----------------------------------------------------------
        DimobjImpSettings AsTextImportExportSettings
        SetobjImpSettings = objSpy.GetTextImportExportSettings

        objImpSettings.ImportFile = "C:\ImportMe.txt"
        objImpSettings.HeaderRow = False

        DimobjList AsElementList
        SetobjList = objSpy.GetTextImportElementList(objImpSettings)

        'exclude first column
        objList.RemoveItem 1

        DimobjImpDoc AsDocument
        OnErrorResume Next
        SetobjImpDoc = objSpy.ImportFromText(objImpSettings, objList)
        CheckForError
```

29.3.2.1.18 GetTextImportExportSettings

Methode: GetTextImportExportSettings() als TextImportExportSettings 1925 (schreibgeschützt)

Beschreibung
GetTextImportExportSettingserstellt ein neues Objekt der allgemeinen Import- udn Exporteinstellungen für Textdateien. Siehe auch Beispiel zu Application.GetTextImportElementList 1755 und Import and Export 1715.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.19 GetXSLT_XQParameterCount

Methode: GetXSLT_XQParameterCount()als Long

Beschreibung
Gibt die Anzahl der XSLT- und XQuery-Parameter zurück.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.20 GetXSLT_XQParameterName

Methode: GetXSLT_XQParameterName(index as Long)als String

Beschreibung
Gibt den Namen des durch den angegebenen Index identifizierten XSLT- oder XQuery-Parameters zurück.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.21 GetXSLT_XQParameterXPath

Methode: GetXSLT_XQParameterXPath(index als Long)als String

Beschreibung
Gibt den XPath-Ausdruck des durch den angegebenen Index identifizierten XSLT- oder XQuery-Parameters zurück.

© 2016-2022 Altova GmbH

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.22 ImportFromDatabase

Methode: ImportFromDatabase(pImportSettingsals DatabaseConnection[1828],pElementListals ElementList[1878]) als Document[1838]

Rückgabewert
Erstellt ein neues Dokument, das die aus der Datenbank importierten Daten enthält.

Beschreibung
ImportFromDatabaseimportiert Daten aus einer Datenbank, wie in pImportSettingsdefiniert und erstellt ein neues Dokument, das diese Daten enthält. Eine der folgenden Eigenschaften muss dazu ausgefüllt werden: DatabaseConnection.File[1831], DatabaseConnection.ADOConnection[1829]oder DatabaseConnection.ODBCConnection[1833]und DatabaseConnection.SQLSelect[1834]. Zusätzlich dazu können Sie mittels DatabaseConnection.AsAttributes[1830], DatabaseConnection.ExcludeKeys[1831], DatabaseConnection.IncludeEmptyElements[1832]und NumberDateTimeFormat[1833]weitere Parameter für den Import eingeben.

Der Parameter pElementListdefiniert, ob Felder der ausgewählten Daten in das neu erstellte Dokument geschrieben werden sollen und welche davon als Elemente und welche als Attribute erstellt werden sollen. Dieser Parameter kann NULL sein, d.h. alle ausgewählten Felder werden als XML-Elemente importiert.

Erforderliche Schritte, die vor dem Import von Daten aus einer Datenbank durchzuführen sind, werden unter GetDatabaseSettings[1753]und GetDatabaseImportElementList[1753]beschrieben.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.
1107 Import aus Datenbank ist fehlgeschlagen.
1112 Es wurde eine ungültige Datenbank angegeben.
1114 Select-Anweisung fehlt.
1117 Transformation in XML ist fehlgeschlagen.
1120 Datenbankimport ist fehlgeschlagen.

Beispiel

```
    DimobjImpSettings AsDatabaseConnection
    SetobjImpSettings = objSpy.GetDatabaseSettings

    objImpSettings.ADOConnection = strADOConnection
    objImpSettings.SQLSelect = "SELECT * FROM MyTable"

    DimobjDoc AsDocument
    OnErrorResume Next
    SetobjDoc = objSpy.ImportFromDatabase(objImpSettings,
objSpy.GetDatabaseImportElementList(objImpSettings))
```

```
' CheckForError here
```

29.3.2.1.23 ImportFromSchema

Methode: ImportFromSchema(*pImportSettings*als DatabaseConnection[1828],*strTable*als String,*pSchemaDoc*als Document[1838])als Document[1838]

Rückgabewert
Erstellt ein neues Dokument, das die aus der Datenbank gemäß der Schemadefinition in *pSchemaDoc*importierten Daten enthält.

Beschreibung
ImportFromSchema importiert Daten aus einer in pImportSettingsdefinierten Datenbank. Dazu muss eine der folgenden Eigenschaften ausgefüllt werden DatabaseConnection.File[1831], DatabaseConnection.ADOConnection[1829]oder DatabaseConnection.ODBCConnection[1833]. Zusätzlich dazu können Sie mittels DatabaseConnection.AsAttributes[1830], DatabaseConnection.ExcludeKeys[1831]und NumberDateTimeFormat[1833]weitere Parameter für den Import eingeben. Alle anderen Parameter werden ignoriert.

ImportFromSchema verwendet zum Auswählen der Daten keine explizite SQL-Anweisung. Statt dessen muss eine Strukturdefinition des Dokuments in Form eines XML-Schema-Dokuments in *pSchemaDoc*erstellt werden. Anhand dieser Definition wird automatisch die Datenbank-Select-Anweisung abgeleitet. Definieren Sie in *strTable*den Tabellennamen der Import-Root, die der Root-Node im neuen Dokument werden soll.

Erforderliche Schritte, die vor dem Import von Daten aus einer Datenbank anhand einer Schemadefinition durchzuführen sind, werden unter GetDatabaseSettings[1753]und GetDatabaseTables[1754]beschrieben. Verwenden Sie zum Erstellen einer Schemadefinitionsdatei den Befehl "Datenbankschema erstellen" aus dem XMLSpy-Menü "Konvertieren".

Fehler

1111	Das application-Objekt ist nicht mehr gültig.
1100	Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.
1107	Import aus Datenbank ist fehlgeschlagen.
1112	Es wurde eine ungültige Datenbank angegeben.
1120	Datenbankimport ist fehlgeschlagen.
1121	Es konnte kein Validator für das angegebene Schema erstellt werden.
1122	Fehler beim Parsen des Schemas für den Datenbankimport.

29.3.2.1.24 ImportFromText

Methode: ImportFromText(*pImportSettings*als TextImportExportSettings[1925],*pElementList*als ElementList[1878]) als Document[1838]

Beschreibung
ImportFromTextimportiert die Textdatei, wie in pImportSettingsdefiniert. Der Parameter pElementListkann als Importfilter verwendet werden. Übergeben Sie entweder die durch einen vorherigen Aufruf von GetTextImportElementList[1755]zurückgegebene Liste oder den Wert nullum alle Spalten zu importieren. Um den

© 2016-2022 Altova GmbH

Import unnötiger Spalten zu vermeiden, entfernen Sie die entsprechenden Feldnamen mit Hilfe von ElementList.RemoveElement[1879] aus pElementListbevor Sie ImportFromTextaufrufen.
Die Methode gibt das neu erstellte Dokument mit den importierten Daten zurück. Dieses Dokument ist dasselbe wie das aktive Dokument von XMLSpy.

Siehe auch Import und Export von Daten[1715].

Fehler

1111	Das application-Objekt ist nicht mehr gültig.
1100	Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.
1107	Import aus Textdatei ist fehlgeschlagen
1117	Transformation in XML ist fehlgeschlagen.

Beispiel

```
' ----------------------------------------------------------
' VBA client code fragment - import from text file
' ----------------------------------------------------------
        DimobjImpSettings AsTextImportExportSettings
        SetobjImpSettings = objSpy.GetTextImportExportSettings

        objImpSettings.ImportFile = strFileName
        objImpSettings.HeaderRow = False

        DimobjImpDoc AsDocument
        OnErrorResume Next
        SetobjImpDoc = objSpy.ImportFromText(objImpSettings,
objSpy.GetTextImportElementList(objImpSettings))

        CheckForError
```

29.3.2.1.25 ImportFromWord

Methode: ImportFromWord(*strFile*als String) als Document[1838]

Beschreibung
ImportFromWordimportiert das MS Word-Dokument strFilein ein neues XML-Dokument.

Fehler

1111	Das application-Objekt ist nicht mehr gültig.
1100	Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben. Import aus Dokument ist fehlgeschlagen.

29.3.2.1.26 IsAPISupported

Eigenschaft: IsAPISupportedals Boolean

Beschreibung
Gibt zurück, ob die API in dieser Version unterstützt wird oder nicht.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.27 MajorVersion

Eigenschaft: MajorVersionals Integer

Beschreibung
Gibt die Hauptreleasenummer der Applikation zurück, z.B. `15`für 2013 Versionen und `16`für 2014 Versionen.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.28 MinorVersion

Eigenschaft: MinorVersionas Integer

Beschreibung
Gibt die Nebenrelease-Versionsnummer der Applikation zurück.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.29 NewProject

Methode: NewProject(*strPath*als String,*bDiscardCurrent*als Boolean)

Beschreibung
NewProjecterstellt ein neues Projekt.

Wenn es bereits ein offenes Projekt gibt, das geändert wurde undbDiscardCurrent"false" ist, schlägt NewProject()fehl.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1102 Es ist bereits ein Projekt geöffnet, aber *bDiscardCurrent*ist *true*.
1103 Das neue Projekt konnte nicht erstellt werden.

© 2016-2022 Altova GmbH

29.3.2.1.30 OpenProject

Methode: OpenProject(*strPath*als String,*bDiscardCurrent*als Boolean,*bDialog*als Boolean)

Parameter
strPath
Der Pfad und Dateiname des zu öffnenden Projekts. Kann leer sein, wenn bDialog"true" ist.

bDiscardCurrent
Aktuell geöffnetes Projekt verwerfen. Änderungen daran gehen verloren.

bDialog
Dialogfelder für Benutzereingabe anzeigen.

Rückgabewert
Keiner

Beschreibung
OpenProjectöffnet ein vorhandenes Projekt. Wenn bereits ein Projekt geöffnet ist, das geändert wurde und bDiscardCurrent"false" ist, schlägt OpenProject()fehl.

Fehler

1111	Das application-Objekt ist nicht mehr gültig.
1100	Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.
1101	Das angegebene Projekt kann nicht geöffnet werden.
1102	Es ist bereits ein Projekt geöffnet, aber *bDiscardCurrent*ist *true*.

29.3.2.1.31 Parent

Eigenschaft: Parent als Application 1746 (schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler

1111	Das application-Objekt ist nicht mehr gültig.
1100	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.32 Quit

Method: Quit()

Rückgabewert
Keiner

Beschreibung

Diese Methode beendet XMLSpy. Alle geänderten Dokumente werden ohne Speicherung der Änderungen geschlossen. Dies gilbt auch für ein offenes Projekt.

Wenn XMLSpy automatisch von einem Client-Programm als Automation Server gestartet wird, wird die Applikation bei Schließen des Client-Programms nicht automatisch geschlossen, falls noch ein Projekt oder Dokument geöffnet ist. Verwenden Sie die Quit-Methode, um sicherzustellen, dass das Programm automatisch beendet wird.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.

29.3.2.1.33 ReloadSettings

Methode: ReloadSettings

Rückgabewert

Beschreibung
Die Applikationseinstellungen werden aus der Registrierungsdatei neu geladen.

Verfügbar mit Typbibliothek Version 1.5

Fehler
1111 Das application-Objekt ist nicht mehr gültig.

29.3.2.1.34 RemoveXSLT_XQParameter

Methode: RemoveXSLT_XQParameter(index als Long)

Beschreibung
Entfernt den durch den angegebenen Index identifizierten XSLT- oder XQuery-Parameter.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.35 RunMacro

Methode: RunMacro(*strMacro*als String)

Rückgabewert

Beschreibung
Ruft das angegebene Makro entweder von den Projektskripts (falls vorhanden) oder den globalen Skripts aus auf.

Verfügbar mit Typbibliothek Version 1.5

© 2016-2022 Altova GmbH

Fehler
1111 Das application-Objekt ist nicht mehr gültig.

29.3.2.1.36 ScriptingEnvironment

Eigenschaft: ScriptingEnvironment als IUnknown (schreibgeschützt)

Beschreibung
Referenz auf jede aktive Skripting-Umgebung. Mit Hilfe dieser Eigenschaft kann die TypeLibrary der XMLSpyFormEditor.exe Applikation, die als aktuelle Skripting-Umgebung verwendet wird, aufgerufen werden.

Verfügbar mit Typbibliothek Version 1.5

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.37 ServicePackVersion

Eigenschaft: ServicePackVersionals Long

Beschreibung
Gibt die Service Pack-Versionsnummer der Applikation zurück, also z.B. 1 für 2010 R2 SP1

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.38 ShowApplication

Methode: ShowApplication(*bShow*als Boolean)

Rückgabewert
Keiner

Beschreibung
Die Methode blendet XMLSpy ein(bShow = True) oder aus (bShow = False).

Fehler
1110 Das application-Objekt ist nicht mehr gültig.

29.3.2.1.39 ShowFindInFiles

Methode: ShowFindInFiles(pSettingsals FindInFilesDlg[1885]) als Boolean

Rückgabewert

Gibt "false" zurück, wenn der Benutzer die Schaltfläche "Abbrechen" gedrückt hat, andernfalls "true".

Beschreibung
Zeigt das Dialogfeld FindInFiles mit den vordefinierten Einstellungen an. Die benutzerdefinierten Änderungen an den Einstellungen werden im übergebenen Dialogfeldobjekt gespeichert.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.

29.3.2.1.40 ShowForm

Methode: ShowForm(*strFormName*als String) als Long

Rückgabewert
Gibt Null zurück, wenn der Benutzer eine Abbrechen-Schaltfläche betätigt hat, oder die Formularaufrufe TheView.Cancel().

Beschreibung
Zeigt das Formular strFormNamean.

Formulare, Events und Event Handler können mit der Skripting Umgebung erstellt werden. Wählen Sie im Menü **Extras**den Befehl "Skripting-Umgebung...", um die Skripting-Umgebung aufzurufen..

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben..

29.3.2.1.41 Status

Eigenschaft: Statusals ENUMApplicationStatus 2020

Beschreibung
Gibt den aktuellen Status der laufenden Applikation zurück.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.42 URLDelete

Methode: URLDelete(*strURL*als String,*strUser*als String,*strPassword*als String)

Rückgabewert
Keiner

© 2016-2022 Altova GmbH

Beschreibung
Die Methode löscht die Datei an der URL strURL.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1109 Fehler beim Löschen der Datei an der angegebenen URL.

29.3.2.1.43 URLMakeDirectory

Methode: URLMakeDirectory(*strURL*als String,*strUser*als String,*strPassword*als String)

Rückgabewert
Keiner

Beschreibung
Die Methode erstellt ein neues Verzeichnis an der URL strURL.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Es wurde ein ungültiger Parameter angegeben.

29.3.2.1.44 Visible

Eigenschaft: Visibleals VARIANT_BOOL

Beschreibung
Setzt oder ruft das Sichbarkeitsattribut von XMLSpyab. Diese Standard-Automation-Eigenschaft macht die Verwendung von ShowApplication[1763] überflüssig.

Fehler
1110 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.45 WarningNumber

Eigenschaft: WarningNumberals integer

Beschreibung
Einige Methoden füllen die Eigenschaft WarningNumbermit zusätzlichen Informationen, wenn ein Fehler auftritt.

Derzeit wird diese Eigenschaft nur mit Documents.OpenFile[1872] befüllt.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.1.46 WarningText

Eigenschaft: WarningTextals String

Beschreibung
Einge Methoden befüllen die Eigenschaft WarningTextmit zusätzlichen Informationen, wenn ein Fehler auftritt.

Derzeit wird diese Eigenschaft nur mit Documents.OpenFile[1872] befüllt.

Fehler
1111 Das application-Objekt ist nicht mehr gültig.
1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.2 AuthenticContextMenu

Über die Kontextmenü-Schnittstelle können die in Authentic angezeigten Kontextmenüs angepasst werden. Die Schnittstelle hat die in diesem Abschnitt aufgelisteten Methoden.

29.3.2.2.1 CountItems

Methode: CountItems `()` nItems als long

Rückgabewert
Gibt die Anzahl der Menüoptionen zurück.

Fehler
2501 Ungültiges Objekt.

29.3.2.2.2 DeleteItem

Methode: DeleteItem `(IndexPosition als long)`

Rückgabewert
Löscht den Menüeintrag mit der im ersten Parameter übergebenen Indexposition.

Fehler
2501 Ungültiges Objekt
2502 Ungültiger Index

© 2016-2022 Altova GmbH

29.3.2.2.3 GetItemText

Methode: GetItemText `(IndexPosition als long)` Menübefehlname als String

Rückgabewert
Ruft den Namen des Menübefehls mit der im ersten Parameter übergebenen Indexposition ab.

Fehler
2501 Ungültiges Objekt
2502 Ungültiger Index

29.3.2.2.4 InsertItem

Methode: InsertItem `(IndexPosition als long, MenuItemName als string, MacroName als string)`

Rückgabewert
Fügt an der durch den ersten Parameter definierten Stelle im Menü einen benutzerdefinierten Menübefehl mit dem im zweiten Parameter definierten Namen ein. Der Menübefehl startet ein Makro, daher muss ein gültiger Makroname angegeben werden.

Errors
2501 Ungültiges Objekt
2502 Ungültiger Index
2503 Kein Makro dieses Namens vorhanden
2504 Interner Fehler

29.3.2.2.5 SetItemText

Methode: SetItemText `(IndexPosition als long, MenuItemName als string)`

Rückgabewert
Definiert den Namen des Menübefehls, der sich an der im ersten Parameter übergebenen Indexposition befindet.

Fehler
2501 Ungültiges Objekt
2502 Ungültiger Index

29.3.2.3 AuthenticDataTransfer

Umbenannt von DocEditDataTransfer in AuthenticDataTransfer

Das DocEditView Objekt wird in OldAuthenticView umbenannt.
DocEditSelection wird in AuthenticSelection umbenannt.
DocEditEvent wird in AuthenticEvent umbenannt.
DocEditDataTransfer wird in AuthenticDataTransfer umbenannt.

Es wird empfohlen, diese Objekte - mit Ausnahme von AuthenticDataTransfer - nicht mehr zu verwenden. Wir werden die bestehenden Funktionalitäten vorerst auf unbestimmte Zeit weiter unterstützen, doch werden keine neuen Funktionen zu diesen Schnittstellen hinzugefügt. Alle bisher in DocEditView[1996], DocEditSelection[1994], DocEditEvent[1981] und DocEditDataTransfer[1769] verfügbaren Funktionalitäten stehen nun über AuthenticView[1803], AuthenticRange[1773] und AuthenticDataTransfer[1768] zur Verfügung. Es wurden viele neue Funktionen hinzugefügt.

Beispiele zum Migrieren von DocEdit in Authentic finden Sie unter der Beschreibung der verschiedenen Methoden und Eigenschaften der einzelnen DocEdit-Objekte.

Methoden

getData[1769]

Eigenschaften

dropEffect[1768]
ownDrag[1769]
type[1769]

Beschreibung

Die Events OnDragOverund OnBeforeDropliefern mit einer Instanz des Typs AuthenticDataTransferInformationen über das gezogene Objekt. Sie enthält eine Beschreibung des gezogenen Objekts und seines Inhalts. Letzterer steht entweder als String oder als Pointer auf ein COM-Objekt, das die IUnkown-Schnittstelle unterstützt, zur Verfügung.

29.3.2.3.1 dropEffect

Eigenschaft: dropEffectals long

Beschreibung
Die Eigenschaft speichert den Drop-Effekt aus dem Standard-Event Handler. Sie können den Drop-Effekt definieren, wenn Sie diesen Wert ändern und für den Event Handler TRUE zurückgeben.(Oder wenn Sie AuthenticEvent.cancelBubble[1985]auf TRUEsetzen, wenn Sie noch die veraltete AuthenticEventSchnittstelle verwenden)

© 2016-2022 Altova GmbH

Fehler
2101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.3.2 getData

Methode: getData()als Variant

Beschreibung
Ruft die Daten im Zusammenhang mit dem gezogenen Objekt ab. Abhängig von AuthenticDataTransfer.type[1769] handelt es sich hierbei entweder um einen String oder um einen COM Interface Pointer vom Typ IUnknown.

Fehler
2101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.3.3 ownDrag

Eigenschaft: ownDragals Boolean (schreibgeschützt)

Beschreibung
Die Eigenschaft ist TRUE,wenn die aktuelle Quelle für das Ziehen des Objekts von innerhalb der Authentic-Ansicht kommt.

Fehler
2101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.3.4 type

Eigenschaft: typeals String (schreibgeschützt)

Beschreibung
Enthält den Datentyp, den Sie mit der DocEditDataTransfer.getData[1769] erhalten.

Derzeit werden die folgenden Datentypen unterstützt:

OWN	Daten aus der Authentic-Ansicht selbst
TEXT	Unformatierter Text
UNICODETEXT	Unformatierter Text als UNICODE

Fehler
2101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.4 AuthenticEventContext

Über die `EventContext` Schnittstellen haben Sie Zugriff auf viele Eigenschaften des Kontexts, in dem ein Makro ausgeführt wird.

29.3.2.4.1 EvaluateXPath

Methode: EvaluateXPath (strExpression als string) als strValue als string

Rückgabewert
Diese Methode wertet den XPath-Ausdruck im Kontext des Node aus, in dem das Event ausgelöst wurde, und gibt einen String zurück.

Beschreibung
`EvaluateXPath()` führt einen XPath-Ausdruck im angegebenen Event-Kontext aus. Das Ergebnis wird als String zurückgegeben. Im Fall einer Sequenz ist der String durch Leerzeichen getrennt.

Fehler
2201 Ungültiges Objekt.
2202 Kein Kontext.
2209 Ungültiger Parameter.
2210 Interner Fehler.
2211 XPath-Fehler.

29.3.2.4.2 GetEventContextType

Methode: GetEventContextType `()` Type als `AuthenticEventContextType` Enumeration

Rückgabewert
Gibt den Kontext-Node-Typ zurück.

Beschreibung
Mit Hilfe von `GetEventContextType` kann der Benutzer feststellen, ob sich das Makro in einem XML-Node oder in einem atomaren XPath Element-Kontext befindet. Die Enumeration `AuthenticEventContextType` ist folgendermaßen definiert:

authenticEventContextXML,
authenticEventContextAtomicItem,
authenticEventContextOther

Wenn der Kontext ein normaler XML-Node ist, erhalten Sie mit der Funktion `GetXMLNode()` Zugriff darauf (falls nicht, wird `NULL` zurückgegeben).

Fehler
2201 Ungültiges Objekt.
2202 Kein Kontext.
2209 Ungültiger Parameter.

© 2016-2022 Altova GmbH

29.3.2.4.3 GetNormalizedTextValue

Methode: GetNormalizedTextValue`()` strValue als String

Rückgabewert
Gibt den Wert des aktuellen Node als String zurück.

Fehler
2201 Ungültiges Objekt.
2202 Kein Kontext.
2203 Ungültiger Kontext
2209 Ungültiger Parameter.

29.3.2.4.4 GetVariableValue

Methode: GetVariableValue`(strName als string) strValue als string`

Rückgabewert
Ruft den Wert der als Parameter übergebenen Variablen ab.

Beschreibung
`GetVariableValue` ruft den Wert der Variablen im Geltungsbereich des Kontexts ab.

```
nZoom =parseInt(AuthenticView.EventContext.GetVariableValue('Zoom'));
if(nZoom >1)
{
    AuthenticView.EventContext.SetVariableValue('Zoom',nZoom -1);
}
```

Fehler
2201 Ungültiges Objekt.
2202 Kein Kontext.
2204 Keine Variable dieses Namens im Geltungsbereich
2205 Die Variable kann nicht ausgewertet werden
2206 Die Variable gibt eine Sequenz zurück
2209 Ungültiger Parameter

29.3.2.4.5 GetXMLNode

Methode: GetXMLNode`()` Node als XMLData Objekt

Rückgabewert
Gibt den XML-Kontext-Node oder `NULL` zurück

Fehler

2201 Ungültiges Objekt.
2202 Kein Kontext.
2203 Ungültiger Kontext
2209 Ungültiger Parameter.

29.3.2.4.6 IsAvailable

Methode: IsAvailable`()` als Boolean

Rückgabewert
Gibt "true" zurück, wenn `EventContext` definiert ist, andernfalls "false".

Fehler
2201 Ungültiges Objekt.

29.3.2.4.7 SetVariableValue

Methode: SetVariableValue`(strName als string, strValue als string)`

Rückgabewert
Definiert den Wert (den zweiten Parameter) der im ersten Parameter übergebenen Variablen.

Beschreibung
`SetVariableValue` definiert den Wert der Variablen im Geltungsbereich des Kontexts.

```
nZoom =parseInt(AuthenticView.EventContext.GetVariableValue('Zoom'));
if(nZoom >1)
{
    AuthenticView.EventContext.SetVariableValue('Zoom',nZoom -1);
}
```

Fehler
2201 Ungültiges Objekt.
2202 Kein Kontext.
2204 Keine Variable dieses Namens im Geltungsbereich
2205 Die Variable kann nicht ausgewertet werden
2206 Die Variable gibt eine Sequenz zurück
2207 Die Variable ist schreibgeschützt
2208 Keine Änderung zulässig

© 2016-2022 Altova GmbH

29.3.2.5 AuthenticRange

Die erste Tabelle enthält eine Liste der Eigenschaften und Methoden von AuthenticRange,die zum Navigieren durch das Dokument und zum Auswählen bestimmter Abschnitte verwendet werden.

Eigenschaften	Methoden	
Application 1775	Clone 1777	MoveBegin 1796
FirstTextPosition 1780	CollapsToBegin 1777	MoveEnd 1796
FirstXMLData 1781	CollapsToEnd 1777	NextCursorPosition 1787
FirstXMLDataOffset 1782	ExpandTo 1780	PreviousCursorPosition 1788
LastTextPosition 1793	Goto 1786	Select 1799
LastXMLData 1794	GotoNext 1786	SelectNext 1799
LastXMLDataOffset 1794	GotoPrevious 1787	SelectPrevious 1800
Parent 1797	IsEmpty 1791	SetFromRange 1802
	IsEqual 1791	

Die folgende Tabelle enthält eine Liste der Methoden zur Bearbeitung von Inhalt. Die meisten dieser Methoden können über Kontextmenüs aufgerufen werden.

Eigenschaften	Bearbeitungsoperationen	Dynamische Tabellenoperationen
Text 1802	Copy 1777	AppendRow 1775
	Cut 1778	DeleteRow 1778
	Delete 1778	DuplicateRow 1779
	IsCopyEnabled 1790	InsertRow 1789
	IsCutEnabled 1790	IsFirstRow 1791
	IsDeleteEnabled 1790	IsInDynamicTable 1791
	IsPasteEnabled 1792	IsLastRow 1792
	Paste 1797	MoveRowDown 1796
		MoveRowUp 1797

Die folgenden Methoden liefern die Funktionen der Authentic-Eingabehilfefenster für Bereichsobjekte.

Operationen der Eingabehilfefenster		
Elemente	**Attribute**	**Entities**
CanPerformActionWith 1776	GetElementAttributeValue 1784	GetEntityNames 1785
CanPerformAction 1776	GetElementAttributeNames 1783	InsertEntity 1789
PerformAction 1798	GetElementHierarchy 1784	
	HasElementAttribute 1788	
	IsTextStateApplied 1793	
	SetElementAttributeValue 1801	

Beschreibung
AuthenticRange Objekte sind die Cursor-Auswahl der Automation-Schnittstelle. Sie können damit auf jede Cursorposition in der Authentic-Ansicht zeigen oder einen Dokumentabschnitt auswählen. Die für AuthenticRange Objekte verfügbaren Operationen funktionieren anschließend an dieser Auswahl auf dieselbe Art wie die entsprechenden Operationen der Benutzeroberfläche an der aktuellen Auswahl des Benutzers. Der wichtigste Unterschied besteht darin, dass Sie beliebig viele AuthenticRange Objekte gleichzeitig verwenden können, während es auf der Benutzeroberfläche immer nur genau eine Cursorauswahl gibt.

Um zu Anfang ein Bereichsobjekt abzurufen, verwenden Sie AuthenticView.Selection 1820, um einen Bereich zu erhalten, der der aktuellen Cursorauswahl auf der Benutzeroberfläche entspricht. Alternativ dazu stehen einige einfache Bereiche über die schreibgeschützten Eigenschaften AuthenticView.DocumentBegin 1815, AuthenticView.DocumentEnd 1815 und AuthenticView.WholeDocument 1822 zur Verfügung. Die flexibelste Methode ist AuthenticView.Goto 1817. Sie können damit in einem Aufruf zu einem bestimmten Bereich des Dokuments navigieren. Um eine komplexere Auswahl zu treffen, kombinieren Sie die obigen Methoden mit den verschiedenen Navigationsmethoden zu den in der ersten Tabelle auf dieser Seite aufgelisteten Bereichsobjekten.

Eine weitere Methode, einen Dokumentabschnitt auszuwählen, ist die Positionseigenschaften des Bereichsobjekts zu verwenden. Es stehen zwei Systeme zur Positionierung zur Verfügung, die beliebig miteinander kombiniert werden können:

- Für den Beginn und das Ende eines Bereichs können **absolute** Textcursorpositionen, beginnend mit der Position 0 am Dokumentanfang, gesetzt und abgerufen werden. Nähere Informationen dazu finden Sie unter FirstTextPosition 1780 und LastTextPosition 1793. Diese Methode erfordert komplexe interne Berechnungen und sollte nur mit Vorsicht eingesetzt werden.

- Für den Beginn und das Ende eines Bereichs können das **XMLData-**Element und eine Textposition innerhalb dieses Elements gesetzt und abgerufen werden. Nähere Informationen dazu finden Sie unter FirstXMLData 1781, FirstXMLDataOffset 1782, LastXMLData 1794 und LastXMLDataOffset 1794. Diese Methode ist äußerst effizient, erfordert aber Kenntnisse über die zugrunde liegende Dokumentstruktur. Sie kann zum Suchen von XMLData-Objekten und zum Durchführen von Operationen, die über die Benutzeroberfläche nicht aufgerufen werden können, verwendet werden.

Änderungen am Dokumentinhalt können durch verschiedene Methoden vorgenommen werden:

© 2016-2022 Altova GmbH

- Mit Hilfe der Text[1802]-Eigenschaft können Sie den durch das Bereichsobjekt ausgewählten Dokumenttext abrufen. Wenn die Eigenschaft gesetzt ist, wird der ausgewählte Dokumenttext durch den neuen Text ersetzt.
- Die Standardbearbeitungsfunktionen Cut[1778], Copy[1777], Paste[1797]und Delete[1778].
- Tabellenoperationen für Tabellen, die dynamisch wachsen können.
- Methoden, die den Funktionalitäten der Authentic-Eingabehilfefenster zugeordnet sind.
- Aufrufe der XMLData[1970]Objekte des zugrunde liegenden Dokuments, um diese direkt zu bearbeiten.

29.3.2.5.1 AppendRow

Methode: AppendRow()als Boolean

Beschreibung
Wenn sich der Anfang eines Bereichs innerhalb einer dynamischen Tabelle befindet, fügt diese Methode eine neue Zeile am Ende der ausgewählten Tabelle hinzu. Die Bereichsauswahl wird geändert, so dass sie auf den Beginn der neuen Zeile zeigt. Die Funktion gibt *true* zurück, wenn die Anhängeoperation erfolgreich war. Anderfalls wird *false* zurückgegeben.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ----------------------------------------------------
' Scripting environment - VBScript
' Append row at end of current dynamically growable table
' ----------------------------------------------------
DimobjRange
' we assume that the active document is open in authentic view mode
SetobjRange = Application.ActiveDocument.AuthenticView.Selection

' check if we can insert something
IfobjRange.IsInDynamicTable Then
        objRange.AppendRow
        ' objRange points to beginning of new row
        objRange.Select
EndIf
```

29.3.2.5.2 Application

Eigenschaft: Applicationals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
2001 The authentic range object or its related view object is no longer valid.Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende

Ansichtsobjekt ist nicht mehr gültig
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.3 CanPerformAction

Methode: CanPerformAction(*eAction*als SPYAuthenticActions[2020],*strElementName*als String) als Boolean

Beschreibung
CanPerformActionund die damit in Zusammenhang stehenden Methoden ermöglichen den Zugriff auf die Eingabehilfenfunktionen von Authentic. Diese Funktion gestattet die mühelose und einheitliche Bearbeitung des Dokumentinhalts, ohne dass Sie wissen müssen, wo genau die Änderung durchgeführt wird. Der Beginn des Bereichsobjekts dient dazu, die nächste gültige Stelle zu finden, an der die angegebene Aktion durchgeführt werden kann. Sobald diese Stelle gefunden wurde, gibt die Methode *True*zurück, andernfalls *False.*

Tipp: Um alle gültigen Elementnamen für eine bestimmte Aktion zu ermitteln, können Sie CanPerformActionWith[1776]verwenden.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
2007 Es wurde eine ungültige Aktion definiert.

Beispiele
Siehe PerformAction[1798].

29.3.2.5.4 CanPerformActionWith

Methode: CanPerformActionWith(*eAction* als SPYAuthenticActions[2020], *out_arrElementNames* als Variant)

Beschreibung
PerformActionWithund die damit in Zusammenhang stehenden Methoden ermöglichen den Zugriff auf die Eingabehilfenfunktionen von Authentic. Diese Funktion gestattet die mühelose und einheitliche Bearbeitung des Dokumentinhalts, ohne dass Sie wissen müssen, wo genau die Änderung durchgeführt wird.

Diese Methode gibt einen Array jener Elementnamen zurück, an denen die angegebene Aktion durchgeführt werden kann.

Tipp: Verwenden Sie zur Anwendung der Aktion CanPerformActionWith[1776].

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
2007 Es wurde eine ungültige Aktion definiert.

Beispiele
Siehe PerformAction[1798].

© 2016-2022 Altova GmbH

29.3.2.5.5 Clone

Methode: Clone() als AuthenticRange[1773]

Beschreibung
Gibt eine Kopie des Bereichsobjekts zurück.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.6 CollapsToBegin

Methode: CollapsToBegin() als AuthenticRange[1773]

Beschreibung
Setzt das Ende des Bereichsobjekts auf seinen Beginn. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.7 CollapsToEnd

Methode: CollapsToEnd() as AuthenticRange[1773]

Beschreibung
Setzt den Beginn des Bereichsobjekts auf sein Ende. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.8 Copy

Methode: Copy() als Boolean

Beschreibung
Gibt *False*zurück, wenn der Bereich keine Abschnitte des Dokuments enthält, die kopiert werden können. Gibt *True*zurück, wenn Text - und im Fall vollständig ausgewählter XML-Elemente auch die Elemente - in den Kopieren/Einfügen-Puffer kopiert wurde.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.9 Cut

Methode: Cut() als Boolean

Beschreibung
Gibt *False*zurück, wenn der Bereich Dokumentabschnitte enthält, die nicht gelöscht werden dürfen.
Gibt *True*zurück, nachdem Text - und im Fall von vollständig ausgewählten XML-Elementen auch die Elemente aus dem Dokument gelöscht und im Kopieren/Einfügen-Puffer gespeichert wurden.-

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.10 Delete

Methode: Delete() als Boolean

Beschreibung
Gibt *False*zurück, wenn der Bereich Dokumentabschnitte enthält, die nicht gelöscht werden dürfen.
Gibt *True*zurück, nachdem Text und im Fall von vollständig ausgewählten XML-Elementen auch die Elemente aus dem Dokument gelöscht wurden.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.11 DeleteRow

Methode: DeleteRow() als Boolean

Beschreibung
Wenn sich der Anfang des Bereichs innerhalb einer dynamischen Tabelle befindet, löscht diese Methode die ausgewählte Zeile. Die Bereichsauswahl wird geändert, so dass sie auf das nächste Element nach der gelöschten Zeile zeigt. Die Funktion gibt *true* zurück, wenn die Operation erfolgreich war, andernfalls ist der Rückgabewert *false*.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

Beispiele

```
' ----------------------------------------------------------
' Scripting environment - VBScript
' Delete selected row from dynamically growing table
' ----------------------------------------------------------
DimobjRange
' we assume that the active document is open in authentic view mode
SetobjRange = Application.ActiveDocument.AuthenticView.Selection

' check if we are in a table
IfobjRange.IsInDynamicTable Then
        objRange.DeleteRow
EndIf
```

29.3.2.5.12 DuplicateRow

Methode: DuplicateRow() als Boolean

Beschreibung

Wenn sich der Anfang des Bereichs innerhalb einer dynamischen Tabelle befindet, fügt diese Methode eine Kopie der aktuellen Zeile nach der ausgewählten ein. Die Bereichsauswahl wird geändert, so dass sie auf den Anfang der nächsten Zeile zeigt. Die Funktion gibt *true* zurück, wenn die Operation erfolgreich war, andernfalls ist der Rückgabewert *false*.

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ----------------------------------------------------------
' Scripting environment - VBScript
' duplicate row in current dynamically growable table
' ----------------------------------------------------------
DimobjRange
' we assume that the active document is open in authentic view mode
SetobjRange = Application.ActiveDocument.AuthenticView.Selection

' check if we can insert something
IfobjRange.IsInDynamicTable Then
        objRange.DuplicateRow
        ' objRange points to beginning of new row
        objRange.Select
EndIf
```

29.3.2.5.13 EvaluateXPath

Methode: EvaluateXPath (strExpression als string) strValue als string

Rückgabewert
Die Methode gibt einen String zurück

Beschreibung
`EvaluateXPath()` führt einen XPath-Ausdruck aus, wobei der Kontext-Node der Anfang der Bereichsauswahl ist. Das als String zurückgegebene Ergebnis ist im Fall einer Sequenz ein durch Leerzeichen getrennter String. Wenn der XML-Node nicht relevant ist, kann der Benutzer jeden Node angeben, wie z.B. `AuthenticView.XMLDataRoot`.

Fehler

2001	Ungültiges Objekt
2005	Ungültiger Parameter
2008	Interner Fehler
2202	Fehlender Kontext-Node
2211	XPath-Fehler

29.3.2.5.14 ExpandTo

Methode: ExpandTo (*eKind* als SPYAuthenticElementKind[2021]), als AuthenticRange[1773]

Beschreibung
Wählt das gesamte Element vom Typ eKindaus, das an der ersten Cursorposition des Bereichs beginnt oder diese enthält. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2003	Die Erweiterung des Bereichs würde über das Ende des Dokuments hinausgehen.
2005	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.15 FirstTextPosition

Eigenschaft: FirstTextPosition als Long

Beschreibung
Setzt oder ruft den am weitesten links gelegenen Positionsindex des Bereichsobjekts ab. Dieser Index ist immer kleiner oder gleich der LastTextPosition[1793]. Die Indizierung beginnt am Dokumentanfang mit 0 und steigt dann mit jeder neuen Position, die der Textcursor einnehmen kann, schrittweise an. Die schrittweise Erhöhung der Textposition um 1 hat denselben Effekt, wie die nach rechts-Taste. Die schrittweise Verringerung der Textposition um 1 hat denselben Effekt, wie die nach lins-Taste.

Wenn Sie FirstTextPosition auf einen höheren Wert als den der aktuellen LastTextPosition[1793] setzen, wird LastTextPosition[1793] auf den neuen Wert von FirstTextPosition gesetzt.

Tipp: Seien Sie beim Setzen der Textcursorposition vorsichtig, da dies im Vergleich zur Cursorpositionierung mittes XMLData eine kostspielige Operation ist.

© 2016-2022 Altova GmbH

Fehler

2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
2006 Es wurde eine Textposition außerhalb des Dokuments definiert.

Beispiele

```
' ----------------------------------------
' Scripting environment - VBScript
' ----------------------------------------
DimobjAuthenticView
' we assume that the active document is open in authentic view mode
SetobjAuthenticView = Application.ActiveDocument.AuthenticView

nDocStartPosition = objAuthenticView.DocumentBegin.FirstTextPosition
nDocEndPosition = objAuthenticView.DocumentEnd.FirstTextPosition

' let's create a range that selects the whole document
' in an inefficient way
DimobjRange
' we need to get a (any) range object first
SetobjRange = objAuthenticView.DocumentBegin
objRange.FirstTextPosition = nDocStartPosition
objRange.LastTextPosition = nDocEndPosition

' let's check if we got it right
IfobjRange.isEqual(objAuthenticView.WholeDocument) Then
        MsgBox "Test using direct text cursor positioning was ok"
Else
        MsgBox "Ooops!"
EndIf
```

29.3.2.5.16 FirstXMLData

Eigenschaft: FirstXMLData als XMLData[1970]

Beschreibung

Setzt oder ruft das erste XMLData Element im zugrunde liegenden Dokument ab, das durch die Bereichsauswahl teilweise oder zur Gänze ausgewählt ist. Der genaue Anfang der Auswahl wird durch das FirstXMLDataOffset[1782] Attribut definiert.

Immer wenn Sie FirstXMLData auf ein neues Datenobjekt setzen, wird FirstXMLDataOffset[1782] auf die erste Cursorposition innerhalb dieses Elements gesetzt. Nur XMLData-Objekte, die eine Cursorposition haben, können verwendet werden. Wenn Sie FirstXMLData / FirstXMLDataOffset[1782] auf eine Position setzen, die größer ist als der aktuelle Wert von LastXMLData[1794] / LastXMLDataOffset[1794] wird zweiteres wird auf die neue Anfangsposition verschoben.

Tipp: Sie können das zugrunde liegende XML-Dokument in Fällen, in denen die mit dem AuthenticRange[1773]-Objekt verfügbaren Methoden nicht ausreichen, mit Hilfe der Eigenschaften FirstXMLData[1781] und LastXMLData[1794] direkt aufrufen und bearbeiten

Fehler

2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
2008 Interner Fehler
2009 Das XMLData-Objekt kann nicht aufgerufen werden.

Beispiele

```
' ---------------------------------------------
' Scripting environment - VBScript
' show name of currently selected XMLData element
' ---------------------------------------------
DimobjAuthenticView
' we assume that the active document is open in authentic view mode
SetobjAuthenticView = Application.ActiveDocument.AuthenticView

DimobjXmlData
SetobjXMLData = objAuthenticView.Selection.FirstXMLData
' authentic view adds a 'text' child element to elements
' of the document which have content. So we have to go one
' element up.
SetobjXMLData = objXMLData.Parent
MsgBox "Current selection selects element " & objXMLData.Name
```

29.3.2.5.17 FirstXMLDataOffset

Eigenschaft: FirstXMLDataOffset as Long

Beschreibung

Setzt oder ruft den Versatz der Cursorposition innerhalb des Elements FirstXMLData[1781] für den Anfang des Bereichs ab. Die Versatzpositionen basieren auf den von der Eigenschaft Text[1802] zurückgegebenen Zeichen und beginnen mit 0. Verwenden Sie beim Definieren eines neuen Versatzes -1, um den Versatz an die letzte mögliche Position im Element zu setzen. Beachten Sie bitte besondes die folgenden Fälle:

- Die Textform von Einträgen in Auswahllisten, Kontrollkästchen und ähnlichen Steuerelementen kann anders aussehen, als angezeigt. Zwar basiert der Datenversatz auf diesem Text, doch gibt es nur zwei gültige Positionen für den Versatz, eine am Anfang und eine am Ende des Eintrags. Wenn Sie versuchen, den Versatz irgendwo mitten im Eintrag zu positionieren, hat dies zur Folge, dass der Versatz an das Ende des Eintrags gesetzt wird.

- Die Textform von XML-Entities kann sich in ihrer Länge von der Darstellung auf dem Bildschirm unterscheiden. Der Versatz basiert auf der Textform.

Wenn FirstXMLData / FirstXMLDataOffset[1782] eine Position nach dem aktuellen LastXMLData[1794] / LastXMLDataOffset[1794] wählt, wird zweiteres an die neue Anfangsposition verschoben.

Fehler

2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht gültig.
2005 Es wurde ein ungültiger Versatz definiert.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

Beispiele

```
' ---------------------------------------------
' Scripting environment - VBScript
' Select the complete text of an XMLData element
' using XMLData based selection and ExpandTo
' ---------------------------------------------
DimobjAuthenticView
' we assume that the active document is open in authentic view mode
SetobjAuthenticView = Application.ActiveDocument.AuthenticView

' first we use the XMLData based range properties
' to select all text of the first XMLData element
' in the current selection
DimobjRange
SetobjRange = objAuthenticView.Selection
objRange.FirstXMLDataOffset = 0  ' start at beginning of element text
objRange.LastXMLData = objRange.FirstXMLData  ' select only one element
objRange.LastXMLDataOffset = -1  ' select till its end

' the same can be achieved with the ExpandTo method
DimobjRange2
SetobjRange2 = objAuthenticView.Selection.ExpandTo(spyAuthenticTag)

' were we successful?
IfobjRange.IsEqual(objRange2) Then
        objRange.Select()
Else
        MsgBox "Oops"
EndIf
```

29.3.2.5.18 GetElementAttributeNames

Methode: GetElementAttributeNames (*strElementName* als String, *out_arrAttributeNames* als Variant)

Beschreibung

Ruft die Namen aller Attribute für das einschließende Element mit dem angegebenen Namen auf. Verwenden Sie die Element/Attribut-Paare, um den Attributwert mit den Methoden GetElementAttributeValue[1784] und SetElementAttributeValue[1801] zu setzen oder abzurufen.

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005	Invalid element name was specified. Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

Siehe SetElementAttributeValue[1801].

29.3.2.5.19 GetElementAttributeValue

Methode: GetElementAttributeValue (*strElementName* als String, *strAttributeName* als String) als String

Beschreibung
Ruft für das in strElementName angegebene Element den Wert des in strAttributeName definierten Attributs ab. Wenn das Attribut unterstütz wird, aber keinen Wert zugewiesen hat, wird der leere String zurückgegeben. Um die Namen der von einem Element unterstützten Attribute herauszufinden, verwenden Sie GetElementAttributeNames[1783], oder HasElementAttribute[1788].

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005	Es wurde ein ungültiger Elementname angegeben. Es wurde ein ungültiger Attributname angegeben. Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele
Siehe SetElementAttributeValue[1801].

29.3.2.5.20 GetElementHierarchy

Methode: GetElementHierarchy (*out_arrElementNames* als Variant)

Beschreibung
Ruft die Namen aller XML-Elemente ab, die der aktuellen Auswahl übergeordnet sind. Innere Elemente werden vor einschließenden Elementen aufgelistet. Wenn sich die aktuelle Auswahl nicht innerhalb eines einzelnen XMLData-Elements befindet, wird eine leere Liste zurückgegeben.

Die Namen der Elementhierarchie identifizieren zusammen mit dem Bereichsobjekt XMLData Elemente im Dokument. Die Attribute dieser Elemente können direkt von GetElementAttributeNames[1783] und damit in Zusammenhang stehenden Methoden aufgerufen werden.

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

C#-Beispiele

```
' ----------------------------------------
' C#
' ----------------------------------------

namespace ConsoleApplication1
{
    class Program
    {
```


© 2016-2022 Altova GmbH

```
        static void Main(string[] args)
        {
            XMLSpyLib.Application app = new XMLSpyLib.Application();

            app.ShowApplication(true);

            XMLSpyLib.AuthenticView view = app.ActiveDocument.AuthenticView;
            XMLSpyLib.AuthenticRange range = view.DocumentBegin;

            object o = null;
            range.GetElementHierarchy(ref o);

            object[] elements = (object[])o;

            foreach (string e in elements)
            {
                Console.WriteLine(e);
            }
        }
    }
}
```

Also see: SetElementAttributeValue[1801].

29.3.2.5.21 GetEntityNames

Methode: GetEntityNames (*out_arrEntityNames* als Variant)

Beschreibung
Ruft die Namen alle definierten Entities ab. Die Liste der abgerufenen Entities hängt nicht von der aktuellen Auswahl oder der aktuellen Position ab. Verwenden Sie einen dieser Namen mit der Funktion InsertEntity[1789].

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele
Siehe GetElementHierarchy[1784] und InsertEntity[1789].

29.3.2.5.22 GetVariableValue

Methode: GetVariableValue `(strName als string) strVal als string`

Rückgabewert
Ruft den Wert der als Parameter der Methode benannten Variablen ab.

Fehler

2001	Ungültiges Objekt.
2202	Kein Kontext.

2204 Es befindet sich keine Variable dieses Namens im Geltungsbereich
2205 Die Variable kann nicht ausgewertet werden
2206 Die Variable gibt eine Sequenz zurück
2209 Ungültiger Parameter

29.3.2.5.23 Goto

Methode: Goto (*eKind* als SPYAuthenticElementKind[2021], *nCount* als Long, *eFrom* als SPYAuthenticDocumentPosition[2021]) als AuthenticRange[1773]

Beschreibung
Definiert den Bereich, der auf den Anfang des nCount Elements vom Typ eKind zeigt. Die Anfangsposition wird durch den Parameter eFrom definiert.

Mit Hilfe positiver Werte für nCount können Sie zum Ende des Dokuments navigieren. Mit Hilfe negativer Werte können Sie zum Beginn des Dokuments navigieren. die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2003 Das Ziel liegt hinter dem Ende des Dokuments.
2004 Das Ziel liegt vor dem Anfang des Dokuments.
2005 Es wurde eine ungültige Elementart definiert.
Es wurde eine ungültige Anfangsposition definiert.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.24 GotoNext

Methode: GotoNext (*eKind* als SPYAuthenticElementKind[2021]) als AuthenticRange[1773]

Beschreibung
Setzt den Bereich auf den Anfang des nächsten Elements vom Typ eKind. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2003 Das Ziel liegt hinter dem Ende des Dokuments.
2005 Es wurde eine ungültige Elementart definiert.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ----------------------------------------
' Scripting environment - VBScript
' Scan through the whole document word-by-word
' ----------------------------------------
DimobjAuthenticView
' we assume that the active document is open in authentic view mode
SetobjAuthenticView = Application.ActiveDocument.AuthenticView
```


© 2016-2022 Altova GmbH

```
DimobjRange
SetobjRange = objAuthenticView.DocumentBegin
DimbEndOfDocument
bEndOfDocument = False

OnErrorResume Next
WhileNotbEndOfDocument
        objRange.GotoNext(spyAuthenticWord).Select
        If((Err.number - vbObjecterror) = 2003) Then
                bEndOfDocument = True
                Err.Clear
        ElseIf(Err.number <> 0) Then
                Err.Raise  ' forward error
        EndIf
Wend
```

29.3.2.5.25 GotoNextCursorPosition

Methode: GotoNextCursorPosition() als AuthenticRange[1773]

Beschreibung
Setzt den Bereich auf die nächste Cursorposition hinter der aktuellen Endposition. Gibt das geänderte Objekt zurück.

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2003	Das Ziel liegt hinter dem Ende des Dokuments.
2005	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.26 GotoPrevious

Methode: GotoPrevious (*eKind* als SPYAuthenticElementKind[2021]) als AuthenticRange[1773]

Beschreibung
Setzt den Bereich auf den Beginn des Elements vom Typ eKind, welches sich vor dem Beginn des aktuellen Bereichs berfindet. Die Methode geibt das geänderte Bereichsobjekt zurück.

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2004	Das Ziel liegt vor dem Anfang des Dokuments.
2005	Es wurde eine ungültige Elementart definiert. Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ----------------------------------------
' Scripting environment - VBScript
' Scan through the whole document tag-by-tag
```

```
' ----------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objRange
Set objRange = objAuthenticView.DocumentEnd
Dim bBeginOfDocument
bBeginOfDocument = False

On Error Resume Next
While Not bBeginOfDocument
        objRange.GotoPrevious(spyAuthenticTag).Select
        If ((Err.number - vbObjecterror) = 2004) Then
                bBeginOfDocument = True
                Err.Clear
        ElseIf (Err.number <> 0) Then
                Err.Raise  ' forward error
        End If
Wend
```

29.3.2.5.27 GotoPreviousCursorPosition

Methode: GotoPreviousCursorPosition() als AuthenticRange[1773]

Beschreibung
Setzt den Bereich auf die Cursorposition unmittelbar vor der aktuellen Position. Gibt das geänderte Objekt zurück.

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2004	Das Ziel liegt vor dem Anfang des Dokuments.
2005	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.28 HasElementAttribute

Methode: HasElementAttribute (*strElementName* als String, *strAttributeName* als String) als Boolean

Beschreibung
Überprüft, ob das einschließende Element mit dem Namen strElementName das in strAttributeName definierte Attribut unterstützt.

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005	Es wurde ein ungültiger Elementname angegeben. Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.5.29 InsertEntity

Methode: InsertEntity (*strEntityName* als String)

Beschreibung
Ersetzt die Bereichsauswahl durch die angegebene Entity. Die angegebene Entity muss einer der von GetEntityNames[1785] zurückgegebenen Entity-Namen sein.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Es wurde ein unbekannter Name für den Eintrag angegeben.

Beispiele

```
' ----------------------------------------------------------
' Scripting environment - VBScript
' Insert the first entity in the list of available entities
' ----------------------------------------------------------
DimobjRange
' we assume that the active document is open in authentic view mode
SetobjRange = Application.ActiveDocument.AuthenticView.Selection

' first we get the names of all available entities as they
' are shown in the entry helper of XMLSpy
DimarrEntities
objRange.GetEntityNames arrEntities

' we insert the first one of the list
IfUBound(arrEntities) >= 0 Then
        objRange.InsertEntity arrEntities(0)
Else
        MsgBox "Sorry, no entities are available for this document"
EndIf
```

29.3.2.5.30 InsertRow

Methode: InsertRow() als Boolean

Beschreibung
Wenn sich der Beginn des Bereichs innerhalb einer dynamischen Tabelle befindet, fügt diese Methode eine neue Zeile vor der aktuellen ein. Der Auswahlbereich wird geändert, sodass er auf den Beginn der neu eingefügten Zeile zeigt. Die Funktion gibt *true* zurück, wenn die Einfügeoperation erfolgreich war, anderfalls ist das Ergebnis *false*.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ----------------------------------------------------------
' Scripting environment - VBScript
' Insert row at beginning of current dynamically growing table
' ----------------------------------------------------------
DimobjRange
' we assume that the active document is open in authentic view mode
SetobjRange = Application.ActiveDocument.AuthenticView.Selection

' check if we can insert something
IfobjRange.IsInDynamicTable Then
        objRange.InsertRow
        ' objRange points to beginning of new row
        objRange.Select
 EndIf
```

29.3.2.5.31 IsCopyEnabled

Eigenschaft: IsCopyEnabled als Boolean (schreibgeschützt)

Beschreibung
Überprüft, ob die Kopieroperation für diesen Bereich unterstützt wird.

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.32 IsCutEnabled

Eigenschaft: IsCutEnabled als Boolean (schreibgeschützt)

Beschreibung
Überprüft, ob die Ausschneideoperation für diesen Bereich unterstützt wird.

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.33 IsDeleteEnabled

Eigenschaft: IsDeleteEnabled als Boolean (schreibgeschützt)

Beschreibung
Überprüft, ob die Löschoperation für diesen Bereich unterstützt wird.

Fehler

© 2016-2022 Altova GmbH

2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.34 IsEmpty

Methode: IsEmpty() als Boolean

Beschreibung
Überprüft, ob die erste und die letzte Position des Bereichs identisch sind.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.35 IsEqual

Methode: IsEqual (*objCmpRange* als AuthenticRange[1773]) als Boolean

Beschreibung
Überprüft, ob der Beginn und das Ende beider Bereiche identisch sind.

Fehler
2001 Eines der beiden verglichenen Bereichsobjekte ist ungültig.
2005 Es wurde eine ungültige Adresse für einen Rückgabeparameter angegeben.

29.3.2.5.36 IsFirstRow

Eigenschaft: IsFirstRow als Boolean (schreibgeschützt)

Beschreibung
Überprüft, ob der Bereich in der ersten Zeile einer Tabelle ist. Welche Tabelle berücksichtigt wird, ist vom Ausmaß des Bereichs abhängig. Wenn die Auswahl über eine einzige Tabellenzeile hinausgeht, wird überprüft, ob diese Tabelle das erste Element in einer einbettenden Tabelle ist. Nähere Informationen dazu finden Sie im Benutzerhandbuch unter Eingabehilfen.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.37 IsInDynamicTable

Methode: IsInDynamicTable() als Boolean

Beschreibung
Überprüft, ob sich der gesamte Bereich innerhalb einer Tabelle befindet, die die verschiedenen Zeilenoperationen wie Einfügen, Anhängen, Kopieren usw. unterstützt.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.38 IsLastRow

Eigenschaft: IsLastRow als Boolean (schreibgeschützt)

Beschreibung
Überprüft, ob der Bereich in der letzten Zeile einer Tabelle ist. Welche Tabelle berücksichtigt wird, hängt vom Ausmaß des Bereichs ab. Wenn die Auswahl über eine einzige Tabellenzeile hinausgeht, wird überprüft, ob diese Tabelle das letzte Element in einer einbettenden Tabelle ist. Nähere Informationen dazu finden Sie im Benutzerhandbuch unter Eingabehilfen.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.39 IsPasteEnabled

Eigenschaft: IsPasteEnabled als Boolean (schreibgeschützt)

Beschreibung
Überprüft, ob die Einfügeoperatione für diesen Bereich unterstützt wird.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.40 IsSelected

Eigenschaft: IsSelected als Boolean

Beschreibung
Gibt true() zurück, wenn die Auswahl vorhanden ist. Der Auswahlbereich kann dennoch leer sein. Dies passiert z.B. wenn der Cursor gesetzt ist.

© 2016-2022 Altova GmbH

29.3.2.5.41 IsTextStateApplied

Methode: IsTextStateApplied (*i_strElementName* als String) als Boolean

Beschreibung
Überprüft, ob der gesamte ausgewählte Text in ein XML-Element mit dem Namen i_strElementName eingebettet ist. Ein typisches Beispiel für den Parameter i_strElementName ist "strong", "bold" oder "italic".

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.42 LastTextPosition

Eigenschaft: LastTextPosition als Long

Beschreibung
Setzt oder ruft den am weitesten rechts gelegenen Positionsindex des Bereichsobjekts ab. Dieser Index ist immer größer oder gleich FirstTextPosition[1780]. Die Indizierung beginnt am Dokumentanfang mit 0 und steigt dann mit jeder neuen Position, die der Textcursor einnehmen kann, schrittweise an. Die schrittweise Erhöhung der Textposition um 1 hat denselben Effekt, wie die nach rechts-Taste. Die schrittweise Verringerung der Textposition um 1 hat denselben Effekt, wie die nach lins-Taste.

Wenn Sie LastTextPosition auf einen niedrigeren Wert als den der aktuellen FirstTextPosition[1780] setzen, wird FirstTextPosition[1780] auf den neuen Wert von LastTextPosition gesetzt.

Tipp: Seien Sie beim Setzen der Textcursorposition vorsichtig, da dies im Vergleich zur Cursorpositionierung mittes XMLData eine kostspielige Operation ist.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
2006 Es wurde eine Textposition außerhalb des Dokuments definiert.

Beispiele

```
' ------------------------------------
' Scripting environment - VBScript
' ------------------------------------
DimobjAuthenticView
' we assume that the active document is open in authentic view mode
SetobjAuthenticView = Application.ActiveDocument.AuthenticView

nDocStartPosition = objAuthenticView.DocumentBegin.FirstTextPosition
nDocEndPosition = objAuthenticView.DocumentEnd.FirstTextPosition

' let's create a range that selects the whole document
' in an inefficient way
DimobjRange
```

```
' we need to get a (any) range object first
SetobjRange = objAuthenticView.DocumentBegin
objRange.FirstTextPosition = nDocStartPosition
objRange.LastTextPosition = nDocEndPosition

' let's check if we got it right
IfobjRange.isEqual(objAuthenticView.WholeDocument) Then
        MsgBox "Test using direct text cursor positioning was ok"
Else
        MsgBox "Oops!"
EndIf
```

29.3.2.5.43 LastXMLData

Eigenschaft: LastXMLData als XMLData[1970]

Beschreibung
Setzt oder ruft das letzte XMLData Element im zugrunde liegenden Dokument ab, das durch die Bereichsauswahl teilweise oder zur Gänze ausgewählt ist. Das genaue Ende der Auswahl wird durch das LastXMLDataOffset[1794] Attribut definiert.

Immer wenn Sie LastXMLData auf ein neues Datenobjekt setzen, wird LastXMLDataOffset[1794] auf die letzte Cursorposition innerhalb dieses Elements gesetzt. Nur XMLData-Objekte, die eine Cursorposition haben, können verwendet werden. Wenn Sie LastXMLData / LastXMLDataOffset[1794] auf eine Position setzen, die niedriger ist als der aktuelle Wert von FirstXMLData[1781] / FirstXMLDataOffset[1782] wird zweiteres wird auf die neue Endposition verschoben.

Tipp: Sie können das zugrunde liegende XML-Dokument in Fällen, in denen die mit dem AuthenticRange[1773]-Objekt verfügbaren Methoden nicht ausreichen, mit Hilfe der Eigenschaften FirstXMLData[1781] und LastXMLData[1794] direkt aufrufen und bearbeiten

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht gültig.
2005	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
2008	Interner Fehler
2009	Das XMLDataObjekt kann nicht aufgerufen werden.

29.3.2.5.44 LastXMLDataOffset

Eigenschaft: LastXMLDataOffset als Long

Beschreibung
Setzt oder ruft den Versatz der Cursorposition innerhalb des Elements LastXMLData[1794] für das Ende des Bereichs ab.

© 2016-2022 Altova GmbH

Die Versatzpositionen basieren auf den von der Eigenschaft Text[1802] zurückgegebenen Zeichen und beginnen mit 0. Verwenden Sie beim Definieren eines neuen Versatzes -1, um den Versatz an die letzte mögliche Position im Element zu setzen. Beachten Sie bitte besondes die folgenden Fälle:

- Die Textform von Einträgen in Auswahllisten, Kontrollkästchen und ähnlichen Steuerelementen kann anders aussehen, als angezeigt. Zwar basiert der Datenversatz auf diesem Text, doch gibt es nur zwei gültige Positionen für den Versatz, eine am Anfang und eine am Ende des Eintrags. Wenn Sie versuchen, den Versatz irgendwo mitten im Eintrag zu positionieren, hat dies zur Folge, dass der Versatz an das Ende des Eintrags gesetzt wird.

- Die Textform von XML-Entities kann sich in ihrer Länge von der Darstellung auf dem Bildschirm unterscheiden. Der Versatz basiert auf der Textform.

Wenn LastXMLData[1794] / LastXMLDataOffset[1794] eine Position vor dem aktuellen FirstXMLData[1781] / FirstXMLDataOffset[1782] wählt, wird zweiteres an die neue Endposition verschoben.

Fehler

2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht gültig.
2005 Es wurde ein ungültiger Versatz definiert.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ---------------------------------------------
' Scripting environment - VBScript
' Select the complete text of an XMLData element
' using XMLData based selection and ExpandTo
' ---------------------------------------------
DimobjAuthenticView
' we assume that the active document is open in authentic view mode
SetobjAuthenticView = Application.ActiveDocument.AuthenticView

' first we use the XMLData based range properties
' to select all text of the first XMLData element
' in the current selection
DimobjRange
SetobjRange = objAuthenticView.Selection
objRange.FirstXMLDataOffset = 0  ' start at beginning of element text
objRange.LastXMLData = objRange.FirstXMLData  ' select only one element
objRange.LastXMLDataOffset = -1  ' select till its end

' the same can be achieved with the ExpandTo method
DimobjRange2
SetobjRange2 = objAuthenticView.Selection.ExpandTo(spyAuthenticTag)

' were we successful?
IfobjRange.IsEqual(objRange2) Then
        objRange.Select()
Else
        MsgBox "Ooops"
EndIf
```

29.3.2.5.45 MoveBegin

Methode: MoveBegin (*eKind* als SPYAuthenticElementKind[2021], *nCount* als Long) als AuthenticRange[1773]

Beschreibung
Verschiebt den Anfang des Bereichs an den Beginn des nCount Elements vom Typ eKind. Die Zählung begint am aktuellen Anfang des Bereichsobjekts.

Verwenden Sie positive Zahlen für nCount, um die Position zum Ende des Dokuments hin zu verschieben und negative Zahlen um die Position zum Anfang des Dokuments hin zu verschieben. Das Ende des Bereichs bleibt gleich, es sei denn, der neue Anfang hat einen höheren Wert als das Ende. In diesem Fall wird das Ende an den neuen Anfang verschoben. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2003	Das Ziel liegt hinter dem Ende des Dokuments.
2004	Das Ziel liegt vor dem Anfang des Dokuments.
2005	Es wurde eine ungültige Elementart definiert. Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.46 MoveEnd

Methode: MoveEnd (*eKind* als SPYAuthenticElementKind[2021], *nCount* als Long) als AuthenticRange[1773]

Beschreibung
Verschiebt das Ende des Bereichs an den Beginn des nCount Elements vom Typ eKind. Die Zählung begint am aktuellen Ende des Bereichsobjekts.

Verwenden Sie positive Zahlen für nCount, um die Position zum Ende des Dokuments hin zu verschieben und negative Zahlen um die Position zum Anfang des Dokuments hin zu verschieben. Der Anfang des Bereichs bleibt gleich, es sei denn, das neue Ende hat einen niedrigeren Wert als der Anfang. In diesem Fall wird der Anfang an das neue Ende verschoben. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2003	Das Ziel liegt hinter dem Ende des Dokuments.
2004	Das Ziel liegt vor dem Anfang des Dokuments.
2005	Es wurde eine ungültige Elementart definiert. Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.47 MoveRowDown

Methode: MoveRowDown() als Boolean

Beschreibung

© 2016-2022 Altova GmbH

Wenn sich der Anfang des Bereichs innerhalb einer dynamischen Tabelle befindet und eine Zeile auswählt, die nicht die letzte Zeile in dieser Tabelle ist, tauscht diese Methode diese Zeile gegen die unmittelbar darunter gelegene Zeile aus. Die Auswahl des Bereichs verschiebt sich mit der Zeile, ändert sich aber ansonsten nicht. Die Funktion gibt *true* zurück, wenn der Verschiebevorgang erfolgreich war, andernfalls wird *false* zurückgegeben.

Fehler

2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.48 MoveRowUp

Method: MoveRowUp() as Boolean

Beschreibung

Wenn sich der Anfang des Bereichs innerhalb einer dynamischen Tabelle befindet und eine Zeile auswählt, die nicht die erste Zeile in dieser Tabelle ist, tauscht diese Methode diese Zeile gegen die unmittelbar darüber gelegene Zeile aus. Die Auswahl des Bereichs verschiebt sich mit der Zeile, ändert sich aber ansonsten nicht. Die Funktion gibt *true* zurück, wenn der Verschiebevorgang erfolgreich war, andernfalls wird *false* zurückgegeben

Fehler

2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.49 Parent

Eigenschaft: Parent als AuthenticView[1803] (schreibgeschützt)

Beschreibung

Ruft die Ansicht auf, die dieses Bereichsobjekt enthält.

Fehler

2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.50 Paste

Methode: Paste() als Boolean

Beschreibung

Gibt *False*zurück, wenn der Kopieren/Einfügen-Puffer leer ist oder sein Inhalt die aktuelle Auswahl nicht ersetzen kann.

Löscht andernfalls die aktuelle Auswahl, fügt den Inhalt des Kopieren/Einfügen-Puffers ein und gibt *True*zurück.

Fehler

2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.5.51 PerformAction

Methode: PerformAction (*eAction* als SPYAuthenticActions[2020], *strElementName* als String) als Boolean

Beschreibung

PerformActionund die dazugehörigen Methoden verschaffen Ihnen Zugriff auf die Eingabehilfenfunktionen von Authentic. Diese Funktion gestattet die einfach und einheitliche Bearbeitung des Dokumentinhalts, ohne dass Sie wissen müssen, wo genau die Änderung stattfinden soll. Der Anfang des Bereichsobjekts dient dazu, die nächste gültige Stelle zu finden, an der die angegebene Aktion durchgeführt werden kann. Wenn keine derartige Stelle gefunden wird, gibt die Methode *False*. zurück. Andernfalls wird das Dokument geändert und der Bereich zeigt auf den Anfang der Änderung.

Tipp: Um Elementnamen zu finden, die als der zweite Parameter übergeben werden können, verwenden Sie CanPerformActionWith[1776].

Fehler

2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
2007 Es wurde eine ungültige Aktion definiert.

Beispiele

```
' ----------------------------------------
' Scripting environment - VBScript
' Insert the innermost element
' ----------------------------------------
DimobjRange
' we assume that the active document is open in authentic view mode
SetobjRange = Application.ActiveDocument.AuthenticView.Selection

' we determine the elements that can be inserted at the current position
DimarrElements()
objRange.CanPerformActionWith spyAuthenticInsertBefore, arrElements

' we insert the first (innermost) element
IfUBound(arrElements) >= 0 Then
        objRange.PerformAction spyAuthenticInsertBefore, arrElements(0)
        ' objRange now points to the beginning of the inserted element
        ' we set a default value and position at its end
        objRange.Text = "Hello"
        objRange.ExpandTo(spyAuthenticTag).CollapsToEnd().Select
Else
        MsgBox "Can't insert any elements at current position"
EndIf
```


© 2016-2022 Altova GmbH

29.3.2.5.52 Select

Methode: Select()

Beschreibung
Macht diesen Bereich zur aktuellen Auswahl der Benutzeroberfläche. Dasselbe Ergebnis können Sie mit: '*objRange.Parent.Selection = objRange*' erzielen.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.

Beispiele
```
' ----------------------------------------
' Scripting environment - VBScript
' ----------------------------------------
DimobjAuthenticView
' we assume that the active document is open in authentic view mode
SetobjAuthenticView = Application.ActiveDocument.AuthenticView

' set current selection to end of document
objAuthenticView.DocumentEnd.Select()
```

29.3.2.5.53 SelectNext

Methode: SelectNext (*eKind* als SPYAuthenticElementKind[2021]) als AuthenticRange[1773]

Beschreibung
Wählt das Element vom Typ eKind nach dem aktuellen Ende des Bereichs aus. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler
2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2003 Das Ziel liegt hinter dem Ende des Dokuments.
2005 Es wurde eine ungültige Elementart definiert.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele
```
' ----------------------------------------
' Scripting environment - VBScript
' Scan through the whole document word-by-word
' ----------------------------------------
DimobjAuthenticView
' we assume that the active document is open in authentic view mode
SetobjAuthenticView = Application.ActiveDocument.AuthenticView

DimobjRange
SetobjRange = objAuthenticView.DocumentBegin
DimbEndOfDocument
```

```
bEndOfDocument = False

OnErrorResume Next
WhileNotbEndOfDocument
        objRange.SelectNext(spyAuthenticWord).Select
        If((Err.number - vbObjecterror) = 2003) Then
                bEndOfDocument = True
                Err.Clear
        ElseIf(Err.number <> 0) Then
                Err.Raise  ' forward error
        EndIf
Wend
```

29.3.2.5.54 SelectPrevious

Methode: GotoPrevious (*eKind* als SPYAuthenticElementKind[2021]) als AuthenticRange[1773]

Beschreibung
Wählt das Element vom Typ eKind vor dem aktuellen Anfang des Bereichs aus. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2004	Das Ziel liegt vor dem Anfang des Dokuments
2005	Es wurde eine ungültige Elementart definiert. Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ----------------------------------------
' Scripting environment - VBScript
' Scan through the whole document tag-by-tag
' ----------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objRange
Set objRange = objAuthenticView.DocumentEnd
Dim bBeginOfDocument
bBeginOfDocument = False

On Error Resume Next
While Not bBeginOfDocument
        objRange.SelectPrevious(spyAuthenticTag).Select
        If ((Err.number - vbObjecterror) = 2004) Then
                bBeginOfDocument = True
                Err.Clear
        ElseIf (Err.number <> 0) Then
```


© 2016-2022 Altova GmbH

```
            Err.Raise  ' forward error
        End If
Wend
```

29.3.2.5.55 SetElementAttributeValue

Methode: SetElementAttributeValue (*strElementName* als String, *strAttributeName* als String, *strAttributeValue* als String)

Beschreibung
Definiert den Wert des in strAttributeName definierten Attributs für das mit strElementName identifizierte Element. Wenn das Attribut unterstützt wird, aber keinen Wert zugewiesen hat, wird der leere String zurückgegeben. Um die Namen der von einem Element unterstützten Attribute zu ermitteln, verwenden Sie GetElementAttributeNames[1783] oder HasElementAttribute[1788].

Fehler

2001	Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005	Es wurde ein ungültiger Elementname angegeben. Es wurde ein ungültiger Attributname definiert. Es wurde ein ungültiger Attributwert definiert.

Beispiele

```
' ----------------------------------------
' Scripting environment - VBScript
' Get and set element attributes
' ----------------------------------------
DimobjRange
' we assume that the active document is open in authentic view mode
SetobjRange = Application.ActiveDocument.AuthenticView.Selection

' first we find out all the elements below the beginning of the range
DimarrElements
objRange.GetElementHierarchy arrElements

IfIsArray(arrElements) Then
        IfUBound(arrElements) >= 0 Then
                ' we use the top level element and find out its valid attributes
                DimarrAttrs()
                objRange.GetElementAttributeNames arrElements(0), arrAttrs

                IfUBound(arrAttrs) >= 0 Then
                        ' we retrieve the current value of the first valid attribute
                        DimstrAttrVal
                        strAttrVal = objRange.GetElementAttributeValue (arrElements(0), arrAttrs(0))
                        msgbox "current value of " & arrElements(0) & "//" & arrAttrs(0) & " is: " & strAttrVal

                        ' we change this value and read it again
                        strAttrVal = "Hello"
                        objRange.SetElementAttributeValue arrElements(0), arrAttrs(0), strAttrVal
```

```
                    strAttrVal = objRange.GetElementAttributeValue (arrElements(0), arrAttrs(0))
                    msgbox "new value of " & arrElements(0) & "//" & arrAttrs(0) & " is: " & strAttrVal
                EndIf
            EndIf
EndIf
```

29.3.2.5.56 SetFromRange

Methode: SetFromRange (*objSrcRange* als AuthenticRange[1773])

Beschreibung
Setzt den Bereich des Objekts auf dieselbe Anfangs- und Endposition wie objSrcRange.

Fehler
2001 Eines der beiden Bereichsobjekte ist ungültig.
2005 Es wurde als Quellobjekt kein Objekt definiert.

29.3.2.5.57 SetVariableValue

Methode: SetVariableValue`(strName als string, strValue als string)`

Rückgabewert
Definiert den Wert (zweiten Parameter) der im ersten Parameter genannten Variablen.

Fehler
2201 Ungültiges Objekt.
2202 Kein Kontext.
2204 Es befindet sich keine Variable dieses Namens im Geltungsbereich.
2205 Die Variable kann nicht ausgewertet werden.
2206 Die Variable gibt eine Sequenz zurück.
2207 Die Variable ist schreibgeschützt.
2208 Es ist keine Änderung zulässig.

29.3.2.5.58 Text

Eigenschaft: Text als String

Beschreibung
Setzt oder ruft den durch ein Bereichsobjekt ausgewählten Textinhalt ab.

Die Anzahl der abgerufenen Zeichen muss nicht unbedingt identisch sein, da es Textcursorpositionen zwischen dem Beginn und dem Ende des ausgewählten Bereichs gibt. Die meisten Dokumentelemente unterstützen eine Cursor-Endposition, die sich von der Cursor-Beginnposition des folgenden Elements unterscheidet. Dropdown-Listen haben nur eine Cursorposition, können jedoch Strings jeder Länge auswählen. Im Fall von Optionsfeldern und Kontrollkästchen enthält die Texteigenschaft den String des entsprechenden XML-Elements.

© 2016-2022 Altova GmbH

Wenn mit dem Bereich mehr als ein Element ausgewählt wird, ist der Text die Verkettung der Einzeltexte. XML Entities werden erweitert, sodass '&' als '&' erwartet wird.

Wenn der Text auf den leeren String gesetzt wird, werden keine XML-Elemente gelöscht. Verwenden Sie stattdessen Cut[1778], Delete[1778] oder PerformAction[1798].

Fehler

2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsobjekt ist nicht mehr gültig.
2005 Es wurde eine ungültige Adresse für einen Rückgabeparameter angegeben.

29.3.2.6 AuthenticView

Eigenschaften
Application[1813]
AsXMLString[1813]
DocumentBegin[1815]
DocumentEnd[1815]
Event[1816]
MarkupVisibility[1819]
Parent[1819]
Selection[1820]
XMLDataRoot[1822]
WholeDocument[1822]

Methoden
Goto[1817]
IsRedoEnabled[1818]
IsUndoEnabled[1818]
Print[1819]
Redo[1820]
Undo[1821]
UpdateXMLInstanceEntities[1821]

Events
OnBeforeCopy[1804]
OnBeforeCut[1805]
OnBeforeDelete[1805]
OnBeforeDrop[1805]
OnBeforePaste[1807]
OnDragOver[1807]
OnKeyBoardEvent[1809]
OnMouseEvent[1810]
OnSelectionChanged[1810]

Beschreibung
Über AuthenticView und seine untergeordneten Objekte AuthenticRange[1773]und AuthenticDataTransfersteht Ihnen eine Schnittstelle für die Authentic-Ansicht zur Verfügung, über die Sie Dokumentinhalt einfach und konsistent ändern können. Diese Schnittstellen treten nun anstelle der folgenden nun als **nicht mehr verwendet** markierten Schnittstellen:
OldAuthenticView(der alte Name war DocEditView)
AuthenticSelection(der alte Name war DocEditSelection, wird nun ersetzt durch AuthenticRange[1773])
AuthenticEvent(der alte Name war DocEditEvent)

Mit AuthenticView erhalten Sie einfachen Zugriff auf spezielle Funktionen wie Drucken, den mehrstufigen Rückgängig-Puffer und die aktuelle Cursorauswahl oder - position.

AuthenticView verwendet zur einfachen Navigation im Dokument und zur flexiblen Auswahl logischer Textelemente Objekte vom Typ AuthenticRange[1773]. Verwenden Sie für einfache Auswahldefinitionen die Eigenschaften DocumentBegin[1815], DocumentEnd[1815] oder WholeDocument[1822] und die Goto[1817] Methode für komplexere Auswahldefinitionen. Informationen zum Navigieren relativ zu einem bestimmten Dokumentbereich finden Sie unter den Methoden und Eigenschaften des AuthenticRange[1773] Objekts.

Beispiele

```
' ---------------------------------------
' Scripting environment - VBScript
' secure access to authentic view object
```

```
' ------------------------------------
DimobjDocument
SetobjDocument = Application.ActiveDocument
If(NotobjDocument Is Nothing) Then
	' we have an active document, now check for view mode
	If(objDocument.CurrentViewMode <> spyViewAuthentic) Then
		If(NotobjDocument.SwitchViewMode (spyViewAuthentic)) Then
			MsgBox "Active document does not support authentic view mode"
		Else
			' now it is safe to access the authentic view object
			DimobjAuthenticView
			SetobjAuthenticView = objDocument.AuthenticView
			' now use the authentic view object

		EndIf
	EndIf
Else
	MsgBox "No document is open"
EndIf
```

29.3.2.6.1 Events

29.3.2.6.1.1 OnBeforeCopy

Event: OnBeforeCopy()als Boolean

Skripting-Umgebung - VBScript:

```
Function On_AuthenticBeforeCopy()
	' On_AuthenticBeforeCopy = False  ' to disable operation
End Function
```

Skripting-Umgebung - JScript:

```
function On_AuthenticBeforeCopy()
{
	// return false;  /* to disable operation */
}
```

IDE Plugin:

```
IXMLSpyPlugIn.OnEvent(21, ...)  // nEventId = 21
```

Beschreibung

Dieses Event wird ausgelöst, bevor eine Kopieroperation an einem Dokument durchgeführt wird. Bei Rückgabe von *True* (oder nichts) wird die Kopieroperation gestattet, bei Rückgabe von *False* wird das Kopieren deaktiviert.

© 2016-2022 Altova GmbH

29.3.2.6.1.2 OnBeforeCut

Event: OnBeforeCut()als Boolean

Skripting-Umgebung - VBScript:
```
Function On_AuthenticBeforeCut()
	' On_AuthenticBeforeCut = False  ' to disable operation
End Function
```

Skripting-Umgebung - JScript:
```
function On_AuthenticBeforeCut()
{
	// return false;  /* to disable operation */
}
```

IDE Plugin:
```
IXMLSpyPlugIn.OnEvent(20, ...)  // nEventId = 20
```

Beschreibung
Dieses Event wird ausgelöst, bevor eine Ausschneideoperation an einem Dokument durchgeführt wird. Geben Sie *True* (oder nichts) zurück, um den Vorgang zu gestatten und *False* um den Vorgang zu deaktivieren.

29.3.2.6.1.3 OnBeforeDelete

Event: OnBeforeDelete()als Boolean

Skripting-Umgebung - VBScript:
```
Function On_AuthenticBeforeDelete()
	' On_AuthenticBeforeDelete = False  ' to disable operation
End Function
```
Skripting-Umgebung - JScript:
```
function On_AuthenticBeforeDelete()
{
	// return false;  /* to disable operation */
}
```

IDE Plugin:
```
IXMLSpyPlugIn.OnEvent(22, ...)  // nEventId = 22
```

Beschreibung
Dieses Event wird ausgelöst, bevor eine Löschoperation an einem Dokument durchgeführt wird. Geben Sie *True* (oder nichts) zurück, um den Vorgang zu gestatten und *False* um den Vorgang zu deaktivieren.

29.3.2.6.1.4 OnBeforeDrop

Event: OnBeforeDrop (*i_nXPos*als Long,*i_nYPos*als Long,*i_ipRange* als AuthenticRange[1773], *i_ipData*als cancelBoolean

Skripting-Umgebung - VBScript:
```
Function On_AuthenticBeforeDrop(nXPos, nYPos, objRange, objData)
        ' On_AuthenticBeforeDrop = False  ' to disable operation
End Function
```

Skripting-Umgebung - JScript:
```
function On_AuthenticBeforeDrop(nXPos, nYPos, objRange, objData)
{
        // return false;  /* to disable operation */
}
```

IDE Plugin:
```
IXMLSpyPlugIn.OnEvent(11, ...)  // nEventId = 11
```

Beschreibung
Dieses Event wird ausgelöst, nachdem ein Objekt mit der Maus in das Applikationsfenster gezogen wurde. Alle mit dem Event in Zusammehang stehenden Informationen werden als Parameter übergeben.

Die ersten beiden Parameter definieren die Mausposition zum Zeitpunkt des Auftretens des Event. Der Parameter *objRange* übergibt ein Bereichsobjekt, das das XML-Element unterhalb der Mausposition auswählt. Der Wert dieses Parameters kann *NULL* sein. Überprüfen Sie dies, bevor Sie das Bereichsobjekt aufrufen. Mit Hilfe des Parameters *objData* können Sie Informationen über das gezogene Objekt aufrufen.

Geben Sie *False* zurück, um den Drop-Vorgang abzubrechen. Geben Sie *True* (oder nichts) zurück, um mit einem normalen Vorgang fortzusetzen.

Beispiele
```
' ---------------------------------------------------------------------------
' VB code snippet - connecting to object level events
' ---------------------------------------------------------------------------
' access XMLSpy (without checking for any errors)
DimobjSpy AsXMLSpyLib.Application
SetobjSpy = GetObject("", "XMLSpy.Application")

' this is the event callback routine connected to the OnBeforeDrop
' event of object objView
PrivateFunctionobjView_OnBeforeDrop(ByVali_nXPos AsLong, ByVali_nYPos AsLong,
                                    ByVali_ipRange AsIAuthenticRange,
                                    ByVali_ipData AsIAuthenticDataTransfer) AsBoolean

  If(Noti_ipRange Is Nothing) Then
     MsgBox ("Dropping on content is prohibited");
     ReturnFalse;
  Else
     ReturnTrue;
  EndIf
EndFunction

' use VBA keyword WithEvents to connect to object-level event
DimWithEvents objView AsXMLSpyLib.AuthenticView
SetobjView = objSpy.ActiveDocument.AuthenticView
```

 © 2016-2022 Altova GmbH

```
' continue here with something useful ...
' and serve the windows message loop
```

29.3.2.6.1.5 OnBeforePaste

Event: OnBeforePaste (*objData* als Variant, *strType* als String) als Boolean

Skripting-Umgebung - VBScript:

```
Function On_AuthenticBeforePaste(objData,strType)
        ' On_AuthenticBeforePaste = False  ' to disable operation
End Function
```

Skripting-Umgebung - JScript:

```
function On_AuthenticBeforePaste(objData,strType)
{
        // return false;  /* to disable operation */
}
```

IDE Plugin:

```
IXMLSpyPlugIn.OnEvent(19, ...)   // nEventId = 19
```

Beschreibung

Dieses Event wird ausgelöst, bevor eine Einfügeoperation am Dokument ausgeführt wird. Der Parameter *strType* ist entweder "TEXT", "UNICODETEXT" oder "IUNKNOWN". In den ersten beiden Fällen enthält *objData* eine String-Darstellung des Objekts, das eingefügt wird. Im letzteren Fall enthält *objData* einen Pointer auf eine IUnknown COM-Schnittstelle.

Geben Sie *True* (oder nichts) zurück, um die Einfügeoperation zu gestatten. Geben Sie *False* zurück, um die Operation zu deaktivieren.

29.3.2.6.1.6 OnBeforeSave

Event: OnBeforeSave (SaveAs flag) als Boolean

Beschreibung: Mit `OnBeforeSave` haben Sie die Möglichkeit den Benutzer vor einem Überschreiben des bestehenden Dokuments zu warnen oder das Dokument mit einem Schreibschutz zu versehen, wenn bestimmte Bedingungen nicht erfüllt werden. Das Event wird ausgelöst, bevor das Dialogfeld "Datei" angezeigt wird.

29.3.2.6.1.7 OnDragOver

Event: OnDragOver(*nXPos* als Long, *nYPos* als Long, *eMouseEvent* als SPYMouseEvent 2026, *objRange* als AuthenticRange 1773, *objData* als AuthenticDataTransfer) als Boolean

Skripting-Umgebung - VBScript:

```
Function On_AuthenticDragOver(nXPos, nYPos, eMouseEvent, objRange, objData)
```

```
        ' On_AuthenticDragOver = False  ' to disable operation
End Function
```

Skripting-Umgebung - JScript:

```
function On_AuthenticDragOver(nXPos, nYPos, eMouseEvent, objRange, objData)
{
        // return false;  /* to disable operation */
}
```

IDE Plugin:

```
IXMLSpyPlugIn.OnEvent(10, ...)  // nEventId = 10
```

Beschreibung

Dieses Event wird ausgelöst, wenn ein Objekt, von innerhalb oder außerhalb der Authentic-Ansicht mit der Maus über das Applikationsfenster gezogen wird. Alle mit dem Event in Zusammenhang stehenden Informationen werden als Parameter übergeben.

Die ersten drei Parameter definieren die Mausposition, den Status der Maustaste und den Status virtueller Schaltflächen zum Zeitpunkt, zu dem das Event auftritt. Der Parameter *objRange* übergibt ein Bereichsobjekt, das das XML-Element unterhalb der Mausposition auswählt. Der Wert dieses Parameters kann *NULL* sein. Überprüfen Sie dies, bevor Sie das Bereichsobjekt aufrufen. Mit Hilfe des Parameters *objData* können Sie Informationen über das gezogene Objekt aufrufen.

Geben Sie *False* zurück, um den Drag-Vorgang abzubrechen. Geben Sie *True* (oder nichts) zurück, um mit einem normalen Vorgang fortzusetzen.

Beispiele

```
' ---------------------------------------------------------------------
' VB code snippet - connecting to object level events
' ---------------------------------------------------------------------
' access XMLSpy (without checking for any errors)
DimobjSpy AsXMLSpyLib.Application
SetobjSpy = GetObject("", "XMLSpy.Application")

' this is the event callback routine connected to the OnDragOver
' event of object objView
PrivateFunctionobjView_OnDragOver(ByVali_nXPos AsLong, ByVali_nYPos AsLong,
                                  ByVali_eMouseEvent AsSPYMouseEvent,
                                  ByVali_ipRange AsIAuthenticRange,
                                  ByVali_ipData AsIAuthenticDataTransfer) AsBoolean

  If(((i_eMouseEvent AndspyShiftKeyDownMask) <> 0) And
          (Noti_ipRange Is Nothing)) Then
     MsgBox ("Floating over element " & i_ipRange.FirstXMLData.Parent.Name);
  EndIf

  ReturnTrue;
EndFunction

' use VBA keyword WithEvents to connect to object-level event
DimWithEvents objView AsXMLSpyLib.AuthenticView
SetobjView = objSpy.ActiveDocument.AuthenticView
```

 © 2016-2022 Altova GmbH

```
' continue here with something useful ...
' and serve the windows message loop
```

29.3.2.6.1.8 OnKeyboardEvent

Event: OnKeyboardEvent (*eKeyEvent*als SPYKeyEvent[2025], *nKeyCode*als Long, *nVirtualKeyStatus*als Long) als Boolean

Skripting-Umgebung - VBScript:

```
Function On_AuthenticKeyboardEvent(eKeyEvent, nKeyCode, nVirtualKeyStatus)
        ' On_AuthenticKeyboardEvent = True  ' to cancel bubbling of event
End Function
```

Skripting-Umgebung - JScript:

```
function On_AuthenticKeyboardEvent(eKeyEvent, nKeyCode, nVirtualKeyStatus)
{
        // return true;  /* to cancel bubbling of event */
}
```

IDE Plugin:

```
IXMLSpyPlugIn.OnEvent(30, ...)  // nEventId = 30
```

Beschreibung

Dieses Event wird für *WM_KEYDOWN*, *WM_KEYUP u*nd *WM_CHAR* Windows-Meldungen ausgelöst.

Der eigentliche Meldungstyp steht im *eKeyEvent* Parameter zur Verfügung. Der Status virtueller Tasten ist im Parameter *nVirtualKeyStatus* kombiniert. Verwenden Sie die im Enumerations-Datentyp SPYVirtualKeyMask[2033] definierten Bit-Masken, um eine Überprüfung der unterschiedlichen Tasten und ihrer Kombinationen durchzuführen.

29.3.2.6.1.9 OnLoad

Event: OnLoad ()

Beschreibung: Mit `OnLoad` können einige Funktionalitäten der Authentic-Ansicht eingeschränkt werden, wie im Beispiel unten gezeigt:

```
functionOn_AuthenticLoad(  )
{
    // We are disabling all entry helpers in order to prevent user from manipulating XML tree
    AuthenticView.DisableElementEntryHelper();
    AuthenticView.DisableAttributeEntryHelper();

    // We are also disabling the markup buttons for the same purpose
    AuthenticView.SetToolbarButtonState('AuthenticMarkupSmall',authenticToolbarButtonDisabled );
    AuthenticView.SetToolbarButtonState('AuthenticMarkupLarge',authenticToolbarButtonDisabled );
    AuthenticView.SetToolbarButtonState('AuthenticMarkupMixed',authenticToolbarButtonDisabled );
}
```

Im Beispiel wird der Status der Symbolleisten-Schaltflächen "Kleine Markup-Symbole", "Große Markup-Symbole" und "Gemischte Markup-Symbole" mit Hilfe der Schaltflächen-Identifier bearbeitet. Siehe vollständige Liste[1811].

29.3.2.6.1.10 OnMouseEvent

Event: OnMouseEvent (*nXPos* als Long, *nYPos* als Long, *eMouseEvent* als SPYMouseEvent[2026], *objRange* als AuthenticRange[1773]) als Boolean

Skripting-Umgebung - VBScript:
```
Function On_AuthenticMouseEvent(nXPos, nYPos, eMouseEvent, objRange)
        ' On_AuthenticMouseEvent = True  ' to cancel bubbling of event
End Function
```

Skripting-Umgebung - JScript:
```
function On_AuthenticMouseEvent(nXPos, nYPos, eMouseEvent, objRange)
{
        // return true;  /* to cancel bubbling of event */
}
```

IDE Plugin:
```
IXMLSpyPlugIn.OnEvent(31, ...)  // nEventId = 31
```

Beschreibung
Dieses Event wird für jede Mausbewegung und jede Mausschaltflächen-Windows-Meldung ausgelöst.

Der eigentliche Meldungstyp und der Status der Maustaste stehen im *eMouseEvent* Parameter zur Verfügung. Verwenden Sie die im Enumerations-Datentyp SPYMouseEvent[2026] definierten Bit-Masken, um eine Überprüfung auf die unterschiedlichen Meldungen, Tastenstatus und ihre Kombinationen durchzuführen.

Der Parameter *objRange* definiert den Teil des Dokuments, der sich an der aktuellen Mauscursorposition befindet. Das Bereichsobjekt wählt immer einen vollständigen Tag des Dokuments aus. (Dies kann sich in zukünftigen Versionen ändern, wenn ein genauerer Positionierungsmechanismus zur Verfügung steht). Wenn sich an der aktuellen Position kein auswählbarer Teil des Dokuments befindet, ist das Bereichsobjekt *Null*.

29.3.2.6.1.11 OnSelectionChanged

Event: OnSelectionChanged (*objNewSelection* als AuthenticRange[1773])

Skripting-Umgebung - VBScript:
```
Function On_AuthenticSelectionChanged(objNewSelection)
End Function
```

Skripting-Umgebung - JScript:
```
function On_AuthenticSelectionChanged(objNewSelection)
{
}
```

 © 2016-2022 Altova GmbH

IDE Plugin:
IXMLSpyPlugIn.OnEvent(23, ...) // nEventId = 23

Beschreibung
Dieses Event wird ausgelöst, wenn die sich die Auswahl auf der Benutzeroberfläche ändert.

Beispiele
```
' ---------------------------------------------------------------------
' VB code snippet - connecting to object level events
' ---------------------------------------------------------------------
' access XMLSpy (without checking for any errors)
DimobjSpy AsXMLSpyLib.Application
SetobjSpy = GetObject("", "XMLSpy.Application")

' this is the event callback routine connected to the OnSelectionChanged
' event of object objView
PrivateSubobjView_OnSelectionChanged (ByVali_ipNewRange AsXMLSpyLib.IAuthenticRange)
   MsgBox ("new selection: " & i_ipNewRange.Text)
EndSub

' use VBA keyword WithEvents to connect to object-level event
DimWithEvents objView AsXMLSpyLib.AuthenticView
SetobjView = objSpy.ActiveDocument.AuthenticView

' continue here with something useful ...
' and serve the windows message loop
```

29.3.2.6.1.12 OnToolbarButtonClicked

Event: OnToolbarButtonClicked (Button identifier)

Beschreibung: `OnToolbarButtonClicked` wird ausgelöst, wenn der Benutzer auf eine Symbolleisten-Schaltfläche klickt. Über den Parameter Schaltflächen-Identifier, kann ermittelt werden, auf welche Schaltfläche geklickt wurde. Unten sehen Sie die Liste der vordefinierten Schaltflächen-Identifier:

- AuthenticPrint
- AuthenticPrintPreview
- AuthenticUndo
- AuthenticRedo
- AuthenticCut
- AuthenticCopy
- AuthenticPaste
- AuthenticClear
- AuthenticMarkupHide
- AuthenticMarkupLarge
- AuthenticMarkupMixed
- AuthenticMarkupSmall
- AuthenticValidate
- AuthenticChangeWorkingDBXMLCell
- AuthenticSave
- AuthenticSaveAs

- AuthenticReload
- AuthenticTableInsertRow
- AuthenticTableAppendRow
- AuthenticTableDeleteRow
- AuthenticTableInsertCol
- AuthenticTableAppendCol
- AuthenticTableDeleteCol
- AuthenticTableJoinCellRight
- AuthenticTableJoinCellLeft
- AuthenticTableJoinCellAbove
- AuthenticTableJoinCellBelow
- AuthenticTableSplitCellHorizontally
- AuthenticTableSplitCellVertically
- AuthenticTableAlignCellContentTop
- AuthenticTableCenterCellVertically
- AuthenticTableAlignCellContentBottom
- AuthenticTableAlignCellContentLeft
- AuthenticTableCenterCellContent
- AuthenticTableAlignCellContentRight
- AuthenticTableJustifyCellContent
- AuthenticTableInsertTable
- AuthenticTableDeleteTable
- AuthenticTableProperties
- AuthenticAppendRow
- AuthenticInsertRow
- AuthenticDuplicateRow
- AuthenticMoveRowUp
- AuthenticMoveRowDown
- AuthenticDeleteRow
- AuthenticDefineEntities
- AuthenticXMLSignature

Der Benutzer kann für seine benutzerdefinierten Schaltflächen seine eigenen Identifier hinzufügen. Beachten Sie bitte, dass der Benutzer aufpassen muss, da die Identifier nicht auf Eindeutigkeit überprüft werden. Dieselben Identifier können zur Kennzeichnung von Schaltflächen in `Set/GetToolbarState()` COM API-Aufrufen verwendet werden. Durch Hinzufügen von Code für unterschiedliche Schaltflächen hat der Benutzer die Möglichkeit, das Verhalten der Symbolleiste der Authentic-Ansicht vollständig neu zu definieren, indem er seine eigenen Methoden zur Tabellenbearbeitung usw. hinzufügt.

29.3.2.6.1.13 OnToolbarButtonExecuted

Event: OnToolbarButtonExecuted (Button identifier)

Beschreibung: `OnToolbarButtonClicked` wird ausgelöst, wenn ein Benutzer auf eine Symbolleisten-Schaltfläche klickt. Über den Parameter Schaltflächen-Identifier, kann ermittelt werden, auf welche Schaltfläche geklickt wurde. Siehe Liste der vordefinierten vordefinierten Schaltflächen-Identifier[1811].

`OnToolbarButtonExecuted` wird ausgelöst, nachdem die Symbolleisten-Aktion ausgeführt wurde und eignet sich z.B. zum Hinzufügen von aktualisiertem Code, wie im Beispiel unten gezeigt:

```
//event fired when a toolbar button action was executed
```

 © 2016-2022 Altova GmbH

```
functionOn_AuthenticToolbarButtonExecuted(varBtnIdentifier )
{
    // After whatever command user has executed - make sure to update toolbar button states
    UpdateOwnToolbarButtonStates();
}
```

In diesem Fall ist `UpdateOwnToolbarButtonStates` eine in den globalen Deklarationen definierte Benutzerfunktion.

29.3.2.6.1.14 OnUserAddedXMLNode

Event: OnUserAddedXMLNode (XML node)

Beschreibung: `OnUserAddedXMLNode` wird ausgelöst, wenn der Benutzer einen XML-Node als Primäraktion hinzufügt. Dies geschieht in Situationen, in denen der Benutzer eine der folgenden Aktionen ausführt:

- Klicken auf automatisch hinzugefügte Hyperlinks (siehe Beispiel OnUserAddedXMLNode.sps)
- Auswahl der Kontextmenübefehle "Einfügen...", "Einfügen nach...", "Einfügen vor..."
- Klicken auf "Zeile anhängen", beim Einfügen neuer Symbolleisten-Schaltflächen
- Ausführen von "Einfügen nach...", "Einfügen vor..." Aktionen in Elementeingabehilfen (außerhalb von StyleVision)

Das Event wird **nicht** bei "Zeile duplizieren" oder, wenn der Node extern (z.B. über die COM API) hinzugefügt wurde, oder bei "Anwenden" (z.B. Textstatus-Symbole) oder in XML-Tabellenoperationen bzw. in Datenbankoperationen ausgelöst.

Der Event-Parameter ist das XML-Node-Objekt, das hinzugefügt wurde, wodurch der Benutzer die Möglichkeit hat, den hinzugefügten XML-Node zu bearbeiten. Ein ausführliches Beispiel für einen Event Handler finden Sie in der Datei `OnUserAddedXMLNode.sps`.

29.3.2.6.2 Application

Eigenschaft: Application als Application[1746] (schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler

2000	Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.6.3 AsXMLString

Eigenschaft: AsXMLString als String

Beschreibung

Gibt den Dokumentinhalt als XML-String zurück oder definiert ihn als solchen. Wenn der Inhalt auf einen neuen Wert gesetzt wird, wird die gerade verwendete Schema- oder sps-Datei nicht geändert. Wenn der neue XMLStringder aktuellen Schemadatei nicht entspricht, wird Fehler 2011 zurückgegeben.

Fehler

2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2011 AsXMLStringwurde auf einen Wert gesetzt, der gemäß der aktuellen Schemadatei kein gültiger XML-Code ist.

29.3.2.6.4 ContextMenu

Eigenschaft: ContextMenu() als ContextMenu

Beschreibung

Über die Eigenschaft `ContextMenu` erhält der Benutzer Zugriff auf das Kontextmenü. Am besten lässt sich dies über den Event Handler `OnContextMenuActivated` durchführen.

Fehler

2000 Ungültiges Objekt.
2005 Ungültiger Parameter.

29.3.2.6.5 CreateXMLNode

Methode: CreateXMLNode (*nKind* als SPYXMLDataKind) als XMLData

Rückgabewert

Die Methode gibt das neue XMLData[1970] Objekt zurück.

Beschreibung

Um ein neues XMLDataObjekt zu erstellen, verwenden Sie die Methode CreateXMLNode().

Fehler

2000 Ungültiges Objekt.
2012 XML-Node kann nicht erstellt werden.

29.3.2.6.6 DisableAttributeEntryHelper

Methode: DisableAttributeEntryHelper`()`

Beschreibung

DisableAttributeEntryHelper`()` deaktiviert die Attribut-Eingabehilfe in XMLSpy, Authentic Desktop und dem Authentic Browser Plug-in.

Fehler

2000 Ungültiges Objekt.

© 2016-2022 Altova GmbH

29.3.2.6.7 DisableElementEntryHelper

Methode: DisableElementEntryHelper()

Beschreibung
DisableElementEntryHelper() deaktiviert die Element-Eingabehilfe in XMLSpy, Authentic Desktop und dem Authentic Browser Plug-in.

Fehler
2000 Ungültiges Objekt.

29.3.2.6.8 DisableEntityEntryHelper

Methode: DisableEntityEntryHelper()

Beschreibung
DisableEntityEntryHelper() deaktiviert die Entity-Eingabehilfe in XMLSpy, Authentic Desktop und dem Authentic Browser Plug-in.

Fehler
2000 Ungültiges Objekt.

29.3.2.6.9 DocumentBegin

Eigenschaft: DocumentBegin als AuthenticRange[1773] (schreibgeschützt)

Beschreibung
Ruft ein Bereichsobjekt auf, das auf den Anfang des Dokuments zeigt.

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.6.10 DocumentEnd

Eigenschaft: DocumentEndals AuthenticRange[1773](schreibgeschützt)

Beschreibung
Ruft ein Bereichsobjekt auf, das auf das Ende des Dokuments zeigt.

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.6.11 DoNotPerformStandardAction

Methode DoNotPerformStandardAction ()

Beschreibung
`DoNotPerformStandardAction()` dient als Abbrech-Bubble für Makros und bricht die weitere Ausführung ab, nachdem das Makro fertig ausgeführt wurde.

Fehler
2000 Ungültiges Objekt.

29.3.2.6.12 EvaluateXPath

Methode: EvaluateXPath (XMLData als XMLData[1970], strExpression als string) strValue als string

Rückgabewert
Diese Methode gibt einen String zurück.

Beschreibung
`EvaluateXPath()` führt einen XPath-Ausdruck im angegebenen XML-Kontext-Node aus. Das Ergebnis wird als String zurückgegeben. Im Fall einer Sequenz ist der String durch Leerzeichen getrennt.

Fehler
2000 Ungültiges Objekt.
2005 Ungültiger Parameter.
2008 Interner Fehler.
2013 XPath-Fehler.

29.3.2.6.13 Event

Eigenschaft: Eventals AuthenticEvent(schreibgeschützt)

Beschreibung
Über diese Eigenschaft erhalten Sie auf dieselbe Art wie bei OldAuthenticView.eventZugriff auf die Parameter des letzten Event. Da nun alle Events für die Skripting-Umgebung und externe Clients mit Parameters verfügbar sind, sollte dieseEventEigenschaft nur von IDE-Plug-ins aus verwendet werden.

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.6.14 EventContext

Eigenschaft: EventContext() als EventContext

Beschreibung
Über die Eigenschaft `EventContext` [1769] erhalten Sie Zugriff auf den Kontext des laufenden Makros. Nähere Informationen dazu finden Sie unter der Beschreibung zur `EventContext` [1769] Schnittstelle.

Fehler
2000 Ungültiges Objekt.

29.3.2.6.15 GetToolbarButtonState

Methode: GetToolbarButtonState (`ButtonIdentifier` als string) als `AuthenticToolbarButtonState`

Rückgabewert
Die Methode gibt `AuthenticToolbarButtonState` zurück.

Beschreibung
`GetToolbarButtonState` fragt den Status der Symbolleisten-Schaltfläche ab und gestattet dem Benutzer, die durch ihren Schaltflächen-Identifier gekennzeichnete Schaltfläche (siehe Liste oben [1811]) zu aktivieren bzw. zu deaktivieren. Auf diese Art können die Symbolleisten-Schaltflächen z.B. permanent deaktiviert werden. Eine andere Verwendung der Methode ist `SetToolbarButtonState` in den `OnSelectionChanged` Event Handler zu setzen, da die Symbolleisten-Schaltflächen regelmäßig aktualisiert werden, wenn sich die Auswahl im Dokument ändert.

Der Status der Symbolleisten-Schaltflächen wird unter der Liste der Enumerationen [2022] aufgelistet.

Der Standardzustand bedeutet, dass die Aktivierung/Deaktivierung der Schaltfläche über AuthenticView erfolgt. Wenn der Benutzer den Zustand auf aktiviert oder deaktiviert setzt, bleibt die Schaltfläche so lange in diesem Zustand, solange der Benutzer den Zustand nicht ändert.

Fehler
2000 Ungültiges Objekt.
2005 Ungültiger Parameter.
2008 Interner Fehler.
2014 Ungültiger Schaltflächen-Identifier.

29.3.2.6.16 Goto

Methode: Goto(*eKind*als SPYAuthenticElementKind [2021], *nCount*als Long, *eFrom* als *SPYAuthenticDocumentPosition* [2021]) als *AuthenticRange* [1773]

Beschreibung

Ruft ein Bereichsobjekt auf, das auf den Anfgang des *nCount* Elements vom Typ *eKind* zeigt. Die Anfangsposition wird durch den Parameter *eFrom* definiert. Verwenden Sie positive Werte für *nCount* um in Richtung des Dokumentendes zu navigieren. Verwenden Sie negative Wert um zum Anfang des Dokuments hin zu navigieren.

Fehler

2000	Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2003	Das Ziel liegt hinter dem Ende des Dokuments.
2004	Das Ziel liegt vor dem Anfang des Dokuments.
2005	Es wurde eine ungültige Elementart definiert. Die Ausgangsposition des Dokuments ist keine *spyAuthenticDocumentBegin* oder *spyAuthenticDocumentEnd* Position. Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ----------------------------------------
' Scripting environment - VBScript
' ----------------------------------------
DimobjAuthenticView
' we assume that the active document is open in authentic view mode
SetobjAuthenticView = Application.ActiveDocument.AuthenticView

OnErrorResume Next
DimobjRange
' goto beginning of first table in document
SetobjRange = objAuthenticView.Goto (spyAuthenticTable, 1, spyAuthenticDocumentBegin)
If(Err.number = 0) Then
        objRange.Select()
Else
        MsgBox "No table found in document"
EndIf
```

29.3.2.6.17 IsRedoEnabled

Eigenschaft: IsRedoEnabled als Boolean (schreibgeschützt)

Beschreibung

True, wenn Wiederhol-Schritte verfügbar sind und Redo[1820] möglich ist.

Fehler

2000	Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.6.18 IsUndoEnabled

Eigenschaft: IsUndoEnabled als Boolean (schreibgeschützt)

Beschreibung

True, wenn Rückgängig-Schritte verfügbar sind und Undo[1821] möglich ist.

© 2016-2022 Altova GmbH

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.6.19 MarkupVisibility

Eigenschaft: MarkupVisibilityals SPYAuthenticMarkupVisibility 2022

Beschreibung
Setzt oder ruft die aktuelle Sichtbarkeit des Markup-Codes ab.

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005 Es wurde ein ungültiger Enumerationswert angegeben.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.6.20 Parent

Eigenschaft: Parent als Document 1838 (schreibgeschützt)

Beschreibung
Ruft das in dieser Ansicht angezeigte Dokument auf.

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.6.21 Print

Methode: Print (*bWithPreview* als Boolean, *bPromptUser* als Boolean)

Beschreibung
Druckt das in dieser Ansicht angezeigte Dokument. Wenn *bWithPreview*auf *True*gesetzt ist, wird das Dialogfeld "Druckvorschau" angezeigt. Wenn *bPromptUser* auf *True*gesetzt ist, wird das Dialogfeld "Drucken" angezeigt. Wenn beide Parameter auf *False*gesetzt sind, wird das Dokument ohne weitere Eingabe durch den Benutzer gedruckt.

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.

29.3.2.6.22 Redo

Methode: Redo() als Boolean

Beschreibung
Stellt die durch den letzten Rückgängig-Befehl rückgängig gemachte Änderung wieder her.

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.6.23 Selection

Eigenschaft: Selection als AuthenticRange[1773]

Beschreibung
Setzt oder ruft die aktuelle Textauswahl auf der Benutzeroberfläche ab.

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2002 Es ist keine Cursorauswahl aktiv.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' ----------------------------------------
' Scripting environment - VBScript
' ----------------------------------------
DimobjAuthenticView
' we assume that the active document is open in authentic view mode
SetobjAuthenticView = Application.ActiveDocument.AuthenticView

' if we are the end of the document, re-start at the beginning
If(objAuthenticView.Selection.IsEqual(objAuthenticView.DocumentEnd)) Then
        objAuthenticView.Selection = objAuthenticView.DocumentBegin
Else
        ' objAuthenticView.Selection = objAuthenticView.Selection.GotoNextCursorPosition()
        ' or shorter:
        objAuthenticView.Selection.GotoNextCursorPosition().Select
EndIf
```

29.3.2.6.24 SetToolbarButtonState

Methode: SetToolbarButtonState (`ButtonIdentifier` als string, `AuthenticToolbarButtonState` state)

Beschreibung
`SetToolbarButtonState` fragt den Status der Symbolleisten-Schaltfläche ab und gestattet dem Benutzer, die durch ihren Schaltflächen-Identifier gekennzeichnete Schaltfläche (siehe Liste oben[1811]) zu aktivieren bzw. zu deaktivieren. Auf diese Art können die Symbolleisten-Schaltflächen z.B. permanent deaktiviert werden. Eine

© 2016-2022 Altova GmbH

andere Verwendung der Methode ist `SetToolbarButtonState` in den `OnSelectionChanged` Event Handler zu setzen, da die Symbolleisten-Schaltflächen regelmäßig aktualisiert werden, wenn sich die Auswahl im Dokument ändert.

Der Status der Symbolleisten-Schaltflächen wird unter der Liste der Enumerationen[2022] aufgelistet.

Der Standardzustand bedeutet, dass die Aktivierung/Deaktivierung der Schaltfläche über AuthenticView erfolgt. Wenn der Benutzer den Zustand auf aktiviert oder deaktiviert setzt, bleibt die Schaltfläche so lange in diesem Zustand, solange der Benutzer den Zustand nicht ändert.

Fehler

2000 Ungültiges Objekt.
2008 Interner Fehler.
2014 Ungültiger Schaltflächen-Identifier.

29.3.2.6.25 Undo

Methode: Undo()als Boolean

Beschreibung
Macht die letzte Änderung des Dokuments innerhalb dieser Ansicht rückgängig.

Fehler

2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.6.26 UpdateXMLInstanceEntities

Methode: UpdateXMLInstanceEntities()

Beschreibung
Aktualisiert die interne Darstellung der deklarierten Entities und befüllt die Eingabehilfen erneut. Zusätzlich dazu wird der Validator neu geladen, sodass die XML-Datei korrekt validiert werden kann. Beachten Sie bitte, dass dadurch auch Schemadateien neu geladen werden können.

Fehler
Die Methode gibt niemals einen Fehler zurück.

Beispiel

```
// ----------------------------------------
// Scripting environment - JavaScript
// ----------------------------------------
if(Application.ActiveDocument &&(Application.ActiveDocument.CurrentViewMode ==4))
{
	varobjDocType;
	objDocType = Application.ActiveDocument.DocEditView.XMLRoot.GetFirstChild(10);

	if(objDocType)
```

```
	{
		varobjEntity = Application.ActiveDocument.CreateChild(14);
		objEntity.Name = "child";
		objEntity.TextValue = "SYSTEM \"child.xml\"";
		objDocType.AppendChild(objEntity);

		Application.ActiveDocument.AuthenticView.UpdateXMLInstanceEntities();
	}
}
```

29.3.2.6.27 WholeDocument

Eigenschaft: WholeDocument als AuthenticRange[1773] (schreibgeschützt)

Beschreibung
Ruft ein Bereichsobjekt ab, das das gesamte Dokument auswählt.

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.6.28 XMLDataRoot

Eigenschaft: XMLDataRoot als XMLData[1970] (schreibgeschützt)

Beschreibung
Gibt das am höchsten gestufte XMLData-Element des aktuellen Dokuments zurück oder definiert dieses. Dieses Element beschreibt normalerweise die Dokumentstruktur und wäre von der Art spyXMLDataXMLDocStruct, spyXMLDataXMLEntityDocStruct oder spyXMLDataDTDDocStruct.

Fehler
2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.7 CodeGeneratorDlg

Nur verfügbar/aktiv in der Enterprise Edition. Bei Aufruf von einer anderen Version aus wird ein Fehler zurückgegeben.

Eigenschaften und Methoden

Standard-Automation-Eigenschaften
Application[1823]
Parent[1827]

Eigenschaften zur Auswahl der Programmiersprache
ProgrammingLanguage[1827]
TemplateFileName[1828]

© 2016-2022 Altova GmbH

Einstellungen für C++-Code
CPPSettings_DOMType[1824]
CPPSettings_LibraryType[1825]
CPPSettings_UseMFC[1825]
CPPSettings_GenerateVC6ProjectFile[1824]
CPPSettings_GenerateVSProjectFile[1825]

Einstellungen für C#-Code
CSharpSettings_ProjectType[1826]

Behandlung von Dialogfeldern für die oben genannten Codegenerierungseigenschaften
PropertySheetDialogAction[1828]

Eigenschaften zur Auswahl des Ausgabepfads
OutputPath[1826]
OutputPathDialogAction[1826]

Darstellung des Ergebnisses
OutputResultDialogAction[1827]

Beschreibung
Mit Hilfe dieses Objekts können Sie die Generierung von Programmcode für Schemadateien konfigurieren. Die Methode GenerateProgramCode[1855] erwartet einen CodeGeneratorDlg als Parameter, um die Codegenerierung sowie die damit in Zusammenhang stehenden Interaktionen des Benutzers zu konfigurieren.

29.3.2.7.1 Application

Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Eigenschaft: Applicationals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.7.2 CompatibilityMode (nicht mehr verwendet)

Eigenschaft: CompatibilityModeals Boolean
Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung
Wird auf truegesetzt, um Code zu generieren, der mit XMLSpy 2005R3 kompatibel ist. Wird auf falsegesetzt, um neu hinzugefügte Codegenerierungsfeatures zu verwenden.

Fehler

2200 Das Objekt ist nicht mehr gültig.
2201 Es wurde eine ungültige Aktion als Parameter übergeben oder es wurde eine ungültige Adresse für den Rückgabeparameter angegeben.

29.3.2.7.3 CPPSettings_DOMType

Eigenschaft: CPPSettings_DOMTypeals SPYDOMType 2023
Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung
Definiert eine der Einstellungen für die Konfiguration der Generierung von C++-Code.

Fehler

2200 Das Objekt ist nicht mehr gültig.
2201 Es wurde eine ungültige Aktion als Parameter übergeben oder es wurde eine ungültige Adresse für den Rückgabeparameter angegeben.

29.3.2.7.4 CPPSettings_GenerateVC6ProjectFile

Eigenschaft: CPPSettings_GenerateVC6ProjectFileals Boolean
Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung
Definiert eine der Einstellungen für die Konfiguration der Generierung von C++-Code.

Fehler

2200 Das Objekt ist nicht mehr gültig.
2201 Es wurde eine ungültige Aktion als Parameter übergeben oder es wurde eine ungültige Adresse für den Rückgabeparameter angegeben.

29.3.2.7.5 CPPSettings_GenerateGCCMakefile

Eigenschaft: CPPSettings_GenerateGCCMakefileals Boolean
Steht nur in der Enterprise Edition zur Verfügung. Bei Aufruf von einer anderen Version aus, wird eine Fehlermeldung zurückgegeben.

Beschreibung
Erstellt Makefiles zum Kompilieren des generierten Codes unter Linux mit GCC.

© 2016-2022 Altova GmbH

Fehler

2200	Das Objekt ist nicht mehr gültig.
2201	Es wurde eine ungültige Aktion als Parameter übergeben oder es wurde eine ungültige Adresse für den Rückgabeparameter angegeben.

29.3.2.7.6 CPPSettings_GenerateVSProjectFile

Eigenschaft: CSharpSettings_GenerateVSProjectFileals SPYProjectType 2027
Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung
Definiert eine der Einstellungen für die Konfiguration der Generierung von C++-Code. Nur spyVisualStudio2005Project(=4), spyVisualStudio2008Project (=5) und `spyVisualStudio2010Project (=6)` sind gültige Projekttypen.

Fehler

2200	Das Objekt ist nicht mehr gültig.
2201	Es wurde eine ungültige Aktion als Parameter übergeben oder es wurde eine ungültige Adresse für den Rückgabeparameter angegeben.

29.3.2.7.7 CPPSettings_LibraryType

Eigenschaft: CPPSettings_LibraryTypeals SPYLibType 2025
Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung
Definiert eine der Einstellungen für die Konfiguration der Generierung von C++-Code.

Fehler

2200	Das Objekt ist nicht mehr gültig.
2201	Es wurde eine ungültige Aktion als Parameter übergeben oder es wurde eine ungültige Adresse für den Rückgabeparameter angegeben.

29.3.2.7.8 CPPSettings_UseMFC

Eigenschaft: CPPSettings_UseMFCals Boolean
Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung
Definiert eine der Einstellungen für die Konfiguration der Generierung von C++-Code.

Fehler

2200 Das Objekt ist nicht mehr gültig.
2201 Es wurde eine ungültige Aktion als Parameter übergeben oder es wurde eine ungültige Adresse für den Rückgabeparameter angegeben.

29.3.2.7.9 CSharpSettings_ProjectType

Eigenschaft: CSharpSettings_ProjectTypeals SPYProjectType[2027]
Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung
Definiert die einzige Einstellung für die Konfiguration der Generierung von C#-Code.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Es wurde eine ungültige Aktion als Parameter übergeben oder es wurde eine ungültige Adresse für den Rückgabeparameter angegeben.

29.3.2.7.10 OutputPath

Eigenschaft: OutputPathals String
Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung
Wählt das Basisverzeichnis für den gesamten generierten Code aus.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.7.11 OutputPathDialogAction

Eigenschaft: OutputPathDialogActionals SPYDialogAction[2023]
Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung
Definiert, wie das untergeordnete Dialogfeld zur Auswahl des Ausgabepfads für die Codegenerierung behandelt wird. Setzen Sie diese Wert auf *spyDialogUserInput(2),*damit das Dialogfeld standardmäßig mit dem aktuellen Wert der Eigenschaft OutputPath[1826]angezeigt wird. Verwenden Sie *spyDialogOK(0),*damit das Dialogfeld nicht angezeigt wird.

Fehler
2200 Das Objekt ist nicht mehr gültig.

© 2016-2022 Altova GmbH

2201 Es wurde eine ungültige Aktion als Parameter übergeben oder es wurde eine ungültige Adresse für den Rückgabeparameter angegeben.

29.3.2.7.12 OutputResultDialogAction

Eigenschaft: OutputResultDialogActionals SPYDialogAction[2023]
Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung
Definiert, wie das untergeordnete Dialogfeld behandelt wird, das fragt, ob das Ergebnis der Codegenerierung angezeigt werden soll. Setzen Sie diesen Wert auf *spyDialogUserInput(2),*damit das Dialogfeld angezeigt wird. Verwenden Sie *spyDialogOK(0),*damit das Dialogfeld dem Benutzer nicht angezeigt wird.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Es wurde eine ungültige Aktion als Parameter übergeben oder es wurde eine ungültige Adresse für den Rückgabeparameter angegeben.

29.3.2.7.13 Parent

Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Eigenschaft: Parentals Dialogs[1835](schreibgeschützt)

Beschreibung
Ruft das übergeordnete Element des Objekts auf.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.7.14 ProgrammingLanguage

Eigenschaft: ProgrammingLanguageals ProgrammingLanguage[2027]
Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung
Wählt die Ausgabesprache aus, in der der Code generiert werden soll.

Achtung: Wenn Sie diese Eigenschaft auf C++, C# oder Java setzen, wird auch die Eigenschaft TemplateFileName[1828]auf die entsprechende mit XMLSpyinstallierte Vorlagendatei gesetzt. Wenn Sie C++, C# oder Java-Code basierend auf Ihren eigenen Vorlagen generieren möchten, definieren Sie zuerst die Programmiersprache und wählen Sie dann Ihre Vorlagendatei aus.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.7.15 PropertySheetDialogAction

Eigenschaft: PropertySheetDialogActionals SPYDialogAction[2023]
Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung
Definiert, wie das untergeordnete Dialogfeld, mit dem die Codegenerierung konfiguriert wird, behandelt wird. Setzen Sie diesen Wert auf *spyDialogUserInput(2)*um das Dialogfeld mit den aktuellen Werten als Standardwerten anzuzeigen. Verwenden Sie *spyDialogOK(0),*damit das Dialogfeld nicht angezeigt wird.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Es wurde eine ungültige Aktion als Parameter übergeben oder es wurde eine ungültige Adresse für den Rückgabeparameter angegeben.

29.3.2.7.16 TemplateFileName

Eigenschaft: TemplateFileNameals String
Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung
Wählt die Vorlagendatei für die Codegenerierung aus. Im Ordner "SPL" Ihres XMLSpyInstallationsverzeichnisses sind Vorlagendateien für C++, C# oder Java enthalten.

Wenn Sie diese Eigenschaft auf eine der Codegenerierungsvorlagendateien Ihrer XMLSpyInstallation setzen, wird die Eigenschaft ProgrammingLanguage[1827]automatisch auf den entsprechenden Wert gesetzt.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.8 DatabaseConnection

Eigenschaften für den Import und Export
File[1831]oder
ADOConnection[1829]oder
ODBCConnection[1833]

Eigenschaften nur für den Import
DatabaseKind[1831]
SQLSelect[1834]

© 2016-2022 Altova GmbH

AsAttributes 1830
ExcludeKeys 1831
IncludeEmptyElements 1832
NumberDateTimeFormat 1833
NullReplacement 1832
CommentIncluded 1830

Eigenschaften nur für den Export
CreateMissingTables 1830
CreateNew 1830
TextFieldLen 1834
DatabaseSchema 1831

Eigenschaften für die Generierung eines XML-Schemas anhand der Datenbankstruktur
PrimaryKeys 1833
ForeignKeys 1832
UniqueKeys 1834
SchemaExtensionType 1833
SchemaFormat 1834
ImportColumnsType 1832

Beschreibung
DatabaseConnectiondefiniert die Parameter für die Datenbankverbindung.

Bitte beachten Sie, dass sich die Eigenschaften der DatabaseConnection Schnittstelle auf die Einstellungen des Import- und Exportdialogfelds von XMLSpy beziehen.

29.3.2.8.1 ADOConnection

Eigenschaft: ADOConnectionals String

Beschreibung
Die Eigenschaft ADOConnectionenthält einen Connection String. Verwenden Sie entweder diese Eigenschaft oder ODBCConnection 1833 oder File 1831 ,um eine Datenbank zu referenzieren.

Fehler
Es werden keine Fehlercodes zurückgegeben.

Beispiel

```
DimobjSpyConn AsDatabaseConnection
SetobjSpyConn = objSpy.GetDatabaseSettings

DimobjADO AsDataLinks
SetobjADO = CreateObject("DataLinks")

IfNot(objADO Is Nothing) Then
        DimobjConn AsConnection
        SetobjConn = objADO.PromptNew
        objSpyConn.ADOConnection = objConn.ConnectionString
EndIf
```

29.3.2.8.2 AsAttributes

Eigenschaft: AsAttributesals Boolean

Beschreibung
Setzen Sie AsAttributesauf "true", wenn alle Importfelder initialisiert werden sollen, um als Attribute importiert zu werden. Der Standardwert ist "false" und initialisiert alle Felder, um als Elemente importiert zu werden. Diese Eigenschaft wird nur in Aufrufen von Application.GetDatabaseImportElementList[1753] verwendet.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.3 CommentIncluded

Eigenschaft: CommentIncludedals Boolean

Beschreibung
Diese Eigenschaft legt fest, ob zusätzliche Kommentare zum generierten XML-Code hinzugefügt werden. Der Standardwert ist true. Diese Eigenschaft wird nur beim Import aus einer Datenbank verwendet.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.4 CreateMissingTables

Eigenschaft: CreateMissingTablesals Boolean

Beschreibung
Wenn CreateMissingTables"true" ist, werden Tabellen, die nicht bereits in der Exportdatenbank definiert sind, beim Export erstellt. Die Standardeinstellung ist true. Diese Eigenschaft wird nur beim Export in Datenbanken verwendet.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.5 CreateNew

Eigenschaft: CreateNewals Boolean

Beschreibung
Setzen Sie CreateNewauf "true", wenn Sie beim Export eine neue Datenbank erstellen möchten. Eine möglicherweise bereits vorhandene Datenbank wird dabei überschrieben. Siehe auch DatabaseConnection.File[1831]. Der Standardwert ist "false". Diese Eigenschaft wird nur beim Export in eine Datenbank verwendet.

Fehler
Es werden keine Fehlercodes zurückgegeben.

© 2016-2022 Altova GmbH

29.3.2.8.6 DatabaseKind

Eigenschaft: DatabaseKindals SPYDatabaseKind 2022

Beschreibung
Wählen Sie aus, welche Art von Datenbank Zugriff erhält. Der Standardwert ist spyDB_Unspecified(7)und ist in den meisten Fällen ausreichend. Diese Eigenschaft wird nur beim Import von Datenbanken verwendet.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.7 DatabaseSchema

Eigenschaft: DatabaseSchemaals String

Beschreibung
Diese Eigenschaft definiert das Schema, das für den Export in schemafähige Datenbanken verwendet wird. Der Standardwert ist "". Diese Eigenschaft wird nur beim Export in eine Datebank verwendet.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.8 ExcludeKeys

Eigenschaft: ExcludeKeysals Boolean

Beschreibung
Setzen Sie ExcludeKeysauf "true", wenn Sie alle Schlüsselspalten vom Import ausnehmen möchten. Der Standardwert ist "false". Diese Eigenschaft wird nur beim Import von Datenbanken verwendet.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.9 File

Eigenschaft: Fileals String

Beschreibung
Die Eigenschaft Filedefiniert den Pfad für die Datenbank beim Export oder Import. Diese Eigenschaft kann nur in Verbindung mit einer Microsoft Access-Datenbank verwendet werden. Verwenden Sie entweder diese Eigenschaft oder ODBCConnection 1833 oder ADOConnection 1829 um die Datenbank zu referenzieren.

Siehe auch Import und Export 1715.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.10 ForeignKeys

Eigenschaft: ForeignKeysals Boolean

Beschreibung
Gibt an, ob der Sekundärschlüssel-Constraint erstellt wird oder nicht. Diese Eigenschaft wird nur bei Erstellung eines XML-Schemas anhand einer Datenbankstruktur verwendet.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.11 ImportColumnsType

Eigenschaft: ImportColumnsTypeals SPYImportColumnsType 2024

Beschreibung
Definiert, ob Spalteninformationen aus der Datenbank im Schema als Element oder Attribut gespeichert werden. Diese Eigenschaft wird nur bei Erstellung eines XML-Schemas anhand einer Datenbankstruktur verwendet.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.12 IncludeEmptyElements

Eigenschaft: IncludeEmptyElementsals Boolean

Beschreibung
Setzen Sie IncludeEmptyElementsauf "false" wenn alle leeren Elemente ausgenommen werden sollen. Der Standardwert ist "true". Diese Eigenschaft wird nur beim Import von Datenbanken verwendet.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.13 NullReplacement

Eigenschaft: NullReplacementals String

Beschreibung
Diese Eigenschaft enthält den Textwert, der beim Import von leeren Elementen (Nullwerten) verwendet wird. Der Standardwert ist "". Diese Eigenschaft wird nur beim Import von Datenbanken verwendet.

Fehler
Es werden keine Fehlercodes zurückgegeben.

© 2016-2022 Altova GmbH

29.3.2.8.14 NumberDateTimeFormat

Eigenschaft: NumberDateTimeFormat als SPYNumberDateTimeFormat 2026

Beschreibung
Die Eigenschaft NumberDateTimeFormat definiert das Format von Datums- und Uhrzeitwerten. Die Standardeinstellung ist spySystemLocale 2026. Diese Eigenschaft wird nur beim Import von Datenbanken verwendet.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.15 ODBCConnection

Eigenschaft: ODBCConnectionals String

Beschreibung
Die Eigenschaft ODBCConnectionenthält einen ODBC Connection String. Verwenden Sie entweder diese Eigenschaft oder ADOConnection 1829 oder File 1831 um eine Datenbank zu referenzieren.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.16 PrimaryKeys

Eigenschaft: PrimaryKeysals Boolean

Beschreibung
Definiert, ob ein Primärschlüssel-Constraint erstellt wird oder nicht. Die Standardeinstellung ist "true". Diese Eigenschaft wird nur verwendet, wenn ein XML-Schema anhand einer Datenbankstruktur erstellt wird.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.17 SchemaExtensionType

Eigenschaft: SchemaExtensionTypeals SPYSchemaExtensionType 2030

Beschreibung
Definiert den bei der Schemagenerierung verwendeten Schema Extension-Typ. Diese Eigenschaft wird nur verwendet, wenn ein XML-Schema anhand einer Datenbankstruktur erstellt wird.

Siehe auch Erstellen eines XML-Schemas anhand einer Datenbankstruktur 1478.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.18 SchemaFormat

Eigenschaft: SchemaFormatals SPYSchemaFormat 2030

Beschreibung
Definiert das bei der Schemagenerierung verwendete Schemaformat. Diese Eigenschaft wird nur verwendet, wenn ein XML-Schema anhand einer Datenbankstruktur erstellt wird.

Siehe auch Erstellen eines XML-Schemas anhand einer Datenbankstruktur 1478.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.19 SQLSelect

Eigenschaft: SQLSelectals String

Beschreibung
Die SQL-Abfrage für den Import ist in der Eigenschaft SQLSelectgespeichert. Diese Eigenschaft wird nur beim Import von Datenbanken verwendet.Siehe auch Import und Export 1715.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.20 TextFieldLen

Eigenschaft: TextFieldLenals long

Beschreibung
Die Eigenschaft TextFieldLen definiert die Länge der beim Export erstellten Textfelder. Standardwert ist 255. Diese Eigenschaft wird nur beim Export in eine Datenbank verwendet.

Fehler
Es werden keine Fehlercodes zurückgegeben.

29.3.2.8.21 UniqueKeys

Eigenschaft: UniqueKeysals Boolean

Beschreibung
Definiert, ob ein Eindeutiger-Schlüsse-Constraint erstellt wird oder nicht. Die Standardeinstellung ist "true". Diese Eigenschaft wird nur verwendet, wenn ein XML-Schema anhand einer Datenbankstruktur erstellt wird.

Fehler
Es werden keine Fehlercodes zurückgegeben.

© 2016-2022 Altova GmbH

29.3.2.9 Dialogs

Eigenschaften und Methoden

Standard-Automation-Eigenschaften
Application[1835]
Parent[1836]

Verschiedene Dialog-Objekte
CodeGeneratorDlg[1835]
FileSelectionDlg[1836]
SchemaDocumentationDlg[1836]
GenerateSampleXMLDlg[1837]
DTDSchemaGeneratorDlg[1837]
FindInFilesDlg[1837]
WSDLDocumentationDlg[1837]
WSDL20DocumentationDlg[1838]
XBRLDocumentationDlg[1838]

Beschreibung
Über das Objekt "Dialogs" haben Sie Zugriff auf die verschiedenen integrierten Dialogfelder von XMLSpy. Mit Hilfe dieser Dialog-Objekte können die Felder von Benutzerdialogfeldern initialisiert werden, bevor sie dem Benutzer angezeigt werden oder es kann von Ihrem Programm eine vollständige Eingabe durch den Benutzer simuliert werden.

29.3.2.9.1 Application

Eigenschaft: Applicationals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
2300 Das Objekt ist nicht mehr gültig.
2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.9.2 CodeGeneratorDlg

Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Eigenschaft: CodeGeneratorDlgals CodeGeneratorDlg[1822](schreibgeschützt)

Beschreibung
Ruft eine neue Instanz eines Dialogfeldobjekts für die Codegenerierung ab. Sie benötigen dieses Objekt, um die nötigen Parameter an die Codegenerierungsmethoden zu übergeben. Als Anfangswerte werden die zuletzt bei diesem Dialogfeld verwendeten genommen.

Fehler

2300 Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.9.3 FileSelectionDlg

Eigenschaft: FileSelectionDlgals FileSelectionDlg[1883](schreibgeschützt)

Beschreibung
Ruft eine neue Instanz eines Dialogfeldobjekts zur Dateiauswahl auf.

Dialogfeldobjekte zur Dateiauswahl werden mit einigen Events an Sie übergeben, die das Öffnen oder Speichern von Dokumenten und Projekten anzeigen.

Fehler

2300 Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.9.4 Parent

Eigenschaft: Parentals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler

2300 Das Objekt ist nicht mehr gültig.
2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.9.5 SchemaDocumentationDlg

Eigenschaft: SchemaDocumentationDlgals SchemaDocumentationDlg[1906](schreibgeschützt)

Beschreibung
Ruft eine neue Instanz eines Dialogfeldobjekts ab, das die Generierung von Schemadokumentation parametrisiert. Informationen zur Verwendung finden Sei unter Document.GenerateSchemaDocumentation[1856].

Fehler

2300 Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.9.6 GenerateSampleXMLDlg

Eigenschaft: GenerateSampleXMLDlgals GenerateSampleXMLDlg[1896](schreibgeschützt)

Beschreibung
Ruft eine neue Instanz eines Dialogfeldobjekts ab, das die Generierung einer XML-Beispieldatei anhand eines W3C-Schemas oder einer DTD parametrisiert. Informationen zur Verwendung finden Sie unter GenerateSampleXML[1855].

Fehler

2300	Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
2301	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.9.7 DTDSchemaGeneratorDlg

Eigenschaft: DTDSchemaGeneratorDlgals DTDSchemaGeneratorDlg[1874](schreibgeschützt)

Beschreibung
Ruft eine neue Instanz eines Dialogfeldobjekts ab, das die Generierung eines Schemas oder einer DTD parametrisiert. Informationen zur Verwendung finden Sie unter Document.GenerateDTDOrSchemaEx[1855].

Fehler

2300	Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
2301	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.9.8 FindInFilesDlg

Eigenschaft: FindInFilesDlgas FindInFilesDlg[1885](schreibgeschützt)

Beschreibung
Ruft eine neue Instanz eines Dialogfeldobjekts ab, das die Suche oder Ersetzung von Stings in Dateien parametrisiert. Informationen zur Verwendung finden Sie unter Application.FindInFiles[1752].

Fehler

2300	Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
2301	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.9.9 WSDLDocumentationDlg

Eigenschaft: WSDLDocumentationDlgals WSDLDocumentationDlg[1934](schreibgeschützt)

Beschreibung
Ruft eine neue Instanz eines Dialogfeldobjekts ab, das die Generierung von WSDL-Dokumentation parametrisiert. Informationen zur Verwendung finden Sie unter Document.GenerateWSDLDocumentation[1857].

Fehler

2300 Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.9.10 WSDL20DocumentationDlg

Eigenschaft: WSDL20DocumentationDlgals WSDL20DocumentationDlg[1946](schreibgeschützt)

Beschreibung
Ruft eine neue Instanz eines Dialogfeldobjekts ab, das die Generierung von WSDL 2.0-Dokumenten parametrisiert. Informationen zur Verwendung finden Sie unter Document.GenerateWSDL20Documentation[1856].

Fehler

2300 Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.9.11 XBRLDocumentationDlg

Eigenschaft: XBRLDocumentationDlg als XBRLDocumentationDlg[1958] (schreibgeschützt)

Beschreibung
Ruft eine neue Instanz eines Dialogfeldobjekts ab, das die Generierung von XBRL-Dokumenten parametrisiert. Informationen zur Verwendung finden Sie unter Document.GenerateXBRLDocumentation[1856].

Fehler

2300 Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.10 Document

Eigenschaften und Methoden

Standard-Automation-Eigenschaften
Application[1843]
Parent[1862]

Verschiedene Dokumenteigenschaften und Methoden
SetActiveDocument[1865]
Encoding[1850]
SetEncoding (obsolete)[1865]
Suggestions[1867]

© 2016-2022 Altova GmbH

XML-Validierung
IsWellFormed 1862
IsValid 1859
SetExternalIsValid 1866

Dokumentkonvertierung und -transformation
AssignDTD 1843
AssignSchema 1844
AssignXSL 1844
AssignXSLFO 1844
ConvertDTDOrSchema 1846
ConvertDTDOrSchemaEx 1847
GenerateDTDOrSchema 1854
GenerateDTDOrSchemaEx 1855
FlattenDTDOrSchema 1853
CreateSchemaDiagram 1848
ExecuteXQuery 1851
TransformXSL 1868
TransformXSLEx 1868
TransformXSLFO 1868
GenerateProgramCode 1855 (nur Enterprise Edition)
GenerateSchemaDocumentation 1856
GenerateSampleXML 1855
GenerateWSDL20Documentation 1856
GenerateWSDLDocumentation 1857
GenerateXBRLDocumentation 1857
ConvertToWSDL20 1847

Dokumentexport
GetExportElementList 1858
ExportToText 1852
ExportToDatabase 1851
CreateDBStructureFromXMLSchema 1848
GetDBStructureList 1858

Dateispeicherung und - benennung
FullName 1854
Name 1862
Path 1863
GetPathName (obsolete) 1858
SetPathName (obsolete) 1866
Title 1868
IsModified 1859
Saved 1864
SaveAs 1864
Save 1863
SaveInString 1864
SaveToURL 1865
Close 1846

Aufruf einer Ansicht
CurrentViewMode 1849

SwitchViewMode 1867
AuthenticView 1845
GridView 1859
DocEditView (obsolete) 1849

Zugriff auf XMLData
RootElement 1863
DataRoot 1849
CreateChild 1848
UpdateViews 1869
StartChanges 1867
EndChanges 1850
UpdateXMLData 1869

Beschreibung
Document-Objekte repräsentieren in XMLSpy geöffnete XML-Dokumente.

Verwenden Sie eine der folgenden Eigenschaften, um Dokumente aufzurufen, die bereits in XMLSpy geöffnet sind:
Application.ActiveDocument 1749
Application.Documents 1752

Verwenden Sie eine der folgenden Methoden, um ein neues Dokument in XMLSpy zu öffnen:
Documents.OpenFile 1872
Documents.OpenURL 1873
Documents.OpenURLDialog 1873
Documents.NewFile 1871
Documents.NewFileFromText 1872
SpyProjectItem.Open 1922
Application.ImportFromDatabase 1757
Application.ImportFromSchema 1758
Application.ImportFromText 1758
Application.ImportFromWord 1759
Document.ConvertDTDOrSchema 1846
Document.GenerateDTDOrSchema 1854

29.3.2.10.1 Events

29.3.2.10.1.1 OnBeforeSaveDocument

Event: OnBeforeSaveDocument(*objDocument* als Document 1838, *objDialog* als FileSelectionDlg 1883)

Skripting-Umgebung - VBScript:
Function On_BeforeSaveDocument(*objDocument*, *objDialog*)
End Function

' old handler - now obsolete
' return string to save to new file name
' return empty string to cancel save operation
' return nothing to save to original name

© 2016-2022 Altova GmbH

```
Function On_SaveDocument(objDocument, strFilePath)
End Function
```

Skripting-Umgebung - JScript:

```
function On_BeforeSaveDocument(objDocument, objDialog)
{
}

// old handler - now obsolete
// return string to save to new file name
// return empty string to cancel save operation
// return nothing to save to original name
function On_SaveDocument(objDocument, strFilePath)
{
}
```

IDE Plugin:

```
IXMLSpyPlugIn.OnEvent(27, ...)  // nEventId = 27
```

Beschreibung

Dieses Event wird bei jedem Versuch, ein Dokument zu speichern ausgelöst. Das Dialogfeldobjekt zur Dateiauswahl wird mit dem für die Dokumentdatei gewählten Namen initialisiert. Sie können diese Auswahl ändern. Um mit dem Speichern des Dokuments fortzufahren, belassen Sie die Eigenschaft FileSelectionDlg.DialogAction[1884] von *io_objDialog* auf ihrem Standardwert spyDialogOK[2023]. Um das Speichern des Dokuments abzubrechen, setzen Sie diese Eigenschaft auf spyDialogCancel[2023].

29.3.2.10.1.2 OnBeforeCloseDocument

Event: OnBeforeCloseDocument(*objDocument* als Document[1838])als Boolean

Skripting-Umgebung - VBScript:

```
Function On_BeforeCloseDocument(objDocument)
        ' On_BeforeCloseDocument = False  ' to prohibit closing of document
End Function
```

Skripting-Umgebung - JScript:

```
function On_BeforeCloseDocument(objDocument)
{
        // return false;  /* to prohibit closing of document */
}
```

IDE Plugin:

```
IXMLSpyPlugIn.OnEvent(28, ...)  // nEventId = 28
```

Beschreibung

Dieses Event wird bei jedem Versuch ein Dokument zu schließen ausgelöst. Um das Schließen des Dokuments zu verhindern, muss der Wert "false" zurückgegeben. werden.

29.3.2.10.1.3 OnBeforeValidate

Event: OnBeforeValidate(*objDocument* als Document[1838], *bOnLoading* als Boolean, *bOnCommand* als Boolean)als Boolean

Skripting-Umgebung - VBScript:
```
Function On_BeforeValidate(objDocument, bOnLoading, bOnCommand)
        On_BeforeValidate = bCancelDefaultValidation     'set by the script if necessary
End Function
```

Skripting-Umgebung - JScript:
```
function On_BeforeValidate(objDocument, bOnLoading, bOnCommand)
{
        returnbCancelDefaultValidation    //set by the script if necessary
}
```

IDE Plugin:
IXMLSpyPlugIn.OnEvent(32, ...) // nEventId = 32

Beschreibung
Dieses Event wird ausgelöst, bevor das Dokument validiert wird. Die Standardvalidierung kann durch Rückgabe von "false" vom Event Handler unterdrückt werden. In diesem Fall sollte das Script auch mit Hilfe der Methode SetExternalIsValid[1866] das Validierungsergebnis definieren.

bOnLoadingist "true", wenn das Event bei der Anfangsvalidierung beim Laden des Dokuments ausgelöst wird.

bOnCommandist "true", wenn der Benutzer in der Symbolleiste oder im Menü den Befehl "Validieren" ausgewählt hat.

Verfügbar mit Typbibliothek Version 1.5

29.3.2.10.1.4 OnCloseDocument

Event: OnCloseDocument(*objDocument* als Document[1838])

Skripting-Umgebung - VBScript:
```
Function On_Close Document(objDocument)
End Function
```

Skripting-Umgebung - JScript:
```
function On_Close Document(objDocument)
{
}
```

IDE Plugin:
IXMLSpyPlugIn.OnEvent(8, ...) // nEventId = 8

Beschreibung

© 2016-2022 Altova GmbH

Dieses Event wird ausgelöst, wenn ein Dokument geschlossen wird. Ändern Sie das Dokument nicht von diesem Event aus.

29.3.2.10.1.5 OnViewActivation

Event: OnViewActivation(*objDocument* als Document[1838], *eViewMode* als SPYViewModes[2032], *bActivated* als Boolean)

Skripting-Umgebung - VBScript:
Function On_ViewActivation(*objDocument*, *eViewMode*, *bActivated*)
End Function

Skripting-Umgebung - JScript:
function On_ViewActivation(*objDocument*, *eViewMode*, *bActivated*)
{
}

IDE Plugin:
IXMLSpyPlugIn.OnEvent(29, ...) // nEventId = 29

Beschreibung
Dieses Event wird immer dann ausgelöst, wenn eine Ansicht eines Dokuments eingeblendet (d.h. zur aktiven Ansicht wird) oder ausgeblendet wird (d.h. eine andere Ansicht wird aktiv oder das Dokument wird geschlossen). Das erste Ansichtsaktivierungsevent nach dem Öffnen eines Dokuments kann jedoch nicht empfangen werden, da es kein document-Objekt gibt, von dem das Event abgerufen werden kann. Verwenden Sie stattdessen das Event Application.OnDocumentOpened[1748].

29.3.2.10.2 Application

Eigenschaft: Applicationals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
1400 Das Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.10.3 AssignDTD

Methode: AssignDTD(*strDTDFile*als String, *bDialog* als Boolean)

Beschreibung
Die Methode platziert eine Referenz auf die DTD-Datei "strDTDFile" in das Dokument. Beachten Sie, dass kein Fehler auftritt, wenn die Datei nicht existiert oder kein Zugriff darauf besteht. Wenn bDialog "true" ist, zeigt XMLSpy ein Dialogfeld zum Definieren der Datei an.

Fehler
1400 Das Objekt ist nicht mehr gültig.
1409 Dem Dokument darf keine DTD zugewiesen werden.

29.3.2.10.4 AssignSchema

Methode: AssignSchema (*strSchemaFile* als String, *bDialog* als Boolean)

Beschreibung
Die Methode platziert eine Referenz auf die Schemadatei "strSchemaFile" in das Dokument. Beachten Sie, dass kein Fehler auftritt, wenn die Datei nicht existiert oder kein Zugriff darauf besteht. Wenn bDialog "true" ist, zeigt XMLSpy ein Dialogfeld zum Definieren der Datei an.

Fehler
1400 Das Objekt ist nicht mehr gültig.
1409 Dem Dokument darf keine Schemadatei zugewiesen werden.

29.3.2.10.5 AssignXSL

Methode: AssignXSL (*strXSLFile* als String, *bDialog* als Boolean)

Beschreibung
Die Methode platziert eine Referenz auf die XLS-Datei "strXSLFile" in das Dokument. Beachten Sie, dass kein Fehler auftritt, wenn die Datei nicht existiert oder kein Zugriff darauf besteht. Wenn bDialog "true" ist, zeigt XMLSpy ein Dialogfeld zum Definieren der Datei an.

Fehler
1400 Das Objekt ist nicht mehr gültig.
1409 Dem Dokument darf keine XSL-Datei zugewiesen werden

29.3.2.10.6 AssignXSLFO

Methode: AssignXSLFO (*strXSLFOFile* als String, *bDialog* als Boolean)

Beschreibung
Die Methode platziert eine Referenz auf die XSLFO-Datei "strXSLFile" in das Dokument. Beachten Sie, dass kein Fehler auftritt, wenn die Datei nicht existiert oder kein Zugriff darauf besteht. Wenn bDialog "true" ist, zeigt XMLSpy ein Dialogfeld zum Definieren der Datei an.

Fehler
1400 Das Objekt ist nicht mehr gültig.
1409 Dem Dokument darf keine XSL-Datei zugewiesen werden

© 2016-2022 Altova GmbH

29.3.2.10.7 AsXMLString

Eigenschaft: AsXMLStringals String

Beschreibung
Diese Eigenschaft dient zum Abrufen oder Definieren des Inhalts des Dokuments.

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1404 Das XMLData-Objekt kann nicht erstellt werden.
1407 Der Ansichtsmodus kann nicht gewechselt werden.

29.3.2.10.8 AuthenticView

Methode: AuthenticView als AuthenticView[1803] (schreibgeschützt)

Beschreibung
Gibt ein Objekt zurück, über das Sie Zugriff auf die Eigenschaften und Methoden haben, die für die Authentic-Ansicht spezifisch sind. Das zurückgegebene Objekt ist nur gültig, wenn das aktuelle Dokument im Authentic-Ansichtsmodus geöffnet wird. Die Lebenszeit eines Objekts endet mit dem nächsten Ansichtswechsel. Jeder Versuch, Objekte oder eines ihrer Children danach aufrufen, hat einen Fehler zur Folge, der angibt, dass das Objekt ungültig ist.

AuthenticView[1803] und DocEditView[1996] bieten beide Automationszugriff auf den Authentic-Ansichtsmodus von XMLSpy. Dass sich die beiden Funktionen überlappen, ist beabsichtigt. Eine zukünftige Version von AuthenticView wird alle Funktionalitäten von DocEditView[1996] und seinen Unterobjekten enthalten, sodass DocEditView[1996] nicht mehr benötigt wird.

Fehler
1400 Das Objekt ist nicht mehr gültig.
1417 Das Dokument muss im Authentic-Ansichtsmodus geöffnet sein.

Beispiele

```
' ----------------------------------------
' Scripting environment - VBScript
' secure access to authentic view object
' ----------------------------------------
DimobjDocument
SetobjDocument = Application.ActiveDocument
If(NotobjDocument Is Nothing) Then
	' we have an active document, now check for view mode
	If(objDocument.CurrentViewMode <> spyViewAuthentic) Then
		If(NotobjDocument.SwitchViewMode (spyViewAuthentic)) Then
			MsgBox "Active document does not support authentic view mode"
		Else
			' now it is safe to access the authentic view object
			DimobjAuthenticView
			SetobjAuthenticView = objDocument.AuthenticView
			' now use the authentic view object
```

```
            EndIf
        EndIf
Else
        MsgBox "No document is open"
EndIf
```

29.3.2.10.9 Close

Methode: Close (*bDiscardChanges* als Boolean)

Beschreibung
Rufen Sie diese Methode auf, um ein Dokument zu schließen. Wenn bDiscardChanges"true" ist und das Dokument geändert wurde, wird das Dokument geschlossen aber nicht gespeichert.

Fehler
1400 Das Objekt ist nicht mehr gültig.
1401 Das Dokument muss zuerst gespeichert werden.

29.3.2.10.10 ConvertDTDOrSchema

Methode: ConvertDTDOrSchema (*nFormat* als SPYDTDSchemaFormat[2023], *nFrequentElements* als SPYFrequentElements[2024])

Parameter
nFormat
Setzt das Schemaausgabeformat auf DTD oder W3C.

nFrequentElements
Erstellt komplexe Elemente als Elemente oder complexTypes.

Beschreibung
ConvertDTDOrSchemanimmt das vorhandene Schemaformat und konvertiert es in ein anderes Format. Zur Feinabstimmung einer DTD/XSD-Konvertierung verwenden Sie ConvertDTDOrSchemaEx[1847].

Fehler
1400 Das Objekt ist nicht mehr gültig.
1412 Fehler bei der Konvertierung. Im Fall einer Konvertierung einer DTD in eine DTD oder einer XSD- in eine XSD-Datei, werden die folgenden Fehler zurückgegeben: *Die Konvertierung einer DTD in eine andere DTD wird nicht unterstützt. Verwenden Sie stattdessen die Funktion "DTD oder Schema nivellieren"!* und *Die Konvertierung eines Schemas in ein anderes Schema wird nicht unterstützt. Verwenden Sie stattdessen die Funktion "DTD oder Schema nivellieren"!*

© 2016-2022 Altova GmbH

29.3.2.10.11 ConvertDTDOrSchemaEx

Methode: ConvertDTDOrSchemaEx (*nFormat* als SPYDTDSchemaFormat[2023], *nFrequentElements* als SPYFrequentElements[2024], sOutputPath als String, nOutputPathDialogAction als SPYDialogAction[2023])

Parameters
nFormat
Setzt das Schemaausgabeformat auf DTD oder W3C.

nFrequentElements
Erstellt komplexe Elemente als Elemente oder complexTypes.

sOutputPath
Der Dateipfad für die neu generierte Datei.

nOutputPathDialogAction
Definiert die Dialogfeldinteraktion für diesen Aufruf.

Beschreibung
ConvertDTDOrSchemaExnimmt ein vorhandenes Schemaformat und konvertiert es in ein anderes Format.

Fehler

1400	Das Objekt ist nicht mehr gültig.
1412	Fehler bei der Konvertierung. Im Fall einer Konvertierung einer DTD in eine DTD oder einer XSD- in eine XSD-Datei, werden die folgenden Fehler zurückgegeben: *Die Konvertierung einer DTD in eine andere DTD wird nicht unterstützt. Verwenden Sie stattdessen die Funktion "DTD oder Schema nivellieren"!* und *Die Konvertierung eines Schemas in ein anderes Schema wird nicht unterstützt. Verwenden Sie stattdessen die Funktion "DTD oder Schema nivellieren"!*

29.3.2.10.12 ConvertToWSDL20

Methode: ConvertToWSDL20 (*sFilePath* als String, *bShowDialogs* als Boolean)

Parameter
sFilePath
Dieser Parameter definiert den Dateinamen der konvertierten WSDL-Datei. Falls auch Include-Dateien aus WSDL-Quelldateien konvertiert werden müssen, wird nur der Verzeichnisteil des angegebenen Pfads verwendet und die Dateinamen werden automatisch generiert.

bShowDialogs
Definiert, ob Dialogfelder zur Auswahl von Dateien/Ordnern angezeigt werden sollen.

Beschreibung
Konvertiert das WSDL 1.1-Dokument in eine WSDL 2.0-Datei. Konvertiert auch alle referenzierten WSDL-Dateien, die von diesem Dokument aus referenziert werden. Beachten Sie, dass diese Funktionalität auf die WSDL-Ansicht beschränkt ist. SiehDocument.CurrentViewMode[1849]. und SPYViewModes[2032].

Fehler

1400 Das Dokument-Objekt ist nicht mehr gültig.
1407 Es wurden ungültige Parameter übergeben oder ein leerer Dateiname wurde als Ausgabeziel definiert.
1417 Das Dokument ist nicht in der WSDL-Ansicht geöffnet. Vielleicht ist es keine '.wsdl' Datei.
1421 Funktion steht in dieser Edition nicht zur Verfügung.
1433 Konvertierung von WSDL 1.1 in WSDL 2.0 ist fehlgeschlagen.

29.3.2.10.13 CreateChild

Methode: CreateChild (*nKind* als SPYXMLDataKind[2033]) als XMLData[1970]

Rückgabewert
Die Methode gibt das neue XMLDataObjekt zurück.

Beschreibung
Um ein neues XMLDataObjekt zu erstellen, verwenden Sie die Methdoe CreateChild().

Fehler
1400 Das Objekt ist nicht mehr gültig.
1404 Das XMLData-Objekt kann nicht erstellt werden.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.10.14 CreateDBStructureFromXMLSchema

Methode: CreateDBStructureFromXMLSchema (*pDatabase* als DatabaseConnection[1828], *pTables* als ElementList[1878], *bDropTableWithExistingName* als Boolean) als String

Beschreibung
CreateDBStructureFromXMLSchemaexportiert die angegebenen Tabellen in die angegebene Datenbank.Die Funktion gibt die SQL-Anweisungen zurück, die zur Durchführung der Änderungen erforderlich waren.

Siehe auch GetDBStructureList[1858].

Fehler
1429 Datenbankauswahl fehlt.
1430 Der Dokumentexport ist fehlgeschlagen.

29.3.2.10.15 CreateSchemaDiagram

Method: CreateSchemaDiagram (*nKind* as SPYSchemaDefKind[2029], *strName* as String, *strFile* as String)

Rückgabewert

© 2016-2022 Altova GmbH

Keiner.

Beschreibung
Die Methode erstellt ein Diagramm des Schematyps strName von der Art nKind und speichert die Ausgabedatei in strFile. Beachten Sie, dass diese Funktionalität auf die Schema-Ansicht beschränkt ist. Siehe Document.CurrentViewMode[1849]. und SPYViewModes[2032].

Fehler
1400 Das Objekt ist nicht mehr gültig.
1414 Das Diagramm konnte nicht gespeichert werden.
1415 Es wurde ein ungültiger Schemadefinitionstyp angegeben.

29.3.2.10.16 CurrentViewMode

Methode: CurrentViewMode als SPYViewModes[2032]

Beschreibung
Die Eigenschaft enthält den aktuellen Ansichtsmodus des Dokuments. Siehe auch Document.SwitchViewMode[1867].

Fehler
1400 Das Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.10.17 DataRoot

Eigenschaft: DataRoot als XMLData[1970] (schreibgeschützt)

Beschreibung
Über diese Methode haben Sie Zugriff auf das erste XMLData-Objekt des Dokuments vom Typ *spyXMLDataElement*. Dies ist normalerweise das Root-Element für alle Dokumentinhaltsdaten. Eine Beschreibung, wie Sie das Root-Element des gesamten Dokuments inkl. XML-Prologdaten abrufen, finden Sie unter XMLSpyDocument.RootElement[1863]. Wenn CurrentViewMode[1849] nicht *spyViewGrid* oder *spyViewAuthentic* ist, ist möglicherweise ein UpdateXMLData[1869] erforderlich, um Zugriff auf die neuesten XMLData[1970] zu erhalten.

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.10.18 DocEditView

Methode: DocEditView als DocEditView

Beschreibung
Enthält eine Referenz auf das aktuelle AuthenticObjekt.

Fehler

1400 Das Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
1417 Das Dokument muss im Authentic-Ansichtsmodus geöffnet sein.

29.3.2.10.19 Encoding

Eigenschaft: Encoding als String

Beschreibung
Über diese Eigenschaft haben Sie Zugriff auf den Encoding-Wert des Dokuments. Diese Eigenschaft kann nur aufgerufen werden, wenn das Dokument in *spyViewGrid*, *spyViewText* oder *spyViewAuthentic* geöffnet ist. Um den aktuellen Ansichtsmodus eines Dokuments abzurufen, siehe Beschreibung CurrentViewMode[1849].

Mit dieser Eigenschaft wird die Methode SetEncoding[1865] nicht mehr benötigt.

Mögliche Werte sind zum Beispiel:

8859-1,
8859-2,
ASCII, ISO-646,
850,
1252,
1255,
SHIFT-JIS, MS-KANJI,
BIG5, FIVE,
UTF-7,
UTF-8,
UTF-16

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
1416 Die Operation wird im aktuellen Ansichtsmodus nicht unterstützt.

29.3.2.10.20 EndChanges

Methode: EndChanges()

Beschreibung
Verwenden Sie die Methode EndChangesum alle Änderungen seit dem letzten Aufruf von Document.StartChanges[1867] anzuzeigen.

Fehler
1400 Das Objekt ist nicht mehr gültig.

© 2016-2022 Altova GmbH

29.3.2.10.21 ExecuteXQuery

Methode: ExecuteXQuery (*strXMLFileName* als String)

Beschreibung
Führt die im Dokument des Document-Objekts enthaltenen XQuery-Anweisungen aus. Je nachdem, welche Dateierweiterung das Dokument hat, wird entweder eine XQuery oder ein XQuery Update ausgeführt. Verwendet die im Argument definierte XML-Datei als XML-Zieldokument, das das XQuery-Dokument verarbeitet.

- Wenn das Dokument eine der XQuery-Dateierweiterungen hat, die im Dialogfeld "Optionen" von XMLSpydefiniert sind, so wird eine XQuery-Ausführung durchgeführt. Standardmäßig sind in XMLSpy `.xq`, `.xql`und `.xquery`als XQuery-Dateierweiterungen definiert.
- Wenn das Dokument eine XQuery Update-Dateierweiterung hat, die im Dialogfeld "Optionen" von XMLSpydefiniert ist, so wird eine XQuery Update-Aktion durchgeführt. Standardmäßig ist in XMLSpy `.xqu`als XQuery Update-Dateierweiterung definiert.

Wenn Ihr XQuery-Script keine XML-Quelle verwendet, setzen Sie den Parameter strXMLFileNameauf einen leeren String.

Fehler

1400	Das document-Objekt ist nicht mehr gültig.
1423	XQuery-Transformationsfehler.
1424	Es konnten nicht alle für die Operation benötigten Dateien geladen werden. Höchstwahrscheinlich existiert die in strXMLFileNamedefinierte Datei nicht oder ist ungültig.

29.3.2.10.22 ExportToDatabase

Methode: ExportToDatabase (*pFromChild* als XMLData[1970], *pExportSettings* als ExportSettings[1880], *pDatabase* als DatabaseConnection[1828])

Beschreibung
ExportToDatabaseexportiert das XML-Dokument beginnend mit dem Element pFromChild. Der Parameter pExportSettingsdefiniert das Exportverhalten (siehe Application.GetExportSettings[1754]). Der Parameter pDatabasedefiniert das Ziel des Exports (siehe Application.GetDatabaseSettings[1753]). `UpdateXMLData()`[1869] wird eventuell indirekt benötigt, da Sie `XMLData`[1970]als Parameter an diese Funktion übergeben müssen.

Fehler

1400	Das Objekt ist nicht mehr gültig.
1407	Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.
1416	Fehler beim Export.
1429	Datenbankauswahl fehlt.
1430	Dokumentexport fehlgeschlagen.

Beispiel

```
DimobjDoc AsDocument
```

```
SetobjDoc = objSpy.ActiveDocument

'set the behaviour of the export with ExportSettings
DimobjExpSettings AsExportSettings
SetobjExpSettings = objSpy.GetExportSettings

'set the destination with DatabaseConnection
DimobjDB AsDatabaseConnection
SetobjDB = objSpy.GetDatabaseSettings

objDB.CreateMissingTables = True
objDB.CreateNew = True
objDB.File = "C:\Export.mdb"

objDoc.ExportToDatabase objDoc.RootElement, objExpSettings, objDB
IfErr.Number <> 0 Then
        a = MsgBox("Error: " & (Err.Number - vbObjectError) & Chr(13) &
                        "Description: " & Err.Description)
EndIf
```

29.3.2.10.23 ExportToText

Methode: ExportToText (*pFromChild* als XMLData[1970], *pExportSettings* als ExportSettings[1880], *pTextSettings* als TextImportExportSettings[1925])

Beschreibung
ExportToTextexportiert Tabellendaten aus dem Dokument beginnend mit pFromChildin eine oder mehrere Textdateien. Die Spalten der erzeugten Tabellen werden in alphabetischer Reihenfolge nach den Namen der Tabellenspalten generiert. Verwenden Sie GetExportElementList[1858],um zu sehen, welche Daten exportiert werden. Der Parameter pExportSettingsdefiniert die näheren Einzelheiten für den Export. Setzen Sie die Eigenschaft ExportSettings.ElementList[1881]auf die - möglicherweise geänderte - Liste, die von GetExportElementList[1858]zurückgegeben wird, damit nicht alle enthaltenen Tabellen exportiert werden. Der Parameter pTextSettingsdefiniert die Optionen für den Textexport und -import. Sie müssen die Eigenschaft TextImportExportSettings.DestinationFolder[1926]setzen, bevor Sie ExportToText aufrufen.`UpdateXMLData()`[1869] wird eventuell indirekt benötigt, da Sie `XMLData`[1970]als Parameter an diese Funktion übergeben müssen.

Siehe auch Import and export of data[1715].

Fehler

1400	Das Objekt ist nicht mehr gültig.
1407	Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.
1416	Fehler beim Export..
1430	Dokumentexport fehlgeschlagen.

Beispiel

```
' ----------------------------------------------------
' VBA client code fragment - export document to text files
' ----------------------------------------------------
```


© 2016-2022 Altova GmbH

```
Dim objDoc As Document
Set objDoc = objSpy.ActiveDocument

Dim objExpSettings As ExportSettings
Set objExpSettings = objSpy.GetExportSettings
objExpSettings.ElementList = objDoc.GetExportElementList(
                                                    objDoc.RootElement,
                                                    objExpSettings)

Dim objTextExp As TextImportExportSettings
Set objTextExp = objSpy.GetTextImportExportSettings
objTextExp.HeaderRow = True
objTextExp.DestinationFolder = "C:\Exports"

On Error Resume Next
objDoc.ExportToText objDoc.RootElement, objExpSettings, objTextExp

If Err.Number <> 0 Then
        a = MsgBox("Error: " & (Err.Number - vbObjectError) & Chr(13) & "Description: "
        & Err.Description)
End If
```

29.3.2.10.24 FlattenDTDOrSchema

Methode: FlattenDTDOrSchema (sOutputPath als String, nOutputPathDialogAction als SPYDialogAction[2023])

Parameter

`sOutputPath`
Der Dateipfad der neu generierten Datei.

`nOutputPathDialogAction`
Definiert die Dialogfeldinteraktion für diesen Aufruf.

Beschreibung

`FlattenDTDOrSchema`generiert anhand einer vorhandenen DTD/eines vorhandenen Schemas eine nivellierte, flache Datei und speichert die generierte Datei unter dem angegebenen Pfad. Bei DTDs werden bei der Nivellierung Parameter Entities entfernt und anhand einer Sammlung von Modulen wird eine einzige DTD erzeugt; Mit `IGNORE` markierte Abschnitte werden unterdrückt und nicht verwendete Parameter Entities werden gelöscht. Bei Nivellierung eines XML-Schemas (i) werden die Komponenten aller inkludierten Schemas als globale Komponenten des aktiven Schemas hinzugefügt (ii) und die inkludierten Schemas gelöscht.

Errors

1400 Das Objekt ist nicht mehr gültig.
1412 Fehler bei der Konvertierung.

29.3.2.10.25 FullName

Eigenschaft: FullName als String

Beschreibung
Mit Hilfe dieser Eigenschaft kann der vollständige Dateiname - einschließlich des Pfads - unter dem das Dokument gespeichert wird, abgerufen oder definiert werden. Die Gültigkeit des Namens wird vor dem nächsten Speichervorgang nicht überprüft.

Diese Eigenschaft macht die Methoden GetPathName[1858] und SetPathName[1866] überflüssig.

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1402 Es wurde ein leerer String als vollständiger Dateiname definiert.

29.3.2.10.26 GenerateDTDOrSchema

Methode: GenerateDTDOrSchema (*nFormat* als SPYDTDSchemaFormat[2023], *nValuesList* als integer, *nDetection* als SPYTypeDetection[2031], *nFrequentElements* als SPYFrequentElements[2024])

Parameter
nFormat
Setzt das Schemaausgabeformat auf DTD oder W3C.

nValuesList
Generiert nur diese Anzahl an enumerations-facets pro Typ. Setzen Sie den Wert auf -1 um die Anzahl nicht zu begrenzen.

nDetection
Definiert die Genauigkeit der Ermittlung eines simpleType.

nFrequentElements
Sollen die Typen für alle Elemente als global definiert werdenl? Verwenden Sie diesen Wert *spyGlobalComplexType* um sie als globale Elemente zu definiert. Verwenden Sie andernfalls den Wert *spyGlobalElements*.

Beschreibung
Mit Hilfe dieser Methode können Sie automatisch eine DTD oder ein Schema für das aktuelle XML-Dokument generieren.
Informationen zur Feinabstimmung der DTD / Schema-Generierung finden Sie unter GenerateDTDOrSchemaEx[1855].
Beachten Sie, dass diese Funktionalität nicht nur in der ZIP-Ansicht verfügbar ist. Siehe Document.CurrentViewMode[1849]. und SPYViewModes[2032].

Fehler
1400 Das Objekt ist nicht mehr gültig.
1407 Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.10.27 GenerateDTDOrSchemaEx

Methode: GenerateDTDOrSchemaEx (*objDlg* als DTDSchemaGeneratorDlg[1874]) als Document[1838]

Beschreibung
Verwenden Sie diese Methode, um automatisch eine DTD oder ein Schema für das aktuelle XML-Dokument zu generieren. Mit Hilfe eines DTDSchemaGeneratorDlg[1874]Objekts werden die Informationen an den Schema-/DTD-Generator übergeben. Der Generierungsvorgang kann so konfiguriert werden, dass der Benutzer entweder Eingaben machen kann oder dass der Prozess ohne weitere Eingaben durch den Benutzer ausgeführt wird.
Beachten Sie, dass diese Funktionalität nicht nur in der ZIP-Ansicht verfügbar ist. Siehe Document.CurrentViewMode[1849]. und SPYViewModes[2032].

Fehler

1400	Das Objekt ist nicht mehr gültig.
1407	Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.

29.3.2.10.28 GenerateProgramCode

Methode: GenerateProgramCode (*objDlg* als CodeGeneratorDlg[1822])
Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung
Generiert Java, C++ oder C#-Klassendateien anhand der XML-Schemadefinitionen in Ihrem Dokument. Die Daten werden mit Hilfe eines CodeGeneratorDlg[1822]Objekts an den Code Generator übergeben. Der Generierungsvorgang kann so konfiguriert werden, dass der Benutzer entweder Eingaben machen kann oder dass der Prozess ohne weitere Eingaben durch den Benutzer ausgeführt wird.

Fehler

1400	Das document-Objekt ist nicht mehr gültig.
1407	Es wurde ein leerer Dateiname angegeben.
1421	Diese Funktion steht in dieser Edition nicht zur Verfügung.

29.3.2.10.29 GenerateSampleXML

Methode: GenerateSampleXML (*objDlg* als GenerateSampleXMLDlg[1896]) als Document[1838]

Beschreibung
Generiert eine XML-Beispieldatei, wenn das Dokument ein Schema oder eine DTD ist. Verwenden Sie Dialogs.GenerateSampleXMLDlg[1837]um eine initialisierte Gruppe von Optionen aufzurufen.

Verfügbar mit Typbibliothek Version 1.5

Fehler

1400 Das document-Objekt ist nicht mehr gültig.

29.3.2.10.30 GenerateSchemaDocumentation

Methode: GenerateSchemaDocumentation (*objDlg* als SchemaDocumentationDlg[1906])

Beschreibung
Generiert Dokumentation für eine Schemadefinitionsdatei im HTML-, MS-Word- oder RTF-Format. Der Parameter "objDlg" dient zum Parametrisieren des Generierungsvorgangs. Verwenden Sie Dialogs.SchemaDocumentationDlg[1836] um eine initialisierte Gruppe von Optionen aufzurufen. Sie müssen zumindest die Eigenschaft SchemaDocumentationDlg.OutputFile[1913] definieren, bevor Sie mit der Generierung beginnen können. Beachten Sie, dass diese Funktion nur für die Schema-Ansicht zur Verfügung steht. Siehe Document.CurrentViewMode[1849] und SPYViewModes[2032].

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1407 Es wurden entweder ungültige Parameter übergeben oder es wurde ein leere Dateiname für die Ausgabe definiert.
1417 Das Dokument ist nicht in der Schemaansicht geöffnet. Vielleicht handelt es sich um keine '.xsd' Datei.
1421 Die Funktion steht in dieser Edition nicht zur Verfügung.
1422 Fehler bei der Generierung

29.3.2.10.31 GenerateWSDL20Documentation

Methode: GenerateWSDL2Documentation (*objDlg* als WSDL20DocumentationDlg[1946])

Beschreibung
Generiert Dokumentation für eine WSDL-Definitionsdatei im HTML-, MS-Word- oder RTF-Format. Der Parameter "objDlg" dient zum Parametrisieren des Generierungsvorgangs. Verwenden Sie Dialogs.WSDL20DocumentationDlg[1838] um eine initialisierte Gruppe von Optionen aufzurufen. Sie müssen zumindest die Eigenschaft WSDL20DocumentationDlg.OutputFile[1952] definieren, bevor Sie mit der Generierung beginnen können. Beachten Sie, dass diese Funktionalität nur für die WSDL-Ansicht zur Verfügung steht.Siehe Document.CurrentViewMode[1849] und SPYViewModes[2032].

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1407 Es wurden entweder ungültige Parameter übergeben oder es wurde ein leerer Dateiname für die Ausgabe definiert.
1417 Das Dokument ist nicht in der Schemaansicht geöffnet. Vielleicht handelt es sich um keine '.xsd' Datei.
1421 Die Funktion steht in dieser Edition nicht zur Verfügung.
1422 Fehler bei der Generierung

© 2016-2022 Altova GmbH

29.3.2.10.32 GenerateWSDLDocumentation

Methode: GenerateWSDLDocumentation (*objDlg* als WSDLDocumentationDlg[1934])

Beschreibung
Generiert Dokumentation für eine WSDL-Definitionsdatei im HTML-, MS-Word- oder RTF-Format. Der Parameter "objDlg" dient zum Parametrisieren des Generierungsvorgangs. Verwenden Sie Dialogs.SchemaDocumentationDlg[1836] um eine initialisierte Gruppe von Optionen aufzurufen. Sie müssen zumindest die Eigenschaft SchemaDocumentationDlg.OutputFile[1913] definieren, bevor Sie mit der Generierung beginnen können. Beachten Sie, dass diese Funktionalität nur für die WSDL-Ansicht zur Verfügung steht.Siehe Document.CurrentViewMode[1849] und SPYViewModes[2032].

Fehler

1400	Das document-Objekt ist nicht mehr gültig.
1407	Es wurden entweder ungültige Parameter übergeben oder es wurde ein leerer Dateiname für die Ausgabe definiert.
1417	Das Dokument ist nicht in der Schemaansicht geöffnet. Vielleicht handelt es sich um keine '.xsd' Datei.
1421	Die Funktion steht in dieser Edition nicht zur Verfügung.
1422	Fehler bei der Generierung

29.3.2.10.33 GenerateXBRLDocumentation

Methode: GenerateXBRLDocumentation (*objDlg* als XBRLDocumentationDlg[1958])

Beschreibung
Generiert Dokumentation für eine XBRL-Definitionsdatei im HTML-, MS-Word- oder RTF-Format. Der Parameter "objDlg" dient zum Parametrisieren des Generierungsvorgangs. Verwenden Sie Dialogs.XBRLDocumentationDlg[1838] um eine initialisierte Gruppe von Optionen aufzurufen. Sie müssen zumindest die Eigenschaft XBRLDocumentationDlg.OutputFile[1964] definieren, bevor Sie mit der Generierung beginnen können. Beachten Sie, dass diese Funktionalität nur für die XBRL-Ansicht zur Verfügung steht.Siehe Document.CurrentViewMode[1849] und SPYViewModes[2032].

Fehler

1400	Das document-Objekt ist nicht mehr gültig.
1407	Es wurden entweder ungültige Parameter übergeben oder es wurde ein leerer Dateiname für die Ausgabe definiert.
1417	Das Dokument ist nicht in der Schemaansicht geöffnet. Vielleicht handelt es sich um keine '.xsd' Datei.
1421	Die Funktion steht in dieser Edition nicht zur Verfügung.
1422	Fehler bei der Generierung

29.3.2.10.34 GetDBStructureList

Methode: GetDBStructureList (*pDatabase* als DatabaseConnection[1823]) als ElementList[1878]

Beschreibung
GetDBStructureListerstellt anhand des Schema-Dokuments eine Sammlung von Elementen, für die Tabellen in der definierten Datenbank erstellt werden. Die Funktion gibt eine Sammlung von ElementListItemszurück, deren ElementListItem.Name[1880]Eigenschaften die Namen der Tabellen enthalten.

Siehe auch CreateDBStructureFromXMLSchema[1848].

Fehler
1400 Das Objekt ist nicht mehr gültig.
1427 Es konnte kein Parser für die definierte XML-Datei erstellt werden.
1428 Der Export der Elementliste ist fehlgeschlagen.
1429 Datenbankauswahl fehlt.

29.3.2.10.35 GetExportElementList

Methode: GetExportElementList (*pFromChild* als XMLData[1970], *pExportSettings* als ExportSettings[1880]) als ElementList[1878]

Beschreibung
GetExportElementListerstellt eine Elementsammlung für den Export aus dem Dokument. Der Export wird durch die Einstellungen in pExportSettingsdefiniert und beginnt beim Element pFromChild.Die Funktion gibt eine Sammlung vonElementListItemszurück, wobei die Eigenschaften ElementListItem.Name[1880]die Namen der Tabellen enthalten, die aus dem Dokument exportiert werden können. Die Eigenschaft ElementListItem.FieldCount[1880]enthält die Anzahl der Spalten in der Tabelle. Die Eigenschaft ElementListItem.RecordCount[1880]enthält die Anzahl der Datensätze in der Tabelle. Die Eigenschaft ElementListItem.ElementKind[1879]wird nicht verwendet. `UpdateXMLData()`[1869]wird eventuell indirekt benötigt, da Sie `XMLData`[1970]als Parameter an diese Funktion übergeben müssen.

Siehe auch Import und Export von Daten[1715].

Fehler
1400 Das Objekt ist nicht mehr gültig.
1407 Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.
1427 Es konnte kein Parser für die angegebene XML-Datei erstellt werden.
1428 Der Export der Elementliste ist fehlgeschlagen.

29.3.2.10.36 GetPathName (nicht mehr verwendet)

Wird ersetzt durch Document.FullName[1854]

```
// ----- javascript sample -----
// instead of:
// strPathName = Application.ActiveDocument.GetPathName();
// use now:
strPathName = Application.ActiveDocument.FullName;
```

Methode: GetPathName() als String

Beschreibung
Die Methode GetPathNameruft den Pfad das aktiven Dokuments auf.

Siehe auch Document.SetPathName[1866] (nicht mehr verwendet).

29.3.2.10.37 GridView

Eigenschaft: GridView als GridView[1902]

Beschreibung
Über diese Eigenschaft erhalten Sie Zugriff auf die Funktionalitäten der Grid-Ansicht für das Dokument.

Fehler
1400 Das Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
1417 Das Dokument muss in der Grid-Ansicht geöffnet sein.

29.3.2.10.38 IsModified

Eigenschaft: IsModified als Boolean

Beschreibung
True, wenn das Dokument geändert wurde.

Fehler
1400 Das Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.10.39 IsValid

Methode: HRESULT IsValid([in, out] VARIANT *strError, [in, out] VARIANT *nErrorPos, [in, out] VARIANT *pBadData, [out,retval] VARIANT_BOOL *bValid);

Rückgabewert
True, wenn das Dokument gültig ist, false, wenn es nicht gültig ist. Um `IsValid()` aufrufen zu können, muss die Benutzeroberfläche der Applikation sichtbar sein. (Wenn Sie eine Validierung durchführen wollen, ohne dass die Benutzeroberfläche angezeigt wird, verwenden Sie bitte Altova RaptorXML Server.)

Beschreibung

`IsValid` validiert das Dokument anhand des damit verknüpften Schemas bzw. der DTD. `strError` gibt dieselbe Fehlermeldung zurück, wie wenn Sie die Datei über die Benutzeroberfläche validieren.

Fehler

1400 Das Objekt ist nicht mehr gültig.
1407 Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.
1408 Datei kann nicht validiert werden.

Beispiel

Im folgenden C++-Codefragment finden Sie ein Beispiel für die Verwendung der Methode **`IsValid`**.

```
#import "XMLSpy.tlb"


CComPtr< XMLSpyLib::IDocument12> ipDoc = ipXMLSpy->GetActiveDocument();
if ( ipDoc )
{
	// prepare in/out parameters for IsValid call
	CComVariant variantError;
	CComVariant variantErrorPos;
	CComVariant variantBadData;
	// IsValid always shows a dialog with the validation result. This cannot be turned
off.
	bool bIsValid = ipDoc->IsValid( &variantError, &variantErrorPos, &variantBadData )
== VARIANT_TRUE;

	if ( !bIsValid )
	{
		// retrieve values from out parameters
		CString strError = (V_VT( &variantError ) == VT_BSTR ?
V_BSTR( &variantError ) : _T( "" ));
		long npos = (V_VT( &variantErrorPos ) == VT_I4 ? V_I4( &variantErrorPos ) : -
1);
		CComQIPtr< XMLSpyLib::IXMLData > ipXMLBadData = (V_VT( &variantBadData ) ==
VT_DISPATCH ? V_DISPATCH( &variantBadData ) : nullptr);

		if ( ipXMLBadData )
			strError += CString( _T("\n\n Node: ") ) + (LPCWSTR)ipXMLBadData-
>GetName();

		if ( !strError.IsEmpty() )
			AfxMessageBox( "Validation failed - " + strError );
	}
}
```


© 2016-2022 Altova GmbH

29.3.2.10.40 IsValidEx

Methode: IsValidEx (*nXSDVersion* als SPYValidateXSDVersion[2031], *nErrorLimit* als `int`, *nErrorFormat* als SPYValidateErrorFormat[2032], out *strError* als Variante) als Boolean

Rückgabewert
"True", wenn das Dokument gültig ist, andernfalls "false'.

Beschreibung
`IsValidEx` validiert das Dokument anhand des damit verknüpften Schemas/der damit verknüpften DTD.

In-Parameter:
`nXSDVersion`, ein Enumerationswert von SPYValidateXSDVersion[2031], womit die XDS-Version ausgewählt wird, anhand der die Validierung durchgeführt werden soll.
`nErrorLimit`, ein Ganzzahlwert. Der Wert muss `0` oder eine positive Ganzzahl sein oder `-1` für `unlimited`.
`nErrorFormat`, ein Enumerationswert von SPYValidateErrorFormat[2032], womit die XDS-Version ausgewählt wird, anhand der die Validierung durchgeführt werden soll.

Out-Parameter:
`strError` ist die Fehlermeldung und mit der Meldung identisch, die man bei Validierung der Datei über die Benutzeroberfläche erhält.

Fehler

1400	Das Objekt ist nicht mehr gültig.
1407	Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.
1408	Datei kann nicht validiert werden.

Beispiel
Im folgenden C++-Codefragment finden Sie ein Beispiel für die Verwendung der Methode **`IsValidEx`**.

```
#import "XMLSpy.tlb"


CComPtr< XMLSpyLib::IDocument12> ipDoc = ipXMLSpy->GetActiveDocument();
if ( ipDoc )
{
	CComVariant variantErrorEx;
	bool bIsValidEx = ipDoc->IsValidEx( XMLSpyLib::spyValidateXSDVersion_AutoDetect,
100, XMLSpyLib::SPYValidateErrorFormat_LongXML, &variantErrorEx ) == VARIANT_TRUE;

	//	// retrieve values from out parameters
	CString strErrorEx = (V_VT( &variantErrorEx ) == VT_BSTR ?
V_BSTR( &variantErrorEx ) : _T( "" ));
	if ( !strErrorEx.IsEmpty() )
		AfxMessageBox( "Validation failed - " + strErrorEx );
}
```

29.3.2.10.41 IsWellFormed

Methode: IsWellFormed (*pData* als XMLData[1970], *bWithChildren* als Boolean, *strError* als Variant, *nErrorPos* als Variant, *pBadXMLData* als Variant) als Boolean

Rückgabewert
True, wenn das Dokument wohlgeformt ist.

Beschreibung
IsWellFormedüberprüft das Dokument beginnend beim Element pDataauf Wohlgeformtheit.

Wenn das Dokument nicht wohlgeformt ist, enthält strErroreine Fehlermeldung, nErrorPosdie Position der Datei und pBadXMLDataenthält eine Referenz auf das Element, das die Wohlgeformtheit verletzt. Diese Out-Parameter sind als VARIANTs definiert, um Scripting-Sprachen wie VBScript zu unterstützen.

Fehler
1400 Das Objekt ist nicht mehr gültig.
1407 Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben

Beispiel

Siehe IsValid.

29.3.2.10.42 Name

Eigenschaft: Name als String (schreibgeschützt)

Beschreibung
Verwenden Sie diese Eigenschaft, um den Namen - ohne Pfad - der Dokumentdatei abzurufen. Um den Dateinamen für ein Dokument zu ändern, verwenden Sie die Eigenschaft FullName[1854].

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.10.43 Parent

Eigenschaft: Parentals Documents[1870](schreibgeschützt)

Beschreibung
Ruft das übergeordnete Objekt des Document-Objekts auf.

Fehler
1400 Das document-Objekt ist nicht mehr gültig.

© 2016-2022 Altova GmbH

1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Eigenschaft: Parent als Application[1746] (schreibgeschützt)

29.3.2.10.44 Path

Eigenschaft: Path als String (schreibgeschützt)

Beschreibung
Verwenden Sie diese Eigenschaft, um den Pfad der Dokumentdatei - ohne Dateinamen - abzurufen. Um den Dateinamen und Pfad für ein Dokument zu ändern, verwenden Sie die Eigenschaft FullName[1854].

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.10.45 RootElement

Eigenschaft: RootElement als XMLData[1970] (schreibgeschützt)

Beschreibung
Über die Eigenschaft RootElementhaben Sie Zugriff auf das Root-Element der XML-Struktur des Dokuments einschließlich der XML-Prologdaten. Um das erste Element des Inhalts eines Dokuments aufzurufen, navigieren Sie zum ersten Child der Art *spyXMLDataElement*oder verwenden Sie die Eigenschaft Document.DataRoot[1849]. Wenn CurrentViewMode[1849] nicht *spyViewGrid* oder *spyViewAuthentic* ist, ist möglicherweise ein UpdateXMLData[1869] erforderlich, um Zugriff auf die neuesten XMLData[1970] zu erhalten.

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.10.46 Save

Methode: Save()

Beschreibung
Die Methode schreibt alle Änderungen das Dokuments in die dazugehörige Datei. Siehe auch Document.FullName[1854].

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1407 Es wurde ein leerer Dateiname angegeben.
1403 Fehler beim Speichern der Datei. Wahrscheinlich ist der Dateiname ungültig.

29.3.2.10.47 SaveAs

Methode: SaveAs (*strFileName* als String)

Beschreibung
Speichert das Dokument in der angegebenen Datei.Wenn das Dokument erfolgreich gespeichert werden konnte, wird die Eigenschaft FullName[1854] auf den angegebenen Dateinamen gesetzt.

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1407 Es wurde ein leerer Dateiname angegeben.
1403 Fehler beim Speichern der Datei. Wahrscheinlich ist der Dateiname ungültig.

29.3.2.10.48 Saved

Eigenschaft: Saved als Boolean (schreibgeschützt)

Beschreibung
Mit Hilfe dieser Eigenschaft können Sie überprüfen, ob das Dokument seit der letzten Änderung gespeichert wurde. Sie gibt die Negation von IsModified[1859] zurück.

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.10.49 SaveInString

Methode: SaveInString (*pData* als XMLData[1970], *bMarked* als Boolean) als String

Parameter
pData
XMLData Element als Start. Setzen Sie pData auf Document.RootElement[1863], wenn die gesamte Datei kopiert werden soll.

bMarked
Wenn bMarked "true" ist, werden nur die in der Grid-Ansicht ausgewählten Elemente kopiert.

Rückgabewert
Gibt einen String mit den XML-Daten zurück.

Beschreibung
SaveInString beginnt am Element pData und konvertiert die XMLData Objekte in eine String-Darstellung. `UpdateXMLData()`[1869] wird eventuell indirekt benötigt, da Sie `XMLData`[1970] als Parameter an diese Funktion übergeben müssen.

Fehler
1400 Das Objekt ist nicht mehr gültig.

© 2016-2022 Altova GmbH

1407 Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.

29.3.2.10.50 SaveToURL

Methode: SaveToURL (*strURL* als String, *strUser* als String, *strPassword* als String)

Rückgabewert

Beschreibung
SaveToURL()schreibt das Dokument in die URL strURL. Diese Methode definiert den permanenten Dateipfad des Dokuments nicht.

Fehler
1400 Das Objekt ist nicht mehr gültig.
1402 Es wurde eine ungültige URL angegeben.
1403 Fehler beim Speichern unter der URL.

29.3.2.10.51 SetActiveDocument

Methode: SetActiveDocument()

Beschreibung
Die Methode definiert das Dokument als das aktive und bringt es in den Vordergrund.

Fehler
1400 Das Objekt ist nicht mehr gültig.

29.3.2.10.52 SetEncoding (nicht mehr verwendet)

Wird ersetzt durch Document.Encoding 1850

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.SetEncoding("UTF-16");
// use now:
Application.ActiveDocument.Encoding = "UTF-16";
```

Methode: SetEncoding (*strEncoding* als String)

Beschreibung
SetEncoding definiert die Kodierung des Dokuments wie der Menübefehl "Datei/Kodierung..." in XMLSpy. Mögliche Werte für strEncoding sind z.B.:

8859-1,
8859-2,
ASCII, ISO-646,

850,
1252,
1255,
SHIFT-JIS, MS-KANJI,
BIG5, FIVE,
UTF-7,
UTF-8,
UTF-16

29.3.2.10.53 SetExternalIsValid

Methode: SetExternalIsValid (*bValid* als Boolean)

Parameter

bValid
Definiert das Ergebnis eines externen Validierungsvorgangs.

Beschreibung
Die mit dieser Methode definierte interne Information wird nur beim Abbruch der Standardvalidierung in jedem OnBeforeValidate[1842] Handler abgefragt.

Verfügbar mit Typbibliothek Version 1.5

Fehler
1400 Das Objekt ist nicht mehr gültig.

29.3.2.10.54 SetPathName (nicht mehr verwendet)

Wird ersetzt durch Document.FullName[1854]

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.SetPathName("C:\\myXMLFiles\\test.xml");
// use now:
Application.ActiveDocument.FullName = "C:\\myXMLFiles\\test.xml";
```

Methode: SetPathName (*strPath* als String)

Beschreibung
Die Methode SetPathNamedefiniert den Pfad des aktiven Dokuments. SetPathNamekopiert den String nur und überprüft nicht, ob der Pfad gültig ist. Alle nachfolgenden Speicheroperationen erfolgen in dieser Datei.

© 2016-2022 Altova GmbH

29.3.2.10.55 StartChanges

Methode: StartChanges()

Beschreibung
Nach Ausführung von StartChanges aktualisiert XMLSpy seine Editor-Fenster erst, wenn Document.EndChanges[1850] aufgerufen wird. Dadurch wird die Geschwindigkeit bei der Ausführung komplexer Operationen an der XML-Struktur verbessert.

Fehler
1400 Das Objekt ist nicht mehr gültig.

29.3.2.10.56 Suggestions

Eigenschaft: Suggestions als Array

Beschreibung
Diese Eigenschaft enthält die letzten gültigen Vorschläge des Benutzers für dieses Dokument. Die von XMLSpy generierten Vorschläge können bearbeitet werden, bevor sie dem Benutzer im OnBeforeShowSuggestions[1929] Event angezeigt werden.

Fehler
1400 Das Objekt ist nicht mehr gültig.
1407 Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.

29.3.2.10.57 SwitchViewMode

Methode: SwitchViewMode (*nMode* als SPYViewModes[2032]) als Boolean

Rückgabewert
Gibt "true" zurück, wenn der Ansichtsmodus gewechselt wird.

Beschreibung
Die Methode definiert den aktuellen Ansichtsmodus des Dokuments in XMLSpy. Siehe auch Document.CurrentViewMode[1849].

Fehler
1400 Das Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
1417 Es wurde ein ungültiger Ansichtsmodus definiert.

29.3.2.10.58 TextView

Eigenschaft: TextView als TextView[1928]

Beschreibung

Über diese Eigenschaft haben Sie Zugriff auf die Textansichtsfunktionalität des Dokuments.

Fehler
1400 Das Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.10.59 Title

Eigenschaft: Title als String (schreibgeschützt)

Beschreibung
Titleenthält den Dateinamen des Dokuments. Um den Pfad und Dateinamen der Datei abzurufen, verwenden Sie FullName[1854].

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.10.60 TransformXSL

Methode: TransformXSL()

Beschreibung
TransformXSL verarbeitet das XML-Dokument anhand der damit verknüpften XSL-Datei. Informationen, wie man in das Dokument eine Referenz auf eine XSL-Datei platziert, finden Sie unter Document.AssignXSL[1844].

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1411 Fehler bei der Transformation.

29.3.2.10.61 TransformXSLEx

Method: TransformXSL(*nAction*as SPYDialogAction[2023])

Beschreibung
TransformXSL verarbeitet das XML-Dokument über die damit verknüpfte XSL-Datei. Informationen, wie man in das Dokument eine Referenz auf eine XSL-Datei platziert, finden Sie unter Document.AssignXSL[1844].

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1411 Fehler bei der Transformation.

29.3.2.10.62 TransformXSLFO

Methode: TransformXSLFO()

Beschreibung

© 2016-2022 Altova GmbH

TransformXSLFO verarbeitet das XML-Dokument anhand der damit verknüpften XSLFO-Datei. Informationen, wie man in das Dokument eine Referenz auf eine XSLFO-Datei platziert, finden Sie unter AssignXSLFO[1844]. Bevor Sie diese Methode verwenden, müssen Sie XMLSpy einen FOP-Prozessor zuweisen.

Fehler
1400 Das document-Objekt ist nicht mehr gültig.
1411 Fehler bei der Transformation.

29.3.2.10.63 TreatXBRLInconsistenciesAsErrors

Eigenschaft: TreatXBRLInconsistenciesAsErrors als Boolean

Beschreibung
Wenn diese Eigenschaft auf `true` gesetzt ist, gibt die Methode `Document.IsValid()` den Wert `false` bei XBRL-Instanzen zurück, die Inkonsistenzen mit der XBRL-Spezifikation enthalten. Der Standardwert dieser Eigenschaft ist `false`.

Errors
1400 Das document-Objekt ist nicht mehr gültig.
1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.10.64 UpdateViews

Methode: UpdateViews()

Beschreibung
Rufen Sie zum Neuzeichnen der Grid-Ansicht und der Strukturansicht UpdateViewsauf. Dies ist manchmal von Bedeutung, nachdem Sie die XMLData-Struktur eines Dokuments geändert haben. Diese Methode zeichnet die Textansicht von XMLSpy nicht neu.

Fehler
1400 Das document-Objekt ist nicht mehr gültig.

29.3.2.10.65 UpdateXMLData

Methode: UpdateXMLData() als Boolean

Beschreibung
Die XMLData[1970]-Struktur wird anhand der aktuellen Ansicht aktualisiert. Beachten Sie bitte, dass dieser Vorgang in der Textansicht fehlschlagen kann, wenn der aktuelle XML-Text nicht wohlgeformt ist. Dies ist nicht notwendig, wenn CurrentViewMode[1849] *spyViewGrid*oder *spyViewAuthentic* ist, weil die XMLData[1970]in diesen Ansichten automatisch aktualisiert werden.

Verfügbar mit Typbibliothek Version 1.5

Fehler
1400 Das document-Objekt ist nicht mehr gültig.

29.3.2.11 Documents

Eigenschaften
Count 1871
Item 1871

Methoden
NewFile 1871
OpenFile 1872
OpenURL 1873
OpenURLDialog 1873
NewFileFromText 1872

Beschreibung
Diese Objekt repräsentiert die gerade in XMLSpy geöffneten Dokumente. Verwenden Sie dieses Objekt, um weitere Dokumente zu öffnen oder durch die bereits offenen Dokumente zu iterieren.

Beispiele

```
' ----------------------------------------
' Scripting environment - VBScript
' iterate through open documents
' ----------------------------------------
DimobjDocuments
SetobjDocuments = Application.Documents

ForEachobjDoc InobjDocuments
        'do something useful with your document
        objDoc.SetActiveDocument()
Next
```

```
// ----------------------------------------
// Scripting environment - JScript
// close all open documents
// ----------------------------------------
for(var iter = newEnumerator (Application.Documents);
   ! iter.atEnd();
   iter.moveNext())
{
        // MsgBox ("Closing file " + iter.item().Name);
        iter.item().Close (true);
}
```


© 2016-2022 Altova GmbH

29.3.2.11.1 Count

Eigenschaft: Count als long

Beschreibung
Anzahl der offenen Dokumente.

29.3.2.11.2 Item

Methode: Item (*n* als long) als Document[1838]

Beschreibung
Ruft das Dokument mit dem Index n in dieser Sammlung ab. Der Index ist 1-basiert.

29.3.2.11.3 NewAuthenticFile

Methode: NewAuthenticFile (*strSPSPath* als String, *strXMLPath* als String) als Document[1838]

Parameter
strSPSPath
Der Pfad zum SPS-Dokument.

strXMLPath
Der neue Name des XML-Dokuments.

Rückgabewert
Die Methode gibt das neue Dokument zurück.

Beschreibung
NewAuthenticFile erstellt eine neue XML-Datei und öffnet es in der Authentic-Ansicht wobei es das SPS-Design strSPSPath verwendet.

29.3.2.11.4 NewFile

Methode: NewFile (*strFile* als String, *strType* als String) als Document[1838]

Parameter
strFile
Der vollständige Pfad der neuen Datei.

strType
Der Typ der neuen Datei als String (z.B. "xml", "xsd", ...)

Rückgabewert
Gibt die neue Datei zurück.

Beschreibung

NewFileerstellt eine neue Datei vom Typ strType(d.h. "xml"). Die neu erstellte Datei ist auch das ActiveDocument.

29.3.2.11.5 NewFileFromText

Methode: NewFileFromText (*strText* als String, *strType* als String) als Document[1838]

Parameter
strText
Der Inhalt des neuen Dokuments ist unformatierter Text.

strType
Der zu erstellende Dokumenttyp (d.h. "xml").

Rückgabewert
Die Methode gibt das neue Dokument zurück.

Beschreibung
NewFileFromText erstellt ein neues Dokument mit strText als seinem Inhalt.

29.3.2.11.6 OpenAuthenticFile

Methode: OpenAuthenticFile (*strSPSPath* als String, *strXMLPath* als String) als Document[1838]

Parameter
strSPSPath
Der Pfad zum SPS-Dokument.

strXMLPath
Der Pfad zum XML-Dokument (kann leer sein).

Rückgabewert
Die Methode gibt das neue Dokument zurück.

Beschreibung
OpenAuthenticFile öffnet eine XML-Datei oder Datenbank in der Authentic-Ansicht wobei es das SPS-Design strSPSPath verwendet.

29.3.2.11.7 OpenFile

Methode: OpenFile (*strPath* als String, *bDialog* als Boolean) als Document[1838]

Parameter
strPath
Pfad und Dateiname der zu öffnenden Datei.

bDialog
Zeigt die Dialogfelder für die Benutzereingaben an..

© 2016-2022 Altova GmbH

Rückgabewert
Gibt die geöffnete Datei zurück, wenn die Methode erfolgreich war.

Beschreibung
OpenFileöffnet die Datei strPath. Wenn bDialogTRUE ist, wird ein Dateidialogfeld angezeigt.

Beispiel

```
DimobjDoc AsDocument
SetobjDoc = objSpy.Documents.OpenFile(strFile, False)
```

29.3.2.11.8 OpenURL

Methode: OpenURL (*strURL* als String, *nURLType* als SPYURLTypes 2031, *nLoading* als SPYLoading 2025, *strUser* als String, *strPassword* als String) als Document 1838

Parameter
strURL
URL zum Öffnen eines Dokuments.

nURLType
Typ des zu öffnenden Dokuments. Setzen Sie den Wert auf -1, um den Typ automatisch zu ermitteln.

nLoading
Setzen Sie nLoading auf 0 (Null), wenn das Dokument aus dem Cache oder vom Proxy-Server geladen werden soll. Setzen Sie nLoading andernfalls auf 1.

strUser
Name des Benutzers, falls erforderlich. Kann leer bleiben.

strPassword
Passwort für die Authentifizierung. Kann leer bleiben.

Rückgabewert
Die Methode gibt das geöffnete Dokument zurück.

Beschreibung
OpenURL öffnet die URL strURL.

29.3.2.11.9 OpenURLDialog

Methode: OpenURLDialog (*strURL* als String, *nURLType* als SPYURLTypes 2031, *nLoading* als SPYLoading 2025, *strUser* als String, *strPassword* als String) als Document 1838

Parameter
strURL
URL zum Öffnen eines Dokuments.

nURLType

Typ des zu öffnenden Dokuments. Setzen Sie den Wert auf -1, um den Typ automatisch zu ermitteln.

nLoading
Setzen Sie nLoading auf 0 (Null), wenn das Dokument aus dem Cache oder vom Proxy-Server geladen werden soll. Setzen Sie nLoading andernfalls auf 1.

strUser
Name des Benutzers, falls erforderlich. Kann leer bleiben.

strPassword
Passwort für die Authentifizierung. Kann leer bleiben.

Rückgabewert
Die Methode gibt das geöffnete Dokument zurück.

Beschreibung
OpenURLDialogzeigt dem Benutzer das Dialogfeld "URL öffnen" an und setzt in die Eingabefelder die angegebenen Parameter ein.

29.3.2.12 DTDSchemaGeneratorDlg

Eigenschaften und Methoden

Standard-Automation-Eigenschaften
Application 1874
Parent 1877

DTDSchemaFormat 1875
ValueList 1878
TypeDetection 1878
FrequentElements 1875
MergeAllEqualNamed 1876
ResolveEntities 1877
AttributeTypeDefinition 1875
GlobalAttributes 1876
OnlyStringEnums 1876
MaxEnumLength 1876
OutputPath 1877
OutputPathDialogAction 1877

Beschreibung
Mit Hilfe dieses Objekts können Sie die Generierung eines Schemas oder einer DTD konfigurieren. Die Methode GenerateDTDOrSchemaEx 1855 erwartet einen DTDSchemaGeneratorDlgals Parameter, um die Generierung sowie die damit in Zusammenhang stehenden Interaktionen mit dem Benutzer zu konfigurieren.

29.3.2.12.1 Application

Eigenschaft: Applicationals Application 1746 (schreibgeschützt)

Beschreibung

Ruft das XMLSpyapplication-Objekt auf.

Fehler
3000 Das Objekt ist nicht mehr gültig.
3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.12.2 AttributeTypeDefinition

Eigenschaft: AttributeTypeDefinitionals SPYAttributeTypeDefinition 2020

Beschreibung
Definiert, wie Attributdefinitionen zusammengeführt werden.

Fehler
3000 Das Objekt ist nicht mehr gültig.
3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.12.3 DTDSchemaFormat

Eigenschaft: DTDSchemaFormatals SPYDTDSchemaFormat 2023

Beschreibung
Setzt das Schemaausgabeformat auf DTD oder W3C.

Fehler
3000 Das Objekt ist nicht mehr gültig.
3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.12.4 FrequentElements

Eigenschaft: FrequentElementsals SPYFrequentElements 2024

Beschreibung
Sollen die Typen für alle Elemente als global definiert werden? Verwenden Sie diesen Wert *spyGlobalComplexType* um sie global zu definieren. Verwenden Sie andernfalls den Wert *spyGlobalElements*.

Fehler
3000 Das Objekt ist nicht mehr gültig.
3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.12.5 GlobalAttributes

Eigenschaft: GlobalAttributesals Boolean

Beschreibung
Sollen Attribute mit demselben Namen und Typ global aufgelöst werden?

Fehler
3000 Das Objekt ist nicht mehr gültig.
3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.12.6 MaxEnumLength

Eigenschaft: MaxEnumLengthals Integer

Beschreibung
Definiert die maximale Anzahl an Zeichen, die bei Enumerationsnamen zulässig ist. Wenn auch nur ein Wert länger als dieser ist, wird keine Enumeration generiert.

Fehler
3000 Das Objekt ist nicht mehr gültig.
3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.12.7 MergeAllEqualNamed

Eigenschaft: MergeAllEqualNamedals Boolean

Beschreibung
Sollen Typen aller Elemente mit demselben Namen in einem Typ zusammengeführt werden?

Fehler
3000 Das Objekt ist nicht mehr gültig.
3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.12.8 OnlyStringEnums

Eigenschaft: OnlyStringEnumsals Boolean

Beschreibung
Definiert, ob Enumerationen nur für unformatierte Strings oder für alle Arten von Werten erstellt werden sollen.

Fehler
3000 Das Objekt ist nicht mehr gültig.
3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.12.9 OutputPath

Eigenschaft: OutputPathals String

Beschreibung
Wählt den Dateinamen für das generierte Schema/die generierte DTD aus.

Fehler
3000 Das Objekt ist nicht mehr gültig.
3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.12.10 OutputPathDialogAction

Eigenschaft: OutputPathDialogActionals SPYDialogAction[2023]

Beschreibung
Definiert wie das untergeordnete Dialogfeld zur Auswahl des Ausgabepfads für das Schema/die DTD behandelt werden soll. Setzen Sie diesen Wert auf *spyDialogUserInput(2)*damit standardmäßig das Dialogfeld mit dem aktuellen Wert der Eigenschaft OutputPath[1877]angezeigt wird. Verwenden Sie *spyDialogOK(0)*um dem Benutzer das Dialogfeld nicht anzuzeigen.

Fehler
3000 Das Objekt ist nicht mehr gültig.
3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.12.11 Parent

Eigenschaft: Parentals Dialogs[1835](schreibgeschützt)

Beschreibung
Ruft das dem Objekt übergeordnete Objekt auf.

Fehler
3000 Das Objekt ist nicht mehr gültig.
3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.12.12 ResolveEntities

Eigenschaft: ResolveEntitiesals Boolean

Beschreibung
Sollen alle Entities vor Beginn der Generierung aufgelöst werden? Wenn ja, wird ein Infoset erstellt.

Fehler
3000 Das Objekt ist nicht mehr gültig.
3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.12.13 TypeDetection

Eigenschaft: TypeDetectionals SPYTypeDetection[2031]

Beschreibung
Definiert die Genauigkeit, mit der ein simpleTyp ermittelt wird.

Fehler
3000 Das Objekt ist nicht mehr gültig.
3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.12.14 ValueList

Eigenschaft: ValueListals Integer

Beschreibung
Generiert nicht mehr als diese Anzahl von Enumeration-Facets pro Typ. Setzen Sie den Wert auf -1, um keine Einschränkung zu setzen.

Fehler
3000 Das Objekt ist nicht mehr gültig.
3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.13 ElementList

Eigenschaften
Count[1879]
Item[1879]

Methoden
RemoveElement[1879]

Beschreibung
Elementlisten werden beim Export und Import von Daten für verschiedene Zwecke verwendet. Je nach Verwendungszweck werden unterschiedliche Eigenschaften von ElementListItem[1879] verwendet.

Die Methode kann die folgenden Eigenschaften enthalten
- eine Liste von Tabellennamen, die durch einen Aufruf von Application.GetDatabaseTables[1754] zurückgegeben werden.
- eine Liste von Feldnamen, die durch einen Aufruf von Application.GetDatabaseImportElementList[1753] oder Application.GetTextImportElementList[1755] zurückgegeben werden,

© 2016-2022 Altova GmbH

- eine Feldnamenfilterliste, die in Application.ImportFromDatabase[1757] und Application.ImportFromText[1758] verwendet wird,
- eine Liste von Tabellennamen und der Anzahl ihrer Zeilen und Spalten, die durch Aufrufen von GetExportElementList[1858] zurückgegeben werden oder
- eine in Document.ExportToDatabase[1851] und Document.ExportToText[1852] verwendete Feldnamenfilterliste.

29.3.2.13.1 Count

Eigenschaft: Countals long (schreibgeschützt)

Beschreibung
Anzahl der Elemente in dieser Sammlung.

29.3.2.13.2 Item

Methode: Item(nals long) als ElementListItem[1879]

Beschreibung
Ruft das Element mit dem Index naus dieser Sammlung ab. Das erste Element hat den Index 1.

29.3.2.13.3 RemoveElement

Methodr: RemoveElement(Indexals long)

Beschreibung
RemoveElemententfernt den Elementindex aus der Sammlung. Das erste Element hat den Index 1.

29.3.2.14 ElementListItem

Eigenschaften
Name[1880]

ElementKind[1879]

FieldCount[1880]
RecordCount[1880]

Beschreibung
Ein Element in einer ElementList[1878]. Die Verwendung seiner Eigenschaften ist vom Verwendungszweck der Elementliste abhängig. Nähere Informationen dazu siehe ElementList[1878].

29.3.2.14.1 ElementKind

Eigenschaft: ElementKindals SPYXMLDataKind[2033]

Beschreibung

Definiert, ob ein Feld als XML-Element (Datenwert von spyXMLDataElement) oder als Attribut (Datenwert von spyXMLDataAttr) importiert werden soll.

29.3.2.14.2 FieldCount

Eigenschaft: FieldCountals long (schreibgeschützt)

Beschreibung
Anzahl der Felder (d.h. Spalten) in der durch dieses Element beschriebenen Tabelle. Diese Eigenschaft ist nur nach einem Aufruf von Document.GetExportElementList[1858] gültig.

29.3.2.14.3 Name

Eigenschaft: Nameals String (schreibgeschützt)

Beschreibung
Name des Elements. Dies ist je nach Verwendungszweck der Elementliste entweder der Name einer Tabelle oder eines Felds.

29.3.2.14.4 RecordCount

Eigenschaft: RecordCountals long (schreibgeschützt)

Beschreibung
Anzahl der Datensätze (d.h. Zeilen) in der durch dieses Element beschriebenen Tabelle. Diese Eigenschaft ist nur nach einem Aufruf von Document.GetExportElementList[1858] gültig.

29.3.2.15 ExportSettings

Eigenschaften

ElementList[1881]

EntitiesToText[1881]

ExportAllElements[1881]
SubLevelLimit[1883]

FromAttributes[1882]
FromSingleSubElements[1882]
FromTextValues[1882]

CreateKeys[1881]
IndependentPrimaryKey[1882]

Namespace[1882]

© 2016-2022 Altova GmbH

ExportCompleteXML[1882]
StartFromElement[1883]

Beschreibung
ExportSettingsenthält Optionen, die beim Export von XML-Daten in eine Datenbank oder Textdatei verwendet werden.Eine allgemeine Übersicht dazu finden Sie unter Import und Export von Daten.[1715]

29.3.2.15.1 CreateKeys

Eigenschaft: CreateKeys als Boolean

Beschreibung
Diese Eigenschaft aktiviert bzw. deaktiviert die Erstellung von Schlüsseln (d.h. Primär- und Sekundärschlüssel). Die Standardeinstellung ist "True".

29.3.2.15.2 ElementList

Eigenschaft: ElementList als ElementList[1878]

Beschreibung
Standardeinstellung ist eine leere Liste. In dieser Elementliste wird definiert, welche Felder exportiert werden. Um die Liste der verfügbaren Felder abzurufen, verwenden Sie Document.GetExportElementList[1858]. Sie können Spalten vom Export ausnehmen, indem Sie Elemente mit ElementList.RemoveElement[1879]aus dieser Liste entfernen, bevor Sie sie an Document.ExportToDatabase[1851]oder Document.ExportToText[1852]übergeben.

29.3.2.15.3 EntitiesToText

Eigenschaft: EntitiesToText als Boolean

Beschreibung
Definiert, ob XML-Entities in Text konvertiert oder unverändert belassen werden sollen. Die Standardeinstellung ist True.

29.3.2.15.4 ExportAllElements

Eigenschaft: ExportAllElements als Boolean

Beschreibung
Wenn diese Eigenschaft auf truegesetzt ist, werden alle Elemente im Dokument exportiert. Wenn sie auf falsegesetzt ist, wird die Anzahl der zu exportierenden Unterebenen mittels ExportSettings.SubLevelLimit[1883] eingeschränkt. Die Standardeinstellung ist true.

29.3.2.15.5 ExportCompleteXML

Eigenschaft: ExportCompleteXML als Boolean

Beschreibung
Definiert, ob die komplette XML-Datei oder nur das durch StartFromElement[1883] definierte Element und seine untergeordneten Elemente exportiert werden sollen. Standardeinstellung ist True.

29.3.2.15.6 FromAttributes

Eigenschaft: FromAttributes als Boolean

Beschreibung
Setzen Sie FromAttributes auf "false", wenn keine Exportdaten anhand von Attributen erstellt werden sollen. Die Standardeinstellung ist True.

29.3.2.15.7 FromSingleSubElements

Eigenschaft: FromSingleSubElements als Boolean

Beschreibung
Setzen Sie FromSingleSubElements auf "false", wenn keine Exportdaten von Elementen erstellt werden sollen. Die Standardeinstellung ist True.

29.3.2.15.8 FromTextValues

Eigenschaft: FromTextValues als Boolean

Beschreibung
Setzen Sie FromTextValues auf "false", wenn keine Exportdaten anhand von Textwerten erstellt werden sollen. Die Standardeinstellung ist "True".

29.3.2.15.9 IndependentPrimaryKey

Eigenschaft: IndependentPrimaryKey als Boolean

Beschreibung
Aktiviert bzw. deaktiviert die Erstellung unabhängiger Primärschlüsselzähler für alle Elemente. Wenn ExportSettings.CreateKeys[1881] "False" ist, wird diese Eigenschaft ignoriert. Die Standardeinstellung ist "True".

29.3.2.15.10 Namespace

Eigenschaft: Namespace als SPYExportNamespace[2024]

© 2016-2022 Altova GmbH

Beschreibung
Mit der Standardeinstellung werden alle Namespace-Präfixe aus den Elementnamen entfernt. In einigen Datenbankformaten ist der Doppelpunkt kein zulässiges Zeichen. Die Standardeinstellung ist spyNoNamespace.

29.3.2.15.11 SubLevelLimit

Eigenschaft: SubLevelLimit als Integer

Beschreibung
Definiert die Anzahl der Unterebenen, die exportiert werden soll. Diese Eigenschaft, wird ignoriert, wenn ExportSettings.ExportAllElements[1881] trueist.

29.3.2.16 FileSelectionDlg

Eigenschaften und Methoden

Standard-Automation-Eigenschaften
Application[1883]
Parent[1884]

Dialogfeldeigenschaften
FullName[1884]

Annahme oder Abbruch der Aktion, die das Event verursacht hat.
DialogAction[1884]

Beschreibung
Mit Hilfe dieses Dialogfeldobjekts können Sie Informationen über ein Event empfangen und Daten an den Event Handler auf dieselbe Art zurückgeben, wie über ein Benutzer-Dialogfeld. Mit Hilfe von FileSelectionDlg.FullName[1884]können Sie den Dateipfad auswählen oder ändern und mit Hilfe der Eigenschaft FileSelectionDlg.DialogAction[1884]können Sie die Aktion, die das Event verursacht hat, abbrechen oder akzeptieren.

29.3.2.16.1 StartFromElement

Eigenschaft: StartFromElement als String

Beschreibung
Definiert das Startelement für den Export. Diese Eigenschaft wird nur berücksichtigt, wenn ExportCompleteXML[1882]"false" ist.

29.3.2.16.2 Application

Eigenschaft: Applicationals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
2400 Das Objekt ist nicht mehr gültig.
2401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.16.3 DialogAction

Eigenschaft: DialogActionals SPYDialogAction 2023

Beschreibung
Wenn Sie möchten, dass Ihr Script die Dateiauswahl ohne Interaktion durch den Benutzer durchführt, simulieren Sie die Interaktion durch den Benutzer, indem Sie die Eigenschaft entweder auf *spyDialogOK(0)*oder *spyDialogCancel(1)*setzen.
Damit Ihr Script die Standardwerte einsetzt, der Benutzer das Dialogfeld aber sehen und darauf reagieren kann, verwenden Sie den Wert *spyDialogUserInput(2)*. Wenn Sie in einem Event Handler ein FileSelectionDlg Objekt empfangen, wird *spyDialogUserInput(2)*nicht unterstützt und als *spyDialogOK(0)*interpretiert.

Fehler
2400 Das Objekt ist nicht mehr gültig.
2401 Ungültiger Wert für die Dialogfeldaktion oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.

29.3.2.16.4 FullName

Eigenschaft: FullNameals String

Beschreibung
Ruft den vollständigen Pfad der Datei auf, die durch das Dialogfeld ausgewählt wird. Bei den meisten Events, die ein FileSelectionDlg Objekt an Sie übergeben, können Sie diesen Wert ändern und dadurch die Aktion, die das Event verursacht hat (also Laden oder Speichern unter einem anderen Pfad), beeinflussen.

Fehler
2400 Das Objekt ist nicht mehr gültig.
2401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.16.5 Parent

Eigenschaft: Parentals Dialogs 1835(schreibgeschützt)

Beschreibung
Ruft das übergeordnete Objekt des Objekts auf.

Fehler
2400 Das Objekt ist nicht mehr gültig.
2401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.17 FindInFilesDlg

Eigenschaften und Methoden

Standard-Automation-Eigenschaften
Application[1886]
Parent[1887]

Find[1886]
RegularExpression[1888]
Replace[1888]
DoReplace[1886]
ReplaceOnDisk[1888]
MatchWholeWord[1887]
MatchCase[1887]
SearchLocation[1889]
StartFolder[1889]
IncludeSubfolders[1887]
SearchInProjectFilesDoExternal[1888]
FileExtension[1886]
AdvancedXMLSearch[1885]
XMLElementNames[1891]
XMLElementContents[1890]
XMLAttributeNames[1890]
XMLAttributeContents[1889]
XMLComments[1890]
XMLCData[1890]
XMLPI[1891]
XMLRest[1891]
ShowResult[1889]

Beschreibung
Mit Hilfe dieses Objekts können Sie die Suche (oder die Erstetzung) von Strings in Dateien konfigurieren. Die Methode FindInFiles[1752] erwartet einen FindInFilesDlgals Parameter.

29.3.2.17.1 AdvancedXMLSearch

Eigenschaft: AdvancedXMLSearchals Boolean

Beschreibung
Legt fest, ob die XML-Sucheigenschaften (XMLElementNames[1891], XMLElementContents[1890], XMLAttributeNames[1890], XMLAttributeContents[1890], XMLComments[1890], XMLCData[1890], XMLPI[1891]und XMLRest[1891]) berücksichtigt werden. Die Standardeinstellung ist "false".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.2 Application

Eigenschaft: Applicationals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.3 DoReplace

Eigenschaft: DoReplaceals Boolean

Beschreibung
Legt fest, ob der gefundene String durch den in Replace[1888]definierten String ersetzt werden soll. Die Standardeinstellung ist "false"

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.4 FileExtension

Eigenschaft: FileExtensionals String

Beschreibung
Definiert den Dateifilter der Dateien, die bei der Suche berücksichtigt werden sollen. Mehrere Dateifilter müssen durch ein Semikolon voneinander getrennt werden (z.B. *.xml;*.dtd;a*.xsd). Verwenden Sie die Platzhalterzeichen * und ? um den Dateifilter zu definieren.

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.5 Find

Eigenschaft: Findals String

Beschreibung
Definiert den zu suchenden String.

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.17.6 IncludeSubfolders

Eigenschaft: IncludeSubfoldersals Boolean

Beschreibung
Legt fest, ob auch Unterordner durchsucht werden sollen. Die Standardeinstellung ist "True".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.7 MatchCase

Eigenschaft: MatchCaseals Boolean

Beschreibung
Legt fest, ob die Groß- und Kleinschreibung bei der Suche beachtet werden soll. Die Standardeinstellung ist "True".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.8 MatchWholeWord

Eigenschaft: MatchWholeWordals Boolean

Beschreibung
Legt fest, ob bei der Suche nur ganze Wörter oder Teile davon gesucht werden sollen. Die Standardeinstellung ist "false".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.9 Parent

Eigenschaft: Parentals Dialogs 1835(schreibgeschützt)

Beschreibung
Ruft das übergeordnete Objekt des Objekts auf.

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.10 RegularExpression

Eigenschaft: RegularExpressionals Boolean

Beschreibung
Definiert, ob Find[1886] eine Regular Expression enthält. Die Standardeinstellung ist "false".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.11 Replace

Eigenschaft: Replaceals String

Beschreibung
Definiert den Ersetzungsstring. Der gefundene String wird nur ersetzt, wenn DoReplace[1886] auf "true" gesetzt ist.

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.12 ReplaceOnDisk

Eigenschaft: ReplaceOnDiskals Boolean

Beschreibung
Legt fest, ob die Ersetzung direkt auf der Festplatte vorgenommen wird. Die geänderte Datei wird nicht geöffnet. Die Standardeinstellung ist "false".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.13 SearchInProjectFilesDoExternal

Eigenschaft: SearchInProjectFilesDoExternalals Boolean

Beschreibung
Definiert, ob bei einer Projektsuche die externen Ordner im offenen Projektdurchsucht werden sollen. Die Standardeinstellung ist "false".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.17.14 SearchLocation

Eigenschaft: SearchLocationals SPYFindInFilesSearchLocation 2024

Beschreibung
Definiert den Ordner für die Suche. Die Standardeinstellung ist spyFindInFiles_Documents.

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.15 ShowResult

Eigenschaft: ShowResultals Boolean

Beschreibung
Definiert, ob das Ergebnis im Ausgabefenster "In Dateien suchen" angezeigt werden soll. Die Standardeinstellung ist "false".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.16 StartFolder

Eigenschaft: StartFolderals String

Beschreibung
Definiert den Ordner, in dem die Suche auf der Festplatte beginnen soll.

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.17 XMLAttributeContents

Eigenschaft: XMLAttributeContentsals Boolean

Beschreibung
Legt fest ob Attributinhalt durchsucht werden soll, wenn AdvancedXMLSearch 1885 true ist. Die Standardeinstellung ist "True".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.18 XMLAttributeNames

Eigenschaft: XMLAttributeNamesals Boolean

Beschreibung
Definiert, ob Attributnamen durchsucht werden sollen, wenn AdvancedXMLSearch[1885]true ist. Die Standardeinstellung ist "True".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.19 XMLCData

Eigenschaft: XMLCDataals Boolean

Beschreibung
Legt fest, ob CData Tags durchsucht werden sollen, wenn AdvancedXMLSearch[1885]true ist. Die Standardeinstellung ist "True".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.20 XMLComments

Eigenschaft: XMLCommentsals Boolean

Beschreibung
Legt fest, ob Kommentare durchsucht werden sollen, wenn AdvancedXMLSearch[1885]true ist. Die Standardeinstellung ist "True".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.21 XMLElementContents

Eigenschaft: XMLElementContentsals Boolean

Beschreibung
Legt fest, ob der Elementinhalt durchsucht werden soll, wenn AdvancedXMLSearch[1885]true ist. Die Standardeinstellung ist "True".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.17.22 XMLElementNames

Eigenschaft: XMLElementNamesals Boolean

Beschreibung
Legt fest, ob Elementnamen durchsucht werden sollen, wenn AdvancedXMLSearch[1885] true ist. Die Standardeinstellung ist "True".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.23 XMLPI

Eigenschaft: XMLPIals Boolean

Beschreibung
Legt fest, ob XML-Verarbeitungsanweisungen durchsucht werden sollen, wenn AdvancedXMLSearch[1885] true ist. Die Standardeinstellung ist "True".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.17.24 XMLRest

Eigenschaft: XMLRestals Boolean

Beschreibung
Legt fest, ob der Rest der XML-Datei (der nicht durch die anderen XML-Sucheigenschaften berücksichtigt wird) durchsucht werden soll, wenn AdvancedXMLSearch[1885] true ist. Die Standardeinstellung ist "True".

Fehler
3500 Das Objekt ist nicht mehr gültig.
3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.18 FindInFilesResult

Eigenschaften und Methoden

Standard-Automation-Eigenschaften
Application[1892]
Parent[1893]

Count[1892]
Item[1892]

Path[1893]
Document[1892]

Beschreibung
Dieses Objekt repräsentiert eine Datei, die den Suchkriterien entsprochen hat. Es enthält eine Liste von FindInFilesResultMatch[1893]Objekten, die die Suchposition beschreiben.

29.3.2.18.1 Application

Eigenschaft: Applicationals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
3700 Das Objekt ist nicht mehr gültig.
3701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.18.2 Count

Eigenschaft: Countals long (schreibgeschützt)

Beschreibung
Anzahl der Elemente in dieser Sammlung.

29.3.2.18.3 Document

Eigenschaft: Pathals Document[1838](schreibgeschützt)

Beschreibung
Diese Eigenschaft gibt das Document[1838]-Objekt zurück, wenn die gefundene Datei bereits in XMLSpy geöffnet ist.

Fehler
3700 Das Objekt ist nicht mehr gültig.
3701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.18.4 Item

Methode: Item(nals long) als FindInFilesResultMatch[1893]

Beschreibung
Ruft das Element mit dem Index naus dieser Sammlung ab. Das erste Element hat den Index 1.

© 2016-2022 Altova GmbH

29.3.2.18.5 Parent

Eigenschaft: Parentals FindInFilesResults[1895] (schreibgeschützt)

Beschreibung
Ruft das dem Objekt übergeordnete Objekt auf.

Fehler
3700 Das Objekt ist nicht mehr gültig.
3701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.18.6 Path

Eigenschaft: Pathals String (schreibgeschützt)

Beschreibung
Gibt den Pfad der Datei zurück, die den Suchkriterien entsprochen hat.

Fehler
3700 Das Objekt ist nicht mehr gültig.
3701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.19 FindInFilesResultMatch

Eigenschaften und Methoden

Standard-Automation-Eigenschaften
Application[1893]
Parent[1894]

Line[1894]
Position[1895]
Length[1894]
LineText[1894]
Replaced[1895]

Beschreibung
Enthält die genaue Position des gesuchten String in der Datei.

29.3.2.19.1 Application

Eigenschaft: Applicationals Application[1746] (schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
3800 Das Objekt ist nicht mehr gültig.
3801 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.19.2 Length

Eigenschaft: Lengthals Long (schreibgeschützt)

Beschreibung
Gibt die Länge des gefundenen String zurück.

Fehler
3800 Das Objekt ist nicht mehr gültig.
3801 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.19.3 Line

Eigenschaft: Lineals Long (schreibgeschützt)

Beschreibung
Gibt die Zeilennummer des gefundenen Suchbegriffs zurück. Die Zeilennummerierung beginnt mit 0.

Fehler
3800 Das Objekt ist nicht mehr gültig.
3801 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.19.4 LineText

Eigenschaft: LineTextals String (schreibgeschützt)

Beschreibung
Gibt den Text der Zeile zurück.

Fehler
3800 Das Objekt ist nicht mehr gültig.
3801 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.19.5 Parent

Eigenschaft: Parentals FindInFilesResult 1891 (schreibgeschützt)

Beschreibung

© 2016-2022 Altova GmbH

Ruft das dem Objekt übergeordnete Objekt auf.

Fehler
3800 Das Objekt ist nicht mehr gültig.
3801 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.19.6 Position

Eigenschaft: Positionals Long (schreibgeschützt)

Beschreibung
Gibt die Anfangsposition des gesuchten Begriffs in der Zeile zurück. Die Positionsnummerierung beginnt mit 0.

Fehler
3800 Das Objekt ist nicht mehr gültig.
3801 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.19.7 Replaced

Eigenschaft: Replacedals Boolean (schreibgeschützt)

Beschreibung
True, wenn der gefundene String ersetzt wurde.

Fehler
3800 Das Objekt ist nicht mehr gültig.
3801 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.20 FindInFilesResults

Eigenschaften und Methoden

Standard-Automation-Eigenschaften
Application 1896
Parent 1896

Count 1896
Item 1896

Beschreibung
Dies ist das Ergebnis der FindInFiles 1752 Methode. Es handelt sich hiebei um eine Liste von FindInFilesResult 1891 Objekten.

29.3.2.20.1 Application

Eigenschaft: Applicationals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
3600 Das Objekt ist nicht mehr gültig.
3601 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.20.2 Count

Eigenschaft: Countals long (schreibgeschützt)

Beschreibung
Anzahl der Elemente in dieser Sammlung.

29.3.2.20.3 Item

Methode: Item(nals long) als FindInFilesResult[1891]

Beschreibung
Ruft das Element mit dem Index naus dieser Sammlung ab. Das erste Element hat den Index 1.

29.3.2.20.4 Parent

Eigenschaft: Parentas Application[1746](schreibgeschützt)

Beschreibung
Ruft das dem Objekt übergeordnete Objekt auf.

Fehler
3600 Das Objekt ist nicht mehr gültig.
3601 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21 GenerateSampleXMLDlg

Eigenschaften und Methoden

Standard-Automation-Eigenschaften
Application[1897]
Parent[1900]

© 2016-2022 Altova GmbH

NonMandatoryAttributes[1899]
NonMandatoryElements[1900]
RepeatCount[1901]
FillAttributesWithSampleData[1898]
FillElementsWithSampleData[1898]
ContentOfNillableElementsIsNonMandatory[1898]
TryToUseNonAbstractTypes[1902]
SchemaOrDTDAssignment[1901]
LocalNameOfRootElement[1899]
NamespaceURIOfRootElement[1899]
OptionsDialogAction[1900]

Nicht mehr unterstützte Eigenschaften
TakeFirstChoice - obsolete[1901]
FillWithSampleData - obsolete[1899]
Optimization - obsolete[1900]

Beschreibung
Dient zum Setzen der Parameter für die Generierung vom XML-Beispielinstanzen auf Basis eines W3C-Schemas oder einer DTD.

29.3.2.21.1 Application

Eigenschaft: Applicationals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21.2 ChoiceMode

Eigenschaft: `ChoiceMode` als SPYSampleXMLGenerationChoiceMode[2028]

Beschreibung
Definiert, welche Elemente generiert werden sollen.

Fehler

2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21.3 ConsiderSampleValueHints

Eigenschaft: `ConsiderSampleValueHints` als Boolean

Beschreibung
Wählt aus, ob `SampleValueHints`[1901] verwendet werden sollen oder nicht.

Fehler

2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21.4 ContentOfNillableElementsIsNonMandatory

Eigenschaft: ContentOfNillableElementsIsNonMandatoryals Boolean

Beschreibung
Wenn der Wert "true" ist, wird der Inhalt von Elementen die "nillable" sind, nicht als obligatorisch behandelt.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21.5 FillAttributesWithSampleData

Eigenschaft: FillAttributesWithSampleDataals Boolean

Beschreibung
Wenn der Wert "true" ist, haben Attribute Beispielinhalt.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21.6 FillElementsWithSampleData

Eigenschaft: FillElementsWithSampleDataals Boolean

Beschreibung
Wenn der Wert "true" ist, haben Elemente Beispielinhalt.

Fehler

© 2016-2022 Altova GmbH

2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21.7 FillWithSampleData - nicht mehr verwendet

Eigenschaft: FillWithSampleDataals Boolean

Beschreibung
Diese Eigenschaft wird nicht mehr verwendet. Verwenden Sie stattdessen FillAttributesWithSampleData[1898]und FillElementsWithSampleData[1898], instead.

Fehler
0001 Die Eigenschaft steht nicht mehr zur Verfügung.

29.3.2.21.8 LocalNameOfRootElement

Eigenschaft: LocalNameOfRootElementals String

Beschreibung
Definiert den lokalen Namen des Root-Elements für die generierte XML-Beispieldatei.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21.9 NamespaceURIOfRootElement

Eigenschaft: NamespaceURIOfRootElementals String

Beschreibung
Definiert die Namespace URI des Root-Elements für die generierte XML-Beispieldatei.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21.10 NonMandatoryAttributes

Eigenschaft: NonMandatoryAttributesals Boolean

Beschreibung
Wenn der Wert "true" ist, werden in der XML-Beispielinstanzdatei auch nicht obligatorische Attribute erstellt.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21.11 NonMandatoryElements

Eigenschaft: NonMandatoryElementsals Boolean

Beschreibung
Bei `true`werden Elemente, die nicht obligatorisch sind, in der XML-Beispielinstanzdatei erstellt.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21.12 Optimization - nicht mehr verwendet

Eigenschaft: Optimizationals SPYSampleXMLGenerationOptimization 2028

Beschreibung
Verwenden Sie diese Eigenschaft nicht mehr. Verwenden Sie stattdessen ChoiceMode und NonMandatoryElements

Fehler

29.3.2.21.13 OptionsDialogAction

Eigenschaft: OptionsDialogActionals SPYDialogAction 2023

Beschreibung
Damit des Script die Standardwerte einsetzt und der Benutzer das Dialogfeld sieht, und dort Eingaben machen kann, setzen Sie die Eigenschaft auf den Wert *spyDialogUserInput(2)*. Wenn Ihr Script alle Optionen im Dialogfeld "Schemadokumentation" definieren soll, ohne dass eine Interaktion durch den Benutzer erforderlich ist, verwenden Sie *spyDialogOK(0)*. Der Standardwert ist *spyDialogOK*.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Zur Definition der Eigenschaft wurde ein ungültiger Wert verwendet.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21.14 Parent

Eigenschaft: Parentas Dialogs 1835(schreibgeschützt)

Beschreibung
Ruft das dem Objekt übergeordnete Objekt auf.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.21.15 RepeatCount

Eigenschaft: RepeatCountals long

Beschreibung
Anzahl der Elemente, die für wiederholt vorkommende Typen erstellt werden soll.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21.16 SampleValueHints

Eigenschaft: `SampleValueHints` als `SPYSampleXMLGenerationSampleValueHints` 2028

Beschreibung
Definiert, wie Daten für die generierte Beispieldatei ausgewählt werden solen.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21.17 SchemaOrDTDAssignment

Eigenschaft: SchemaOrDTDAssignmentals SPYSampleXMLGenerationSchemaOrDTDAssignment 2029

Beschreibung
Definiert, wie eine Referenz auf das damit in Zusammenhang stehende Schema oder die DTD, welche sich im Dokument befindet, in eine XML-Beispieldatei umgewandelt wird.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.21.18 TakeFirstChoice - nicht mehr verwendet

Eigenschaft: TakeFirstChoiceals Boolean

Beschreibung
Diese Eigenschaft wird nicht mehr verwendet.

Fehler
0001 Die Eigenschaft steht nicht mehr zur Verfügung.

29.3.2.21.19 TryToUseNonAbstractTypes

Eigenschaft: TryToUseNonAbstractTypesals Boolean

Beschreibung
Versucht bei "true" einen nicht abstrakten Typen für xsi:type zu verwenden, wenn das Element einen abstrakten Typen hat.

Fehler
2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.22 GridView

Methoden
Deselect 1905
Select 1905

SetFocus 1905

Eigenschaften
CurrentFocus 1905

IsVisible 1905

Beschreibung
GridViewClass

29.3.2.22.1 Events

29.3.2.22.1.1 OnBeforeDrag

Event: OnBeforeDrag() als Boolean

Skripting-Umgebung - VBScript:
```
Function On_BeforeDrag()
        ' On_BeforeStartEditing = False  ' to prohibit dragging
End Function
```

Skripting-Umgebung - JScript:
```
function On_BeforeDrag()
{
        // return false;  /* to prohibit dragging */
}
```

IDE Plugin:
```
IXMLSpyPlugIn.OnEvent(4, ...)    // nEventId = 4
```

Beschreibung
Dieses Event wird beim Versuch ausgelöst, ein XMLData-Element in der Grid-Ansicht mit der Maus zu verschieben. Geben Sie *false* zurück, um zu verhindern, dass das Datenelement an eine andere Position gezogen werden kann.

29.3.2.22.1.2 OnBeforeDrop

Event: OnBeforeDrop(*objXMLData* als XMLData[1970]) als Boolean

Skripting-Umgebung - VBScript:
```
Function On_BeforeDrop(objXMLData)
        ' On_BeforeStartEditing = False  ' to prohibit dropping
End Function
```

Skripting-Umgebung - JScript:
```
function On_BeforeDrop(objXMLData)
{
        // return false;  /* to prohibit dropping */
}
```

IDE Plugin:
```
IXMLSpyPlugIn.OnEvent(5, ...)    // nEventId = 5
```

Beschreibung
This event gets fired on an attempt to drop a previously dragged XMLData element on the grid view. Return *false* to prevent the data element to be moved from its original position to the drop destination position.
Dieses Event wird beim Versuch ausgelöst, ein zuvor mit der Maus gezogenes XMLData-Element in der Grid-Ansicht an dieser Stelle abzulegen. Geben Sie *false* zurück, um zu verhindern, dass das Datenelement von seiner ursprünglichen Position an eine andere Position gezogen werden kann.

29.3.2.22.1.3 OnBeforeStartEditing

Event: OnBeforeStartEditing(*objXMLData* als XMLData[1970], *bEditingName*als Boolean)als Boolean

Skripting-Umgebung - VBScript:
```
Function On_BeforeStartEditing(objXMLData, bEditingName)
        ' On_BeforeStartEditing = False  ' to prohibit editing the field
End Function
```

Skripting-Umgebung - JScript:
```
function On_BeforeStartEditing(objXMLData, bEditingName)
{
        // return false;  /* to prohibit editing the field */
}
```

IDE Plugin:
IXMLSpyPlugIn.OnEvent(1, ...) // nEventId = 1

Beschreibung
Dieses Event wird ausgelöst, bevor der Bearbeitungsmodus für eine Grid-Zelle aktiviert wird. Wenn der Parameter *bEditingName* "true" ist, wird der Namensteil des Elements bearbeitet, wenn der Wert "false" ist, wird der Werteteil bearbeitet.

29.3.2.22.1.4 OnEditingFinished

Event: OnEditingFinished(*objXMLData* als XMLData[1970], *bEditingName*als Boolean)

Skripting-Umgebung - VBScript:
Function On_EditingFinished(*objXMLData*, *bEditingName*)
End Function

Skripting-Umgebung - JScript:
function On_EditingFinished(*objXMLData*, *bEditingName*)
{
}

IDE Plugin:
IXMLSpyPlugIn.OnEvent(2, ...) // nEventId = 2

Beschreibung
Dieses Event wird ausgelöst, wenn der Bearbeitungsmodus einer Grid-Zelle verlassen wird. Der Parameter *bEditingName* definiert, ob der Namensteil des Elements bearbeitet wurde.

29.3.2.22.1.5 OnFocusChanged

Event: OnFocusChanged(*objXMLData* als XMLData[1970], *bSetFocus*als Boolean, *bEditingName*als Boolean)

Skripting-Umgebung - VBScript:
Function On_FocusChanged(*objXMLData*, *bSetFocus*, *bEditingName*)
End Function

Skripting-Umgebung - JScript:
function On_FocusChanged(*objXMLData*, *bSetFocus*, *bEditingName*)
{
}

IDE Plugin:
IXMLSpyPlugIn.OnEvent(3, ...) // nEventId = 3

Beschreibung

© 2016-2022 Altova GmbH

Dieses Event wird ausgelöst, wenn der Cursorfokus auf eine Grid-Zelle verschoben wird oder diese nicht mehr im Fokus ist. Wenn der Parameter *bEditingName* "*true*" ist, hat sich der Fokus auf dem Namensteil des Grid-Elements geändert. Anderfalls hat sich der Fokus auf dem Werteteil geändert.

29.3.2.22.2 CurrentFocus

Eigenschaft: CurrentFocusals XMLData[1970]

Beschreibung
Enthält das XML-Element mit dem aktuellen Fokus. Diese Eigenschaft ist schreibgeschützt.

29.3.2.22.3 Deselect

Methode: Deselect(*pData*als XMLData[1970])

Beschreibung
Hebt die Auswahl des Elements pDatain der Grid-Ansicht auf.

29.3.2.22.4 IsVisible

Eigenschaft: IsVisibleals Boolean

Beschreibung
True, wenn die Grid-Ansicht die aktive Ansicht des Dokuments ist. Diese Eigenschaft ist schreibgeschützt.

29.3.2.22.5 Select

Methode: Select(*pData*als XMLData[1970])

Beschreibung
Wählt das XML-Element pDatain der Grid-Ansicht aus.

29.3.2.22.6 SetFocus

Methode: SetFocus (*pFocusData* als XMLData[1970])

Beschreibung
Setzt den Fokus auf das Element pFocusDatain der Grid-Ansicht.

29.3.2.23 SchemaDocumentationDlg

Eigenschaften und Methoden

Standard-Automation-Eigenschaften
Application 1907
Parent 1914

Eigenschaften bzgl. Interaktion und Sichtbarkeit
OutputFile 1913
OutputFileDialogAction 1913
OptionsDialogAction 1913
ShowProgressBar 1917
ShowResult 1917

Optionen und Methoden für die Dokumentgenerierung
OutputFormat 1914
EmbedDiagrams 1908
DiagramFormat 1908
MultipleOutputFiles 1912
EmbedCSSInHTML 1908
CreateDiagramsFolder 1907
GenerateRelativeLinks 1909

IncludeAll 1909
IncludeIndex 1911
IncludeGlobalAttributes 1910
IncludeGlobalElements 1910
IncludeLocalAttributes 1911
IncludeLocalElements 1911
IncludeGroups 1910
IncludeComplexTypes 1909
IncludeSimpleTypes 1912
IncludeAttributeGroups 1909
IncludeRedefines 1912
IncludeReferencedSchemas 1912

AllDetails 1907
ShowDiagram 1915
ShowNamespace 1916
ShowType 1918
ShowChildren 1915
ShowUsedBy 1919
ShowProperties 1917
ShowSingleFacets 1918
ShowPatterns 1917
ShowEnumerations 1916
ShowAttributes 1915
ShowIdentityConstraints 1916
ShowAnnotations 1914
ShowSourceCode 1918

© 2016-2022 Altova GmbH

Beschreibung
Dieses Objekt kombiniert alle Optionen für die Schemadokumentgenerierung, die auch über die Dialogfelder der Benutzeroberfläche von XMLSpyverfügbar sind. Die Dokumentgenerierungsoptionen werden mit den Werten initialisiert, die bei der letzten Generierung von Schemadokumentation verwendet wurden. Vor Verwendung des Objekts müssen Sie jedoch die EigenschaftSetOutputFile[1913]auf einen gültigen Dateipfad setzen. Verwenden Sie OptionsDialogAction[1913], OutputFileDialogAction[1913]und ShowProgressBar[1917]um den Grad der gewünschten Benutzerinteraktion festzulegen. Mit Hilfe von IncludeAll[1909]und AllDetails[1907]können Sie ganze Optionsgruppen auf einmal setzen. Durch Definition der einzelnen Eigenschaften können Sie die Optionen einzeln definieren.

29.3.2.23.1 AllDetails

Methode: AllDetails(i_bDetailsOnals Boolean)

Beschreibung
Mit Hilfe dieser Methode können Sie alle Detailoptionen aktivieren bzw. deaktivieren.

Fehler
2900 Das Objekt ist nicht mehr gültig.

29.3.2.23.2 Application

Eigenschaft: Applicationals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.3 CreateDiagramsFolder

Eigenschaft: CreateDiagramsFolderals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um ein Verzeichnis für die erstellten Bilder zu erstellen. Andernfalls werden die Diagramme neben der Dokumentation erstellt. Diese Eigenschaft steht nur zur Verfügung, wenn die Diagramme nicht eingebettet sind.Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857]verwendet wurde.Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.4 DiagramFormat

Eigenschaft: DiagramFormatals SPYImageKind 2024

Beschreibung
Diese Eigenschaft definiert den generierten Diagrammbildtyp. Diese Eigenschaft steht für HTML-Dokumentation nicht zur Verfügung. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Verwendung der Eigenschaft ist PNG.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.5 EmbedCSSInHTML

Eigenschaft: EmbedCSSInHTMLals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um die CSS-Daten in das generierte HTML-Dokument einzubetten. Andernfalls wird eine separate Datei erstellt und verknüpft.Diese Eigenschaft steht nur für die HTML-Dokumentation zur Verfügung. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation 1857 verwendet wurde.Der Standardwert für die erste Ausführung ist true

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.6 EmbedDiagrams

Eigenschaft: EmbedDiagramsals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um das Diagramm in das generierte Dokument einzubetten. Diese Eigenschaft steht für HTML-Dokumentation nicht zur Verfügung. Die Eigenschaft wird mit den Werten initialisiert, die beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurden. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.23.7 GenerateRelativeLinks

Eigenschaft: GenerateRelativeLinksals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um relative Pfade zu lokalen Dateien zu erstellen. Diese Eigenschaft steht für die HTML-Dokumentation nicht zur Verfügung. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation[1856] verwendet wurde. Die Standardeinstellung für die erste Ausführung ist "false".

Fehler
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

29.3.2.23.8 IncludeAll

Methode: IncludeAll(i_bIncludeals Boolean)

Beschreibung
Verwenden Sie diese Methode um alle Include-Optionen auszuwählen, bzw. die Auswahl für alle aufzuheben.

Fehler
2900 Das Objekt ist nicht mehr gültig.

29.3.2.23.9 IncludeAttributeGroups

Eigenschaft: IncludeAttributeGroupsals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf rue, um Attributgruppen in die Schemadokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation[1855] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.10 IncludeComplexTypes

Eigenschaft: IncludeComplexTypesals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um complexTypes in die Schemadokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.11 IncludeGlobalAttributes

Eigenschaft: IncludeGlobalAttributesals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um globale Attribute in die Schemadokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.12 IncludeGlobalElements

Eigenschaft: IncludeGlobalElementsals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um globale Elemente in die Schemadokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.13 IncludeGroups

Eigenschaft: IncludeGroupsals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um Gruppen in die Schemadokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

© 2016-2022 Altova GmbH

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.14 IncludeIndex

Eigenschaft: IncludeIndexals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um einen Index in die Schemadokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.15 IncludeLocalAttributes

Eigenschaft: IncludeLocalAttributesas Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um lokale Attribute in die Schemadokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.16 IncludeLocalElements

Eigenschaft: IncludeLocalElementsals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um lokale Elemente in die Schemadokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.17 IncludeRedefines

Eigenschaft: IncludeRedefinesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Redefines in die Schemadokumenation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.18 IncludeReferencedSchemas

Eigenschaft: IncludeReferencedSchemasals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um referenzierte Schemas in die Schemadokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.19 IncludeSimpleTypes

Eigenschaft: IncludeSimpleTypesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um simpleTypes in die Schemadokumenation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.20 MultipleOutputFiles

Eigenschaft: MultipleOutputFilesals Boolean

© 2016-2022 Altova GmbH

Beschreibung
Setzen Sie diese Eigenschaft auf true, um die Dokumentationsdateien aufzuteilen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist false.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Es wurde zum Definieren der Eigenschaft ein ungültiger Wert verwendet.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.21 OptionsDialogAction

Eigenschaft: OptionsDialogActionals SPYDialogAction[2023]

Beschreibung
Damit Ihr Script die Standardwerte einsetzt und der Benutzer das Dialogfeld sieht, bzw. dort Einstellungen vornehmen kann, setzen Sie diese Eigenschaft auf den Wert *spyDialogUserInput(2)*. Wenn Ihr Script alle Optionen im Dialogfeld "Schemadokumentation" definieren soll, ohne dass eine Interaktion mit dem Benutzer stattfindet, verwenden Sie *spyDialogOK(0)*. Die Standardeinstellung ist *spyDialogOK*.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Es wurde zum Definieren der Eigenschaft ein ungültiger Wert verwendet.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.22 OutputFile

Eigenschaft: SetOutputFileals String

Beschreibung
Der vollständige Pfad und Name der Datei, die die generierte Dokumentation enthalten soll. Bei einer HTML-Ausgabe werden zusätzliche '.png' Dateien auf Basis dieses Dateinamens generiert. Der Standardwert für diese Eigenschaft ist ein leerer String, der ersetzt werden muss, bevor Sie diesese Objekten in einem Aufruf von Document.GenerateSchemaDocumentation[1856] verwenden.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.23 OutputFileDialogAction

Eigenschaft: OutputFileDialogActionals SPYDialogAction[2023]

Beschreibung
Damit der Benutzer die Ausgabedatei in einem Dialogfeld zur Dateiauswahl auswählen kann, setzen Sie diese Eigenschaft auf *spyDialogUserInput(2)*. Wenn der in OutputFile[1913] gespeicherte Wert verwendet werden soll

und keinen Interaktion mit dem Benutzer vorgesehen ist, verwenden Sie *spyDialogOK(0)*. Der Standardwert ist *spyDialogOK*.

Fehler

2900 Das Objekt ist nicht mehr gültig.
2901 Es wurde zum Definieren der Eigenschaft ein ungültiger Wert verwendet.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.24 OutputFormat

Eigenschaft: OutputFormatals SPYSchemaDocumentationFormat[2030]

Beschreibung

Definiert, welche Art von Dokumentation generiert werden soll: HTML (Wert=0), MS-Word (Wert=1) oder RTF (Werte=2). Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist HTML.

Fehler

2900 Das Objekt ist nicht mehr gültig.
2901 Es wurde zum Definieren der Eigenschaft ein ungültiger Wert verwendet.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.25 Parent

Eigenschaft: Parentals Dialogs[1835](schreibgeschützt)

Beschreibung

Ruft das dem Objekt übergeordnete Objekt auf.

Fehler

2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.26 ShowAnnotations

Eigenschaft: ShowAnnotationsals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um die Annotationen zu einer Typdefinition in der Schemadokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.23.27 ShowAttributes

Eigenschaft: ShowAttributesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um die Typdefinitionsattribute in der Schemadokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.28 ShowChildren

Eigenschaft: ShowChildrenals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um die untergeordneten Elemente einer Typdefinition als Links in der Schemadokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.29 ShowDiagram

Eigenschaft: ShowDiagramals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um die Typdefinitionen als Diagramme in der Schemadokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.30 ShowEnumerations

Eigenschaft: ShowEnumerationsals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um die in einer Typdefinition enthaltenen Enumerationen in der Schemadokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.31 ShowIdentityConstraints

Eigenschaft: ShowIdentityConstraints als Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um die Identity Constraints einer Typdefinition in der Schemadokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.32 ShowNamespace

Eigenschaft: ShowNamespaceals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um den Namespace der Typdefinitionen in der Schemadokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.23.33 ShowPatterns

Eigenschaft: ShowPatternsals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um die Patterns einer Typdefinition der Schemadokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.34 ShowProgressBar

Eigenschaft: ShowProgressBarals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um das Fenster für die Dokumentgenerierung sichtbar zu machen. Verwenden Sie false, um es auszublenden. Der Standardwert ist false.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.35 ShowProperties

Eigenschaft: ShowPropertiesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um die Eigenschaften der Typdefinition in der Schemadokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.36 ShowResult

Eigenschaft: ShowResultals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um das erzeugte Dokument automatisch zu öffnen, wenn die Generierung erfolgreich abgeschlossen wurde. HTML-Dokumentation wird in XMLSpygeöffnet. Um Word-Dokumentation anzuzeigen, wird MS-Word gestartet. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.37 ShowSingleFacets

Eigenschaft: ShowSingleFacetsals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um die Facets einer Typdefinition in der Schemadokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.38 ShowSourceCode

Eigenschaft: ShowSourceCodeals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um den XML-Quellcode für die Typdefinitionen in der Schemadokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.39 ShowType

Eigenschaft: ShowTypeals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um den Typ von Typdefinitionen in der Schemadokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von

© 2016-2022 Altova GmbH

Document.GenerateSchemaDocumentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.40 ShowUsedBy

Eigenschaft: ShowUsedByals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um die used-by-Beziehung für Typdefinitionen in der Schemadokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateSchemaDocumentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.41 SPSFile

Eigenschaft: SPSFileals String

Beschreibung
Vollständiger Pfad der SPS-Datei, die zur Generierung der Dokumentation verwendet wird.

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.23.42 UseFixedDesign

Eigenschaft: UseFixedDesignals Boolean

Beschreibung
Gibt an, on die Dokumentation mit einem festgelegten Design oder mit einem Design erstellt werden soll, das durch eine SPS-Datei festgelegt wird (wobei StyleVision benötigt wird).

Fehler
2900 Das Objekt ist nicht mehr gültig.
2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.24 SpyProject

Methoden
CloseProject 1920
SaveProject 1921
SaveProjectAs 1921

Eigenschaften
RootItems 1920
ProjectFile 1920

Beschreibung
SpyProjectClass

29.3.2.24.1 CloseProject

Deklaration: CloseProject(*bDiscardChanges* als Boolean, *bCloseFiles* als Boolean, *bDialog* als Boolean)

Parameter
bDiscardChanges
Setzen Sie bDiscardChanges auf FALSE, wenn die Änderungen in den offenen Projektdateien und im Projekt gespeichert werden sollen.

bCloseFiles
Setzen Sie bCloseFiles auf TRUE, um alle offenen Projektdateien zu schließen.

bDialog
Zeigt die Dialogfelder für die Benutzereingabe an.

Beschreibung
CloseProjectschließt das aktuelle Projekt.

29.3.2.24.2 ProjectFile

Deklaration: ProjectFile als String

Beschreibung
Pfad und Dateiname des Projekts.

29.3.2.24.3 RootItems

Deklaration: RootItems als SpyProjectItems 1924

Beschreibung
Root-Ebene der Sammlung der Projektelemente.

© 2016-2022 Altova GmbH

29.3.2.24.4 SaveProject

Deklaration: SaveProject

Beschreibung
SaveProjectspeichert das aktuelle Projekt.

29.3.2.24.5 SaveProjectAs

Deklaration: SaveProjectAs (strPath als String, bDialog als Boolean)

Parameter
strPath
Der vollständige Pfad mit dem Dateinamen der neuen Projektdatei.

bDialog
Wenn bDialogTRUE ist, wird ein Datei-Dialogfeld angezeigt.

Beschreibung
SaveProjectAsspeichertt die Projektdaten unter einem neuen Pfad.

29.3.2.25 SpyProjectItem

Methoden
Open 1922

Eigenschaften
ChildItems 1921
ParentItem 1922
FileExtensions 1922
ItemType 1922
Name 1922
Path 1923
ValidateWith 1923
XMLForXSLTransformation 1923
XSLForXMLTransformation 1923
XSLTransformationFileExtension 1923
XSLTransformationFolder 1923

Beschreibung
SpyProjectItemClass

29.3.2.25.1 ChildItems

Deklaration: ChildItems als SpyProjectItems 1924

Beschreibung

Wenn es sich bei dem Element um einen Ordner handelt, ist ChildItemsdie Sammlung des Ordnerinhalts.

29.3.2.25.2 FileExtensions

Deklaration: FileExtensions als String

Beschreibung
Dient zum Definieren der Dateierweiterungen, wenn das Projektelement ein Ordner ist.

29.3.2.25.3 ItemType

Deklaration: ItemType als SPYProjectItemTypes 2027

Beschreibung
Die Eigenschaft ist schreibgeschützt.

29.3.2.25.4 Name

Deklaration: Name als String

Beschreibung
Name des Projektelements. Die Eigenschaft ist schreibgeschützt.

29.3.2.25.5 Open

Deklaration: Open als Document 1838

Rückgabewert
Das Projektelement wird als Dokument geöffnet.

Beschreibung
Öffnet das Projektelement.

29.3.2.25.6 ParentItem

Deklaration: ParentItem als SpyProjectItem 1921

Beschreibung
Das dem aktuellen Projektelement übergeordnete Element. Kann NULL (nichts) sein, wenn das Projektelement ein Element der obersten Ebene ist.

© 2016-2022 Altova GmbH

29.3.2.25.7 Path

Deklaration: Path als String

Beschreibung
Pfad des Projektelements. Diese Eigenschaft ist schreibgeschützt.

29.3.2.25.8 ValidateWith

Deklaration: ValidateWith als String

Beschreibung
Dient zum Definieren des Schemas/der DTD für die Validierung.

29.3.2.25.9 XMLForXSLTransformation

Deklaration: XMLForXSLTransformation als String

Beschreibung
Dient zum Definieren der XML-Datei für die XSL-Transformation.

29.3.2.25.10 XSLForXMLTransformation

Deklaration: XSLForXMLTransformation als String

Beschreibung
Dient zum Definieren der XSL-Datei für die XML-Transformation.

29.3.2.25.11 XSLTransformationFileExtension

Deklaration: XSLTransformationFileExtension als String

Beschreibung
Dient zum Definieren der Dateierweiterung für die Ausgabedateien der XSL-Transformation.

29.3.2.25.12 XSLTransformationFolder

Deklaration: XSLTransformationFolder als String

Beschreibung
Dient zum Definieren des Zielordners für die Ausgabedateien der XSL-Transformation.

29.3.2.26 SpyProjectItems

Methoden
AddFile 1924
AddFolder 1924
AddURL 1924
RemoveItem 1925

Eigenschaften
Count 1925
Item 1925

Beschreibung
SpyProjectItemsClass

29.3.2.26.1 AddFile

Deklaration: AddFile(*strPath*als String)

Parameters

strPath
Der vollständige Pfadname eines neuen Projektelements

Beschreibung
Die Methode fügt eine neue Datei zur Sammlung der Projektelemente hinzu.

29.3.2.26.2 AddFolder

Deklaration: AddFolder(*strName* als String)

Parameter

strName
Name des neuen Ordners

Beschreibung
Die Methode AddFolderfügt einen Ordner mit dem Namen strNamezur Sammlung der Projektelemente hinzu.

29.3.2.26.3 AddURL

Deklaration: AddURL(*strURL* als String, *nURLType* als SPYURLTypes 2031, *strUser* als String, *strPassword* als String, *bSave* als Boolean)

Beschreibung

strURL

© 2016-2022 Altova GmbH

URL zum Öffnen als Dokument.

nURLType
Typ des zu öffnenden Dokuments. Setzen Sie die Wert zur automatischen Ermittlung auf -1.

strUser
Name des Benutzers, falls erforderlich. Kann leer bleiben.

strPassword
Passwort für die Authentifizierung. Kann leer bleiben.

bSave
Speichert die Benutzer- und Passwortinformationen.

Beschreibung
Die Methode fügt einen URL-Eintrag zur Projektsammlung hinzu.

29.3.2.26.4 Count

Deklaration: Count als long

Beschreibung
Diese Eigenschaft ruft die Anzahl der Projektelemente in der Sammlung ab. Die Eigenschaft ist schreibgeschützt.

29.3.2.26.5 Item

Deklaration: Item(*n* als long) als SpyProjectItem[1921]

Beschreibung
Ruft das nte Element der Sammlung der Projektelemente auf. Das erste Element hat den Index 1.

29.3.2.26.6 RemoveItem

Deklaration: RemoveItem(*pItem* als SpyProjectItem[1921])

Beschreibung
`RemoveItem`löscht das Element `pItem`aus der Sammlung der Projektelemente.

29.3.2.27 TextImportExportSettings

Eigenschaften nur für den Import
ImportFile[1927]

Eigenschaften nur für den Export
DestinationFolder[1926]
FileExtension[1927]

CommentIncluded 1926
RemoveDelimiter 1928
RemoveNewline 1928

Eigenschaften für Import und Export
HeaderRow 1927
FieldDelimiter 1927
EnclosingCharacter 1926
Encoding 1926
EncodingByteOrder 1927

Beschreibung
TextImportExportSettingsenthält Optionen, die Textimport- und -exportfunktionen gemeinsam haben.

29.3.2.27.1 CommentIncluded

Eigenschaft: CommentIncluded als Boolean

Beschreibung
Diese Eigenschaft legt fest, ob zusätzliche Kommentare zur generierten Textdatei hinzugefügt werden sollen. Die Standardeinstellung ist true. Diese Eigenschaft wird nur beim Export von Textdateien verwendet.

29.3.2.27.2 DestinationFolder

Eigenschaft: DestinationFolder als String

Beschreibung
Die Eigenschaft DestinationFolderlegt den Ordner fest, in dem die gespeicherten Dateien beim Textexport abgelegt werden.

29.3.2.27.3 EnclosingCharacter

Eigenschaft: EnclosingCharacter als SPYTextEnclosing 2031

Beschreibung
Diese Eigenschaft definiert das Zeichen, das alle Feldwerte für den Import und Export umschließt. Der Standardwert ist spyNoEnclosing 2031.

29.3.2.27.4 Encoding

Eigenschaft: Encoding als String

Beschreibung
Die Eigenschaft Encodingdefiniert die Zeichensatzcodierung für die zu importierenden und exportierenden Textdateien.

© 2016-2022 Altova GmbH

29.3.2.27.5 EncodingByteOrder

Eigenschaft: EncodingByteOrder als SPYEncodingByteOrder[2023]

Beschreibung
Die Eigenschaft EncodingByteOrderdefiniert die Bytefolge für Unicode-Zeichen. Standardeinstellung ist spyNONE[2023].

29.3.2.27.6 FieldDelimiter

Eigenschaft: FieldDelimiter als SPYTextDelimiters[2030]

Beschreibung
Die Eigenschaft FieldDelimiterdefiniert die Trennzeichen zwischen den Feldern beim Import und Export. Standardeinstellung ist spyTabulator[2030].

29.3.2.27.7 FileExtension

Eigenschaft: FileExtension als String

Beschreibung
Diese Eigenschaft definiert die Dateierweiterung für beim Textexport erstellte Dateien.

29.3.2.27.8 HeaderRow

Eigenschaft: HeaderRow als Boolean

Beschreibung
Die Eigenschaft HeaderRow wird beim Export und Import verwendet. Setzen Sie beim Import HeaderRow auf true, wenn die erste Zeile der Textdatei die Namen der Spalten enthält. Setzen Sie HeaderRow beim Export auf true, wenn die erste Zeile der erstellten Textdateien die Namen der Spalten enthalten soll. Der Standardwert ist true.

29.3.2.27.9 ImportFile

Eigenschaft: ImportFile als String

Beschreibung
Diese Eigenschaft dient dazu, die Textdatei für den Import zu definieren. Der String muss ein vollständig qualifizierter Pfad sein.Siehe auch Import und Export[1715].

29.3.2.27.10 RemoveDelimiter

Eigenschaft: RemoveDelimiter als Boolean

Beschreibung
Die Eigenschaft RemoveDelimiterdefiniert, ob Zeichen im Text, die dem Trennzeichen entsprechen, entfernt werden sollen. Der Standardwert ist false. Diese Eigenschaft wird beim Export in Textdateien verwendet.

29.3.2.27.11 RemoveNewline

Eigenschaft: RemoveNewline als Boolean

Beschreibung
Die Eigenschaft RemoveNewlinedefiniert, ob Zeichen für den Beginn einer neuen Zeile im Text entfernt werden sollen. Die Standardeinstellung ist false. Die Eigenschaft wird nur beim Export in Textdateien verwendet.

29.3.2.28 TextView

Eigenschaften und Methoden

Application 1930
Parent 1932

LineFromPosition 1931
PositionFromLine 1932
LineLength 1931
SelText 1933
GetRangeText 1930
ReplaceText 1932
MoveCaret 1932
GoToLineChar 1930
SelectText 1933
SelectionStart 1933
SelectionEnd 1933
Text 1934
LineCount 1931
Length 1931

Beschreibung

© 2016-2022 Altova GmbH

29.3.2.28.1 Events

29.3.2.28.1.1 OnBeforeShowSuggestions

Event: OnBeforeShowSuggestions() als Boolean

Beschreibung
Dieses Event wird ausgelöst, bevor das Vorschlagsfenster angezeigt wird. Die Document[1838]-Eigenschaft Suggestions[1867]enthält einen String Array, der dem Benutzer empfohlen wird. Die angezeigten Empfehlungen können vom Benutzer während des Events bearbeitet werden. Davor muss der Eigenschaft Suggestions[1867]ein leerer Array zugewiesen werden. Die beste Stelle dafür ist das OnDocumentOpened[1748]Event. Damit das Vorschlagsfenster nicht angezeigt wird, geben Sie falsezurück. Um es weiter anzuzeigen, verwenden Sie den Werttrue.

Beispiele
Unten finden Sie Beispiele dafür, wie dieses Event in Form eines Scripts geschrieben werden kann.

Skripting-Umgebung - VBScript:
```
Function On_BeforeShowSuggestions()
End Function
```

Skripting-Umgebung - JScript:
```
function On_BeforeShowSuggestions()
{
}
```

IDE Plugin:
```
IXMLSpyPlugIn.OnEvent(33, ...)   // nEventId = 33
```

29.3.2.28.1.2 OnChar

Event: OnChar(nCharals Long, bExistSuggestionals Boolean) als Boolean

Beschreibung
Dieses Event wird bei jedem Tastenanschlag ausgelöst. Der Parameter nCharist die Taste, die gedrückt wurde und bExistSuggestionsgibt darüber Auskunft ob ein von XMLSpygeneriertes Vorschlagsfenster nach dem Drücken dieser Taste angezeigt wird. Die Document[1838] Eigenschaft Suggestions[1867] enthält einen String Array, der dem Benutzer empfohlen wird. Die angezeigten Empfehlungen können vom Benutzer während des Events bearbeitet werden. Davor muss der Eigenschaft Suggestions[1867] ein leerer Array zugewiesen werden. Die beste Stelle dafür ist das OnDocumentOpened[1748] Event. Damit das Vorschlagsfenster nicht angezeigt wird, geben Sie false zurück. Um es weiter anzuzeigen, verwenden Sie den Wert true.
Es kann auch ein neues Vorschlagsfenster erstellt werden, wenn XMLSpy keines bereitstellt. Setzen Sie die Document[1838] Eigenschaft Suggestions[1867] auf einen String Array mit Ihren Empfehlungen und geben Sie true zurück.
Dieses Event wird vor dem OnBeforeShowSuggestions[1929] Event ausgelöst. Wenn Sie festlegen, dass das Vorschlagsfenster nicht angezeigt werden soll, indem Sie den Wert auf false setzen, dann wird OnBeforeShowSuggestions[1929] nicht ausgelöst.

Beispiele
Unterhalb finden Sit Beispiele, wie ein Skript für dieses Event erstellt werden kann.

Skripting-Umgebung - VBScript:
```
Function On_Char(nChar, bExistSuggestions)
End Function
```

Skripting-Umgebung - JScript:
```
function On_Char(nChar, bExistSuggestions)
{
}
```

IDE Plugin:
```
IXMLSpyPlugIn.OnEvent(35, ...)  // nEventId = 35
```

29.3.2.28.2 Application

Eigenschaft: Applicationals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.28.3 GetRangeText

Methode: GetRangeText(nStartals Long, nEndals Long)als String

Beschreibung
Gibt den Text im angegebenen Bereich zurück.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.28.4 GoToLineChar

Methode: GoToLineChar(nLineals Long, nCharals Long)

Beschreibung
Verschiebt den Cursor an die angegebene Zeilen- und Zeichenposition.

© 2016-2022 Altova GmbH

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.28.5 Length

Eigenschaft: Lengthals Long

Beschreibung
Gibt die Anzahl der Zeichen des Dokuments zurück.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.28.6 LineCount

Eigenschaft: LineCountals Long

Beschreibung
Gibt die Anzahl der Zeilen im Dokument zurück.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.28.7 LineFromPosition

Methode: LineFromPosition(nCharPosals Long)als Long

Beschreibung
Gibt die Zeilennummer der Zeichenposition zurück.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.28.8 LineLength

Methode: LineLength(nLineals Long)als Long

Beschreibung
Gibt die Länge der Zeile zurück.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.28.9 MoveCaret

Methode: MoveCaret(nDiffals Long)

Beschreibung
Verschiebt den Cursor um nDiffZeichen.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.28.10 Parent

Eigenschaft: Parentals Document 1838 (schreibgeschützt)

Beschreibung
Ruft das dem Objekt übergeordnete Objekt auf.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.28.11 PositionFromLine

Methode: PositionFromLine(nLineals Long) als Long

Beschreibung
Gibt die Anfangsposition der Zeile zurück.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.28.12 ReplaceText

Methode: ReplaceText(nPosFromals Long, nPosTillals Long, sTextals String)

Beschreibung
Ersetzt den Text im angegebenen Bereich.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.28.13 SelectionEnd

Eigenschaft: SelectionEndals Long

Beschreibung
Gibt die Endposition der Textauswahl zurück bzw. definiert diese.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.28.14 SelectionStart

Eigenschaft: SelectionStartals Long

Beschreibung
Gibt die Anfangsposition der Textauswahl zurück bzw. definiert diese.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.28.15 SelectText

Methode: SelectText(nPosFromals Long, nPosTillals Long)

Beschreibung
Wählt den Text im angegebenen Bereich aus.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.28.16 SelText

Eigenschaft: SelTextals String

Beschreibung
Gibt den ausgewählten Text zurück bzw. definiert diesen.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.28.17 Text

Eigenschaft: Textals String

Beschreibung
Gibt den Dokumenttext zurück bzw. definiert diesen.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29 WSDLDocumentationDlg

Eigenschaften und Methoden

Standard-Automation-Eigenschaften
Application 1935
Parent 1941

Eigenschaften zur Interaktion und Sichtbarkeit
OptionsDialogAction 1940
OutputFile 1940
OutputFileDialogAction 1940
ShowProgressBar 1944
ShowResult 1944

Optionen und Methoden zur Dokumentgenerierung
OutputFormat 1941
UseFixedDesign 1946
SPSFile 1946
EmbedDiagrams 1936
DiagramFormat 1936
MultipleOutputFiles 1939
EmbedCSSInHTML 1936
CreateDiagramsFolder 1935

IncludeAll 1937
IncludeBinding 1937
IncludeImportedWSDLFiles 1938
IncludeMessages 1938
IncludeOverview 1938
IncludePortType 1939
IncludeService 1939
IncludeTypes 1939

AllDetails 1935
ShowBindingDiagram 1942
ShowExtensibility 1942
ShowMessageParts 1942
ShowPort 1943

© 2016-2022 Altova GmbH

ShowPortTypeDiagram 1943
ShowPortTypeOperations 1943
ShowServiceDiagram 1944
ShowSourceCode 1945
ShowTypesDiagram 1945
ShowUsedBy 1945

Beschreibung
Dieses Objekt kombiniert alle Optionen für die WSDL-Dokumentgenerierung, die über Dialogfelder der Benutzeroberfläche von XMLSpyzur Verfügung stehen. Die Dokumentgenerierungsoptionen werden mit den Werten initialisiert, die bei der letzten Generierung von WSLD-Dokumentation verwendet wurden. Bevor Sie das Objekt jedoch verwenden, müssen Sie dieOutputFile 1940Eigenschaft auf einen gültigen Dateipfad setzen. Mit Hilfe von OptionsDialogAction 1940, OutputFileDialogAction 1940und ShowProgressBar 1944können Sie den Grad der gewünschten Benutzerinteraktion definieren. Mit Hilfe von IncludeAll 1937und AllDetails 1935können Sie ganze Optionsgruppen auf einmal definieren. Oder Sie definieren die Eigenschaften einzeln, um die Feineinstellung vorzunehmen.

29.3.2.29.1 AllDetails

Methode: AllDetails(i_bDetailsOnals Boolean)

Beschreibung
Mit Hilfe dieser Methode können Sie alle detaillierten Optionen aktivieren bzw. deaktivieren.

Fehler
3900 Das Objekt ist nicht mehr gültig.

29.3.2.29.2 Application

Eigenschaft: Applicationals Application 1746(schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.3 CreateDiagramsFolder

Eigenschaft: CreateDiagramsFolderals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um ein Verzeichnis für die erstellten Bilder zu erstellen. Andernfalls werden die Diagramme neben der Dokumentation erstellt. Diese Eigenschaft steht nur zur Verfügung, wenn die Diagramme nicht eingebettet sind. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.4 DiagramFormat

Eigenschaft: DiagramFormatals SPYImageKind[2024]

Beschreibung

Diese Eigenschaft definiert den generierten Diagrammbildtyp. Diese Eigenschaft steht für HTML-Dokumentation nicht zur Verfügung. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Die Standardeinstellung für die erste Ausführung ist PNG.

Fehler

3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.5 EmbedCSSInHTML

Eigenschaft: EmbedCSSInHTMLals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um die CSS-Daten in das generierte HTML-Dokument einzubetten. Andernfalls wird eine separate Datei erstellt und verknüpft. Diese Eigenschaft steht nur für die HTML-Dokumentation zur Verfügung. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true

Fehler

3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.6 EmbedDiagrams

Eigenschaft: EmbedDiagramsals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um das Diagramm in das generierte Dokument einzubetten. Diese Eigenschaft steht für HTML-Dokumentation nicht zur Verfügung. Die Eigenschaft wird mit den Werten

© 2016-2022 Altova GmbH

initialisiert, die beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurden. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.7 GlobalElementsAndTypesOnly

Eigenschaft: GlobalElementsAndTypesOnlyals Boolean

Beschreibung
Gibt einen Wert zurück bzw. definiert einen Wert, der angibt, ob eine komplette Schemadokumentation vorgenommen wird, oder ob nur globale Elemente und Typen dokumentiert werden.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.8 IncludeAll

Methode: IncludeAll(i_bIncludeals Boolean)

Beschreibung
Verwenden Sie diese Methode um alle Include-Optionen auszuwählen, bzw. die Auswahl für alle aufzuheben.

Fehler
3900 Das Objekt ist nicht mehr gültig.

29.3.2.29.9 IncludeBinding

Eigenschaft: IncludeBindingals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Bindings in die WSDL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.10 IncludeImportedWSDLFiles

Eigenschaft: IncludeImportedWSDLFilesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um importierte WSDL-Dateien in die WSDL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.11 IncludeMessages

Eigenschaft: IncludeMessagesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Messages in die WSDL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.12 IncludeOverview

Eigenschaft: IncludeOverviewals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um eine Übersicht in die WSDL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.29.13 IncludePortType

Eigenschaft: IncludePortTypeals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Port Types in die WSDL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.14 IncludeService

Eigenschaft: IncludeServiceals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Services in die WSDL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.15 IncludeTypes

Eigenschaft: IncludeTypesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Typen in die WSDL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.16 MultipleOutputFiles

Eigenschaft: MultipleOutputFilesals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um die Dokumentationsdateien aufzuteilen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist false.

Fehler

3900	Das Objekt ist nicht mehr gültig.
3901	Zum Definieren der Eigenschaft wurde ein ungültiger Wert verwendet. Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.17 OptionsDialogAction

Eigenschaft: OptionsDialogActionals SPYDialogAction[2023]

Beschreibung
Damit Ihr Script die Standardwerte ausfüllen kann und der Benutzer das Dialogfeld sieht und darauf reagieren kann, setzen Sie diese Eigenschaft auf *spyDialogUserInput(2)*. Wenn alle Optionen im Dialogfeld "Schemadokumentation" vom Script definiert werden sollen und keine Interaktion mit dem Benutzer vorgesehen ist, verwenden Sie *spyDialogOK(0)*. Der Standardwert ist *spyDialogOK*.

Fehler

3900	Das Objekt ist nicht mehr gültig.
3901	Zum Definieren der Eigenschaft wurde ein ungültiger Wert verwendet. Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.18 OutputFile

Eigenschaft: OutputFileals String

Beschreibung
Der vollständige Pfad und Name der Datei, die die generierte Dokumentation enthalten soll. Bei einer HTML-Ausgabe werden zusätzliche '.png' Dateien auf Basis dieses Dateinamens generiert. Der Standardwert für diese Eigenschaft ist ein leerer String, der ersetzt werden muss, bevor Sie diesese Objekten in einem Aufruf von Document.GenerateWSDLDocumentation[1857] verwenden.

Fehler

3900	Das Objekt ist nicht mehr gültig.
3901	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.19 OutputFileDialogAction

Eigenschaft: OutputFileDialogActionals SPYDialogAction[2023]

Beschreibung
Damit der Benutzer die Ausgabedatei in einem Dialogfeld zur Dateiauswahl auswählen kann, setzen Sie diese Eigenschaft auf *spyDialogUserInput(2)*. Wenn der in OutputFile[1940] gespeicherte Wert verwendet werden soll und keinen Interaktion mit dem Benutzer vorgesehen ist, verwenden Sie *spyDialogOK(0)*. Der Standardwert ist *spyDialogOK*.

© 2016-2022 Altova GmbH

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Zum Definieren der Eigenschaft wurde ein ungültiger Wert verwendet.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.20 OutputFormat

Eigenschaft: OutputFormatals SPYSchemaDocumentationFormat[2030]

Beschreibung
Definiert, welche Art von Dokumentation generiert werden soll: HTML (Wert=`0`), MS-Word (Wert=`1`) oder RTF (Werte=`2`). Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist HTML.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Zum Definieren der Eigenschaft wurde ein ungültiger Wert verwendet.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.21 Parent

Eigenschaft: Parentals Dialogs[1835](schreibgeschützt)

Beschreibung
Ruft das dem Objekt übergeordnete Objekt auf.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.22 SeparateSchemaDocument

Eigenschaft: SeparateSchemaDocumentals Boolean

Beschreibung
Gibt einen Wert zurück bzw. definiert einen Wert, der angibt, ob die Schemadokumentation in ein separates Dokument geschrieben werden soll.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.23 ShowBindingDiagram

Eigenschaft: ShowBindingDiagramals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Binding-Diagramme in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.24 ShowExtensibility

Eigenschaft: ShowExtensibilityals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Service- und Binding-Erweiterungen der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.25 ShowMessageParts

Eigenschaft: ShowMessagePartsals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Message-Teile von Messagesn der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.29.26 ShowPort

Eigenschaft: ShowPortals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Service Ports in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.27 ShowPortTypeDiagram

Eigenschaft: ShowPortTypeDiagramals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Port Type-Diagramme in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.28 ShowPortTypeOperations

Eigenschaft: ShowPortTypeOperationsas Boolean

Beschreibung
Set this property to true, to show port type operations in the WSDL documentation. The property is initialized with the value used during the last call to Document.GenerateWSDLDocumentation[1857]. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.29 ShowProgressBar

Eigenschaft: ShowProgressBarals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um das Fenster, in dem die Fortschrittsanzeige für die Dokumentgeneration angezeigt wird, sichtbar zu machen. Mit falsewird es ausgeblendet. Die Standardeinstellung ist false.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.30 ShowResult

Eigenschaft: ShowResultals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um das erzeugte Dokument automatisch zu öffnen, wenn die Generierung erfolgreich abgeschlossen wurde. HTML-Dokumentation wird in XMLSpygeöffnet. Um Word-Dokumentation anzuzeigen, wird MS-Word gestartet. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.31 ShowServiceDiagram

Eigenschaft: ShowServiceDiagramas Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Service-Diagramme in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.29.32 ShowSourceCode

Eigenschaft: ShowSourceCodeas Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Quellcode für die Includes in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.33 ShowTypesDiagram

Eigenschaft: ShowTypesDiagramas Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Typ-Diagramme in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.34 ShowUsedBy

Eigenschaft: ShowUsedByas Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um used-by-Beziehungen für Types, Bindings und Message-Definitionen in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.35 UseFixedDesign

Eigenschaft: UseFixedDesignals Boolean

Beschreibung
Gibt an, on die Dokumentation mit einem festgelegten Design oder mit einem Design erstellt werden soll, das durch eine SPS-Datei festgelegt wird (wobei StyleVision benötigt wird).

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.29.36 SPSFile

Eigenschaft: SPSFileals String

Beschreibung
Vollständiger Pfad der SPS-Datei, die zur Generierung der Dokumentation verwendet wird.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30 WSDL20DocumentationDlg

Eigenschaften und Methoden

Standard-Automation-Eigenschaften
Application 1948
Parent 1953

Eigenschaften zur Interaktion und Sichtbarkeit
OptionsDialogAction 1952
OutputFile 1952
OutputFileDialogAction 1953
ShowProgressBar 1956
ShowResult 1956

Optionen und Methoden zur Dokumentgenerierung
OutputFormat 1953
UseFixedDesign 1958
SPSFile 1958
EmbedDiagrams 1949
DiagramFormat 1948
MultipleOutputFiles 1952
EmbedCSSInHTML 1948

© 2016-2022 Altova GmbH

CreateDiagramsFolder 1948

IncludeAll 1949
IncludeBinding 1950
IncludeImportedWSDLFiles 1950
IncludeInterface 1950
IncludeOverview 1951
IncludeService 1951
IncludeTypes 1951

AllDetails 1947
ShowBindingDiagram 1954
ShowExtensibility 1954
ShowEndpoint 1954
ShowFault 1955
ShowInterfaceDiagram 1955
ShowOperation 1955
ShowServiceDiagram 1956
ShowSourceCode 1957
ShowTypesDiagram 1957
ShowUsedBy 1957

Beschreibung
Dieses Objekt kombiniert alle Optionen für die WSDL-Dokumentgenerierung, die über Dialogfelder der Benutzeroberfläche von XMLSpyzur Verfügung stehen. Die Dokumentgenerierungsoptionen werden mit den Werten initialisiert, die bei der letzten Generierung von WSLD-Dokumentation verwendet wurden. Bevor Sie das Objekt jedoch verwenden, müssen Sie dieOutputFile 1952Eigenschaft auf einen gültigen Dateipfad setzen. Mit Hilfe von OptionsDialogAction 1952, OutputFileDialogAction 1953und ShowProgressBar 1956können Sie den Grad der gewünschten Benutzerinteraktion definieren. Mit Hilfe von IncludeAll 1949und AllDetails 1947können Sie ganze Optionsgruppen auf einmal definieren. Oder Sie definieren die Eigenschaften einzeln, um die Feineinstellung vorzunehmen.

29.3.2.30.1 AllDetails

Methode: AllDetails(i_bDetailsOnals Boolean)

Beschreibung
Mit Hilfe dieser Methode können Sie alle detaillierten Optionen aktivieren bzw. deaktivieren.

Fehler
4300 Das Objekt ist nicht mehr gültig.

29.3.2.30.2 Application

Eigenschaft: Applicationals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.3 CreateDiagramsFolder

Eigenschaft: CreateDiagramsFolderals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um ein Verzeichnis für die erstellten Bilder zu erstellen. Andernfalls werden die Diagramme neben der Dokumentation erstellt. Diese Eigenschaft steht nur zur Verfügung, wenn die Diagramme nicht eingebettet sind. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.4 DiagramFormat

Eigenschaft: DiagramFormatals SPYImageKind[2024]

Beschreibung
Diese Eigenschaft definiert den generierten Diagrammbildtyp. Diese Eigenschaft steht für HTML-Dokumentation nicht zur Verfügung. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856]verwendet wurde. Die Standardeinstellung für die erste Ausführung ist PNG.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.5 EmbedCSSInHTML

Eigenschaft: EmbedCSSInHTMLals Boolean

Beschreibung

© 2016-2022 Altova GmbH

Setzen Sie diese Eigenschaft auf true, um die CSS-Daten in das generierte HTML-Dokument einzubetten. Andernfalls wird eine separate Datei erstellt und verknüpft. Diese Eigenschaft steht nur für die HTML-Dokumentation zur Verfügung. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.6 EmbedDiagrams

Eigenschaft: EmbedDiagramsals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um das Diagramm in das generierte Dokument einzubetten. Diese Eigenschaft steht für HTML-Dokumentation nicht zur Verfügung. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.7 GlobalElementsAndTypesOnly

Eigenschaft: GlobalElementsAndTypesOnlyals Boolean

Beschreibung
Gibt einen Wert zurück bzw. definiert einen Wert, der angibt, ob eine komplette Schemadokumentation vorgenommen wird, oder ob nur globale Elemente und Typen dokumentiert werden.

Fehler
3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.8 IncludeAll

Methode: IncludeAll(i_bIncludeals Boolean)

Beschreibung
Verwenden Sie diese Methode um alle Include-Optionen auszuwählen, bzw. die Auswahl für alle aufzuheben.

Fehler
4300 Das Objekt ist nicht mehr gültig.

29.3.2.30.9 IncludeBinding

Eigenschaft: IncludeBindingals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Bindings in die WSDL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.10 IncludeImportedWSDLFiles

Eigenschaft: IncludeImportedWSDLFilesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um importierte WSDL-Dateien in die WSDL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.11 IncludeInterface

Eigenschaft: IncludeInterfaceals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Interfaces in die WSDL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.30.12 IncludeOverview

Eigenschaft: IncludeOverviewals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um eine Übersicht in die WSDL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.13 IncludeService

Eigenschaft: IncludeServiceals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Services in die WSDL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.14 IncludeTypes

Eigenschaft: IncludeTypesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Types in die WSDL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.15 MultipleOutputFiles

Eigenschaft: MultipleOutputFilesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um die Dokumentationsdateien aufzuteilen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist false.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Zum Definieren der Eigenschaft wurde ein ungültiger Wert verwendet.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.16 OptionsDialogAction

Eigenschaft: OptionsDialogActionals SPYDialogAction[2023]

Beschreibung
Damit Ihr Script die Standardwerte ausfüllen kann und der Benutzer das Dialogfeld sieht und darauf reagieren kann, setzen Sie diese Eigenschaft auf *spyDialogUserInput(2)*. Wenn alle Optionen im Dialogfeld "Schemadokumentation" vom Script definiert werden sollen und keine Interaktion mit dem Benutzer vorgesehen ist, verwenden Sie *spyDialogOK(0)*. Der Standardwert ist *spyDialogOK*.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Zum Definieren der Eigenschaft wurde ein ungültiger Wert verwendet.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.17 OutputFile

Eigenschaft: OutputFileals String

Beschreibung
Der vollständige Pfad und Name der Datei, die die generierte Dokumentation enthalten soll. Bei einer HTML-Ausgabe werden zusätzliche '.png' Dateien auf Basis dieses Dateinamens generiert. Der Standardwert für diese Eigenschaft ist ein leerer String, der ersetzt werden muss, bevor Sie diesese Objekten in einem Aufruf von Document.GenerateWSDL20Documentation[1856] verwenden.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.30.18 OutputFileDialogAction

Eigenschaft: OutputFileDialogActionals SPYDialogAction[2023]

Beschreibung
Damit der Benutzer die Ausgabedatei in einem Dialogfeld zur Dateiauswahl auswählen kann, setzen Sie diese Eigenschaft auf *spyDialogUserInput(2)*. Wenn der in OutputFile[1952] gespeicherte Wert verwendet werden soll und keinen Interaktion mit dem Benutzer vorgesehen ist, verwenden Sie *spyDialogOK(0)*. Der Standardwert ist *spyDialogOK*.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Zum Definieren der Eigenschaft wurde ein ungültiger Wert verwendet.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.19 OutputFormat

Eigenschaft: OutputFormatals SPYSchemaDocumentationFormat[2030]

Beschreibung
Definiert, welche Art von Dokumentation generiert werden soll: HTML (Wert=`0`), MS-Word (Wert=`1`) oder RTF (Werte=`2`). Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist HTML.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Zum Definieren der Eigenschaft wurde ein ungültiger Wert verwendet.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.20 Parent

Eigenschaft: Parentals Dialogs[1835](schreibgeschützt)

Beschreibung
Ruft das dem Objekt übergeordnete Objekt auf.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.21 SeparateSchemaDocument

Eigenschaft: SeparateSchemaDocumentals Boolean

Beschreibung

Gibt einen Wert zurück bzw. definiert einen Wert, der angibt, ob die Schemadokumentation in ein separates Dokument geschrieben werden soll.

Fehler

3900 Das Objekt ist nicht mehr gültig.
3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.22 ShowBindingDiagram

Eigenschaft: ShowBindingDiagramals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um Binding-Diagramme in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.23 ShowEndpoint

Eigenschaft: ShowEndpointals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um Sercice Endpoints in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.24 ShowExtensibility

Eigenschaft: ShowExtensibilityals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um Service- und Binding-Erweiterungen der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation 1856 verwendet wurde. Der Standardwert für die erste Ausführung ist true.

© 2016-2022 Altova GmbH

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.25 ShowFault

Eigenschaft: ShowFaultals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Fault-Elemente in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.26 ShowInterfaceDiagram

Eigenschaft: ShowInterfaceDiagramals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Interface-Diagramme in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.27 ShowOperation

Eigenschaft: ShowOperationals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Interface- und Binding-Operationen in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.28 ShowProgressBar

Eigenschaft: ShowProgressBarals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, damit das Fenster mit der Fortschrittsanzeige für die Dokumentgenerierung angezeigt wird. Verwenden Sie `false`, um es auszublenden. Der Standardwert für ist `false`.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.29 ShowResult

Eigenschaft: ShowResultals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um das erzeugte Dokument automatisch zu öffnen, wenn die Generierung erfolgreich abgeschlossen wurde. HTML-Dokumentation wird in XMLSpygeöffnet. Um Word-Dokumentation anzuzeigen, wird MS-Word gestartet. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.30 ShowServiceDiagram

Eigenschaft: ShowServiceDiagramas Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Service-Diagramme in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.30.31 ShowSourceCode

Eigenschaft: ShowSourceCodeas Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Quellcode für die Includes in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.32 ShowTypesDiagram

Eigenschaft: ShowTypesDiagramas Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Typ-Diagramme in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.33 ShowUsedBy

Eigenschaft: ShowUsedByas Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um used-by-Beziehungen für Types, Bindings und Message-Definitionen in der WSDL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateWSDL20Documentation[1856] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.34 SPSFile

Eigenschaft: SPSFileals String

Beschreibung
Vollständiger Pfad der SPS-Datei, die zur Generierung der Dokumentation verwendet wird.

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.30.35 UseFixedDesign

Eigenschaft: UseFixedDesignals Boolean

Beschreibung
Gibt an, on die Dokumentation mit einem festgelegten Design oder mit einem Design erstellt werden soll, das durch eine SPS-Datei festgelegt wird (wobei StyleVision benötigt wird).

Fehler
4300 Das Objekt ist nicht mehr gültig.
4301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31 XBRLDocumentationDlg

Eigenschaften und Methoden

Standard-Automation-Eigenschaften
Application 1960
Parent 1965

Eigenschaften zur Interaktion und Sichtbarkeit
OptionsDialogAction 1963
OutputFile 1964
OutputFileDialogAction 1964
ShowProgressBar 1968
ShowResult 1968

Optionen und Methoden zur Dokumentgenerierung
OutputFormat 1964
EmbedDiagrams 1961
DiagramFormat 1960
EmbedCSSInHTML 1960
CreateDiagramsFolder 1960

IncludeAll 1961

© 2016-2022 Altova GmbH

IncludeOverview 1963
IncludeNamespacePrefixes 1962
IncludeGlobalElements 1962
IncludeDefinitionLinkroles 1962
IncludePresentationLinkroles 1963
IncludeCalculationLinkroles 1961

AllDetails 1959
ShowDiagram 1966
ShowSubstitutiongroup 1969
ShowItemtype 1966
ShowBalance 1965
ShowPeriod 1968
ShowAbstract 1965
ShowNillable 1967
ShowLabels 1967
ShowReferences 1968
ShowLinkbaseReferences 1967

ShortQualifiedName 1965
ShowImportedElements 1966

Beschreibung
Dieses Objekt kombiniert alle Optionen für die XBRL-Dokumentgenerierung, die über Dialogfelder der Benutzeroberfläche von XMLSpyzur Verfügung stehen. Die Dokumentgenerierungsoptionen werden mit den Werten initialisiert, die bei der letzten Generierung von WSLD-Dokumentation verwendet wurden. Bevor Sie das Objekt jedoch verwenden, müssen Sie dieOutputFile 1964Eigenschaft auf einen gültigen Dateipfad setzen. Mit Hilfe von OptionsDialogAction 1963, OutputFileDialogAction 1964und ShowProgressBar 1968können Sie den Grad der gewünschten Benutzerinteraktion definieren. Mit Hilfe von IncludeAll 1961und AllDetails 1959können Sie ganze Optionsgruppen auf einmal definieren. Oder Sie definieren die Eigenschaften einzeln, um die Feineinstellung vorzunehmen.

29.3.2.31.1 AllDetails

Methode: AllDetails(i_bDetailsOnals Boolean)

Beschreibung
Mit Hilfe dieser Methode können Sie alle detaillierten Optionen aktivieren bzw. deaktivieren.

Fehler
4400 Das Objekt ist nicht mehr gültig.

29.3.2.31.2 Application

Eigenschaft: Applicationals Application[1746](schreibgeschützt)

Beschreibung
Ruft das XMLSpyapplication-Objekt auf.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.3 CreateDiagramsFolder

Eigenschaft: CreateDiagramsFolderals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um ein Verzeichnis für die erstellten Bilder zu erstellen. Andernfalls werden die Diagramme neben der Dokumentation erstellt. Diese Eigenschaft steht nur zur Verfügung, wenn die Diagramme nicht eingebettet sind. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857]verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.4 DiagramFormat

Eigenschaft: DiagramFormatals SPYImageKind[2024]

Beschreibung
Diese Eigenschaft definiert den generierten Diagrammbildtyp. Diese Eigenschaft steht für HTML-Dokumentation nicht zur Verfügung. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857]verwendet wurde. Die Standardeinstellung für die erste Ausführung ist PNG.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.5 EmbedCSSInHTML

Eigenschaft: EmbedCSSInHTMLals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um die CSS-Daten in das generierte HTML-Dokument einzubetten. Andernfalls wird eine separate Datei erstellt und verknüpft. Diese Eigenschaft steht nur für die HTML-

© 2016-2022 Altova GmbH

Dokumentation zur Verfügung. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.6 EmbedDiagrams

Eigenschaft: EmbedDiagramsals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um das Diagramm in das generierte Dokument einzubetten. Diese Eigenschaft steht für HTML-Dokumentation nicht zur Verfügung. Die Eigenschaft wird mit den Werten initialisiert, die beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurden. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.7 IncludeAll

Methode: IncludeAll(i_bIncludeals Boolean)

Beschreibung
Verwenden Sie diese Methode um alle Include-Optionen auszuwählen, bzw. die Auswahl für alle aufzuheben.

Fehler
4400 Das Objekt ist nicht mehr gültig.

29.3.2.31.8 IncludeCalculationLinkroles

Eigenschaft: IncludeCalculationLinkrolesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Calculation Linkroles in die XBRL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.9 IncludeDefinitionLinkroles

Eigenschaft: IncludeDefinitionLinkrolesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Definition Linkroles in die XBRL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.10 IncludeGlobalElements

Eigenschaft: IncludeGlobalElementsals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um globale Elemente in die XBRL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.11 IncludeNamespacePrefixes

Eigenschaft: IncludeNamespacePrefixesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Namespace-Präfixe in die XBRL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.31.12 IncludeOverview

Eigenschaft: IncludeOverviewals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um eine Übersicht in die XBRL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.13 IncludePresentationLinkroles

Eigenschaft: IncludePresentationLinkrolesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Presentation Linkroles in die XBRL-Dokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.14 OptionsDialogAction

Eigenschaft: OptionsDialogActionals SPYDialogAction[2023]

Beschreibung
Damit Ihr Script die Standardwerte ausfüllen kann und der Benutzer das Dialogfeld sieht und darauf reagieren kann, setzen Sie diese Eigenschaft auf *spyDialogUserInput(2)*. Wenn alle Optionen im Dialogfeld "Schemadokumentation" vom Script definiert werden sollen und keine Interaktion mit dem Benutzer vorgesehen ist, verwenden Sie *spyDialogOK(0)*. Der Standardwert ist *spyDialogOK*.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Zum Definieren der Eigenschaft wurde ein ungültiger Wert verwendet.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.15 OutputFile

Eigenschaft: OutputFileals String

Beschreibung
Der vollständige Pfad und Name der Datei, die die generierte Dokumentation enthalten soll. Bei einer HTML-Ausgabe werden zusätzliche '.png' Dateien auf Basis dieses Dateinamens generiert. Der Standardwert für diese Eigenschaft ist ein leerer String, der ersetzt werden muss, bevor Sie diesese Objekten in einem Aufruf von Document.GenerateXBRLDocumentation[1857] verwenden.

Fehler

4400	Das Objekt ist nicht mehr gültig.
4401	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.16 OutputFileDialogAction

Eigenschaft: OutputFileDialogActionals SPYDialogAction[2023]

Beschreibung
Damit der Benutzer die Ausgabedatei in einem Dialogfeld zur Dateiauswahl auswählen kann, setzen Sie diese Eigenschaft auf *spyDialogUserInput(2)*. Wenn der in OutputFile[1964] gespeicherte Wert verwendet werden soll und keinen Interaktion mit dem Benutzer vorgesehen ist, verwenden Sie *spyDialogOK(0)*. Der Standardwert ist *spyDialogOK*.

Fehler

4400	Das Objekt ist nicht mehr gültig.
4401	Zum Definieren der Eigenschaft wurde ein ungültiger Wert verwendet. Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.17 OutputFormat

Eigenschaft: OutputFormatals SPYSchemaDocumentationFormat[2030]

Beschreibung
Definiert, welche Art von Dokumentation generiert werden soll: HTML (Wert=0), MS-Word (Wert=1) oder RTF (Werte=2). Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist HTML.

Fehler

4400	Das Objekt ist nicht mehr gültig.
4401	Zum Definieren der Eigenschaft wurde ein ungültiger Wert verwendet. Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.31.18 Parent

Eigenschaft: Parentals Dialogs[1835](schreibgeschützt)

Beschreibung
Ruft das dem Objekt übergeordnete Objekt auf.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.19 ShortQualifiedName

Eigenschaft: ShortQualifiedNameals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um in der XBRL-Dokumentation kurze qualifizierte Namen zu verwenden. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857]verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.20 ShowAbstract

Eigenschaft: ShowAbstractals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um abstrakte Elemente in der XBRL-Dokumentation zu verwenden. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857]verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.21 ShowBalance

Eigenschaft: ShowBalanceals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um balance-Elemente in der XBRL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von

Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.22 ShowDiagram

Eigenschaft: ShowDiagramals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um Diagramme in der XBRL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.23 ShowImportedElements

Eigenschaft: ShowImportedElementsals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um importierte Elemente in der XBRL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.24 ShowItemtype

Eigenschaft: ShowItemtypeals Boolean

Beschreibung

Setzen Sie diese Eigenschaft auf true, um Item Types in der XBRL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler

© 2016-2022 Altova GmbH

4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.25 ShowLabels

Eigenschaft: ShowLabelsals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Labels in der XBRL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.26 ShowLinkbaseReferences

Eigenschaft: ShowLinkbaseReferencesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Linkbase-Referenzen in der XBRL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.27 ShowNillable

Eigenschaft: ShowNillableals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um nillable-Eigenschaften in der XBRL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.28 ShowPeriod

Eigenschaft: ShowPeriodals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um periods (Zeitdauer) in der XBRL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.29 ShowProgressBar

Eigenschaft: ShowProgressBarals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um das Fenster, in dem die Fortschrittsanzeige für die Dokumentgeneration angezeigt wird, sichtbar zu machen. Mit falsewird es ausgeblendet. Die Standardeinstellung ist false.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.30 ShowReferences

Eigenschaft: ShowReferencesals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Referenzen in der XBRL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.31 ShowResult

Eigenschaft: ShowResultals Boolean

Beschreibung

© 2016-2022 Altova GmbH

Setzen Sie diese Eigenschaft auf true, um das erzeugte Dokument automatisch zu öffnen, wenn die Generierung erfolgreich abgeschlossen wurde. HTML-Dokumentation wird in XMLSpygeöffnet. Um Word-Dokumentation anzuzeigen, wird MS-Word gestartet. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.32 ShowSubstitutiongroup

Eigenschaft: ShowSubstitutiongroupals Boolean

Beschreibung
Setzen Sie diese Eigenschaft auf true, um Substitution Groups in der XBRL-Dokumentation anzuzeigen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von Document.GenerateXBRLDocumentation[1857] verwendet wurde. Der Standardwert für die erste Ausführung ist true.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.33 SPSFile

Eigenschaft: SPSFileals String

Beschreibung
Vollständiger Pfad der SPS-Datei, die zur Generierung der Dokumentation verwendet wird.

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.31.34 UseFixedDesign

Eigenschaft: UseFixedDesignals Boolean

Beschreibung
Gibt an, on die Dokumentation mit einem festgelegten Design oder mit einem Design erstellt werden soll, das durch eine SPS-Datei festgelegt wird (wobei StyleVision benötigt wird).

Fehler
4400 Das Objekt ist nicht mehr gültig.
4401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.32 XMLData

Eigenschaften
Kind 1980
Name 1980
TextValue 1981

HasChildren 1978
MayHaveChildren 1980
Parent 1980

Methoden
GetFirstChild 1975
GetNextChild 1976
GetCurrentChild 1975

InsertChild 1978
InsertChildAfter 1979
InsertChildBefore 1979
AppendChild 1971

EraseAllChildren 1972
EraseChild 1973
EraseCurrentChild 1973

IsSameNode 1979

CountChildren 1972
CountChildrenKind 1972

GetChild 1974
GetChildAttribute 1974
GetChildElement 1974
GetChildKind 1975
GetNamespacePrefixForURI 1976

HasChildrenKind 1978
SetTextValueXMLEncoded 1981

Beschreibung
Über die XMLData-Schnittstelle haben Sie auf XML-Ebene direkten Zugriff auf das Dokument. Sie können die XML-Darstellung des Dokuments anzeigen und direkt bearbeiten. Beachten Sie jedoch die folgenden Einschränkungen:

- Die XMLData-Darstellung ist nur gültig, wenn das Dokument in der Grid-Ansicht oder der Authentic-Ansicht angezeigt wird.
- In der Authentic-Ansicht werden zusätzliche XMLData-Elemente automatisch als übergeordnete Elemente der einzelnen sichtbaren Dokumentelemente eingefügt. Normalerweise handelt es sich dabei um ein XMLData-Element von der Art spyXMLDataElement,wobei die Name 1980 Eigenschaft auf 'Text' gesetzt ist.

© 2016-2022 Altova GmbH

- Wenn Sie die XMLData-Schnittstelle von einem anderen Ansichtsmodus aus verwenden, werden zwar keine Fehler angezeigt, aber die Änderungen werden in der Ansicht nicht angezeigt und gehen u.U. beim nächsten Ansichtswechsel verloren.

Beachten Sie bitte außerdem:

- Das Definieren eines neuen Textwerts für ein XML-Element ist möglich, wenn das Element keine Unterelemente hat, die keinen Text enthalten. Ein Textwert kann sogar definiert werden, wenn das Element Attribute enthält.
- Wenn Sie für ein XML-Element, das mehr als ein Text-Child-Element hat, einen neuen Textwert definieren, so wird das letzte davon gelöscht und durch ein neues Text-Child-Element ersetzt.
- Wenn der Textwert eines XML-Elements mit mehr als einem Text-Child-Element gelesen wird, so wird nur der Wert des ersten Text-Child-Elements zurückgegeben.

Objekte dieser Klasse repräsentieren die unterschiedlichen Bestandteile eines XML-Dokuments. Informationen zu den verfügbaren Teiltypen finden Sie unter dem Enumerationstyp SPYXMLDataKind[2033].Jeder Teil kennt seine untergeordneten Elemente, wodurch eine XMLData-Struktur mit Document.RootElement[1863]auf der obersten Ebene gebildet wird. Um das oberste Element des Dokumentinhalts (ohne XML Header) abzurufen, verwenden Sie Document.DataRoot[1849]. Beispiele, wie man die XMLData-Struktur durchquert, finden Sie unter GetNextChild[1976].

29.3.2.32.1 AppendChild

Deklaration: AppendChild (*pNewData* als XMLData[1970])

Beschreibung
AppendChildhängt pNewDataals letztes Child-Element an das XMLData-Objekt an.

Fehler

1500	Das XMLData-Element ist nicht mehr gültig.
1505	Es wurde eine ungültige XMLData-Art definiert.
1506	Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
1507	Das Element kann keine untergeordneten Elemente haben.
1512	Zyklische Einfügung - ein neues Datenelement ist bereits Teil des Dokuments.
1514	Es wurde eine ungülte XMLData-Art für diese Position definiert.
1900	Das Dokument darf nicht geändert werden.

Beispiel

```
DimobjCurrentParent AsXMLData
DimobjNewChild AsXMLData

SetobjNewChild = objSpy.ActiveDocument.CreateChild(spyXMLDataElement)
SetobjCurrentParent = objSpy.ActiveDocument.RootElement

objCurrentParent.AppendChild objNewChild

SetobjNewChild = Nothing
```

29.3.2.32.2 CountChildren

Deklaration: CountChildren als long

Beschreibung
CountChildrenruft die Anzahl der Child-Elemente ab.

Verfügbar mit Typbibliothek Version 1.5

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.

29.3.2.32.3 CountChildrenKind

Deklaration: CountChildrenKind (*nKind* als SPYXMLDataKind[2033]) als long

Beschreibung
CountChildrenKindruft die Anzahl der Child-Elemente dieser bestimmten Art ab.

Verfügbar mit Typbibliothek Version 1.5

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.

29.3.2.32.4 EraseAllChildren

Deklaration: EraseAllChildren

Beschreibung
EraseAllChildrenlöscht alle zum XMLDataObjekt gehörigen Child-Elemente.

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1900 Das Dokument darf nicht geändert werden.

Beispiel
In diesem Beispiel werden alle Elemente des aktiven Dokuments gelöscht.

```
DimobjCurrentParent AsXMLData

SetobjCurrentParent = objSpy.ActiveDocument.RootElement
objCurrentParent.EraseAllChildren
```


© 2016-2022 Altova GmbH

29.3.2.32.5 EraseChild

Methode: EraseChild (Child als XMLData[1970])

Beschreibung
Löscht den angegebenen Child Node.

Fehler
1500 Ungültiges Objekt.
1506 Ungültige XML-Input-Datei.
1510 Ungültiger Parameter.

29.3.2.32.6 EraseCurrentChild

Deklaration: EraseCurrentChild

Beschreibung
EraseCurrentChildlöscht das aktuelle XMLDataChild-Objekt. Bevor Sie EraseCurrentChildaufrufen, müssen Sie einen internen Iterator mit XMLData.GetFirstChild[1975]initialisieren. Nachdem Sie das aktuelle Child-Element gelöscht haben, erhöht EraseCurrentChild den internen Iterator des XMLData-Elements. Es wird kein Fehler zurückgegeben, wenn das letzte Child-Element gelöscht wird und der Iterator über das Ende der Child-Liste hinaus verschoben wird. Der nächste Aufruf von EraseCurrentChildgibt allerdings den Fehler 1503 zurück.

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1503 Es wurde für dieses XMLData-Objekt kein Iterator initialisiert oder der Iterator zeigt hinter das letzte Child-Element.
1900 Das Dokument darf nicht geändert werden.

Beispiele

```
// ------------------------------------
// Scripting environment - JScript
// erase all children of XMLData
// ------------------------------------
// let's get an XMLData element, we assume that the
// cursor selects the parent of a list in grid view
var objList = Application.ActiveDocument.GridView.CurrentFocus;

// the following line would be shorter, of course
//        objList.EraseAllChildren ();

// but we want to demonstrate the usage of EraseCurrentChild
if((objList != null) && (objList.HasChildren))
{
        try
        {
                objEle = objList.GetFirstChild(-1);
                while(objEle != null)
                        objList.EraseCurrentChild();
```

```
                    // no need to call GetNextChild
        }
        catch(err)
            // 1503 - we reached end of child list
            { if((err.number & 0xffff) != 1503) throw(err); }
}
```

29.3.2.32.7 GetChild

Deklaration: GetChild (*position* als long) als XMLData[1970]

Rückgabewert
Gibt ein XML-Element als XMLDataObjekt zurück.

Beschreibung
GetChild() gibt eine Referenz auf das Child am angegebenen Index (nullbasiert) zurück.

Verfügbar mit Typbibliothek Version 1.5

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.32.8 GetChildAttribute

Methode: GetChildAttribute (strName als string) Child als XMLData Objekt (`NULL` bei Fehler)

Beschreibung
Ruft das Attribut mit dem angegebenen Namen auf.

Fehler
1500 Ungültiges Objekt.
1510 Ungültiger Parameter.

29.3.2.32.9 GetChildElement

Methode: GetChildElement (strName als string, nIndex als long) Child als XMLData-Objekt (`NULL` bei Fehler)

Beschreibung
Ruft das nte Child Element des angegebenen Namens auf.

Fehler
1500 Ungültiges Objekt.
1510 Ungültiger Parameter.

© 2016-2022 Altova GmbH

29.3.2.32.10 GetChildKind

Deklaration: GetChildKind (*position* als long, *nKind* als SPYXMLDataKind[2033]) als XMLData[1970]

Rückgabewert
Gibt ein XML-Element als XMLDataObjekt zurück.

Beschreibung
GetChildKind() gibt eine Referenz auf ein Child dieser Art am angegebenen Index (nullbasiert) zurück. Der Positionsparameter ist relativ zur Anzahl der Child-Elemente der angegebenen Art und nicht zu allen Child-Elementen des Objekts.

Verfügbar mit Typbibliothek Version 1.5

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.32.11 GetCurrentChild

Deklaration: GetCurrentChild als XMLData[1970]

Rückgabewert
Gibt ein XML-Element als XMLDataObjekt zurück.

Beschreibung
GetCurrentChildruft das aktuelle Child-Element ab. Bevor Sie GetCurrentChildaufrufen, müssen Sie einen internen Iterator mit XMLData.GetFirstChild[1975]initialisieren.

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1503 Es wurde für dieses XMLData-Objekt kein Iterator initialisiert.
1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.32.12 GetFirstChild

Deklaration: GetFirstChild (*nKind* als SPYXMLDataKind[2033]) als XMLData[1970]

Rückgabewert
Gibt ein XML-Element als XMLDataObjekt zurück.

Beschreibung

GetFirstChildinitialisiert einen neuen Iterator und gibt das erste Child-Element zurück. Setzen Sie nKind= -1, um einen Iterator für alle Arten von Child-Elementen abzurufen.
Anmerkung: Der Iterator wird innerhalb des XMLData-Objekts gespeichert und wird zerstört, wenn das XMLData-Objekt zerstört wird. Stellen Sie sicher, dass Sie eine Referenz auf dieses Objekt beibehalten, solange Sie GetCurrentChild[1975], GetNextChild[1976]oder EraseCurrentChild[1973]verwenden möchten.

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1501 Es wurde eine ungültige XMLData-Art definiert.
1504 Das Element hat keine untergeordneten Elemente der angegebenen Art.
1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiel
Siehe Beispiel unter XMLData.GetNextChild[1976].

29.3.2.32.13 GetNamespacePrefixForURI

Methode: GetNamespacePrefixForURI (strURI als string) strNS als string

Beschreibung
Gibt das Namespace-Präfix der vorgegebenen URI zurück.

Fehler
1500 Ungültiges Objekt.
1510 Ungültiger Parameter.

29.3.2.32.14 GetNextChild

Deklaration: GetNextChild als XMLData[1970]

Rückgabewert
Gibt ein XML-Element als XMLDataObjekt zurück.

Beschreibung
GetNextChildgeht zum nächsten Child-Element dieses Elements. Bevor Sie GetNextChildaufrufen, müssen Sie einen internen Iterator mit XMLData.GetFirstChild[1975]initialisieren.

Überprüfen Sie das letzte Child-Element des Elements, wie im Beispiel unten gezeigt.
Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1503 Für dieses XMLData-Objekt wurde kein Iterator initialisiert.
1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele
```
' ----------------------------------------
' VBA code snippet - iterate XMLData children
' ----------------------------------------
OnErrorResume Next
```

 © 2016-2022 Altova GmbH

```
SetobjParent = objSpy.ActiveDocument.RootElement

'get elements of all kinds
SetobjCurrentChild = objParent.GetFirstChild(-1)

Do
	'do something useful with the child

	'step to next child
	SetobjCurrentChild = objParent.GetNextChild
LoopUntil(Err.Number - vbObjectError = 1503)
```

```
// ---------------------------------------
// Scripting environment - JScript
// iterate through children of XMLData
// ---------------------------------------
try
{
	var objXMLData = ... // initialize somehow
	var objChild = objXMLData.GetFirstChild(-1);

	while(true)
	{
		// do something usefull with objChild

		objChild = objXMLData.GetNextChild();
	}
}
catch(err)
{
	if((err.number & 0xffff) == 1504)
		; // element has no children
	elseif((err.number & 0xffff) == 1503)
		;		// last child reached
	else
		throw(err);
}
```

29.3.2.32.15 GetTextValueXMLDecoded

Methode: GetTextValueXMLDecoded () als string

Beschreibung
Ruft den dekodierten Textwert der XML-Datei ab.

Fehler

1500	Ungültiges Objekt.
1510	Ungültiger Parameter.

29.3.2.32.16 HasChildren

Deklaration: HasChildren als Boolean

Beschreibung
Die Eigenschaft ist true, wenn das Objekt das übergeordnete Objekt von anderen XMLDataObjekten ist. Diese Eigenschaft ist schreibgeschützt.

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.32.17 HasChildrenKind

Deklaration: HasChildrenKind (*nKind* als SPYXMLDataKind[2033]) als Boolean

Beschreibung
Die Methode gibt true zurück, wenn das Objekt das übergeordnete Objekt von anderen XMLDataObjekten dieser Art ist.

Verfügbar mit Typbibliothek Version 1.5

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.32.18 InsertChild

Deklaration: InsertChild(*pNewData* als XMLData[1970])

Beschreibung
InsertChildfügt das neue Child-Element vor dem aktuellen Child-Element ein (Informationen zum Setzen des aktuellen Child-Elements finden Sie auch unter XMLData.GetFirstChild[1975], XMLData.GetNextChild[1976]).

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1503 Es wurde für dieses XMLData-Objekt kein Iterator initialisiert.
1505 Es wurde eine ungültige XMLData-Art definiert.
1506 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
1507 Das Element kann keine Child-Elemente haben.
1512 Zyklische Einfügung - ein neues Datenelement ist bereits Teil des Dokuments.
1514 Es wurde eine ungültige XMLData-Art für diese Position definiert.
1900 Das Dokument darf nicht geändert werden.

© 2016-2022 Altova GmbH

29.3.2.32.19 InsertChildAfter

Methode: InsertChildAfter (Node als XMLData, NewData als XMLData)

Beschreibung
Fügt einen neuen XML-Node (definiert im zweiten Parameter) nach dem (mit dem ersten Parameter) angegebenen Node ein.

Errors
1500 Ungültiges Objekt.
1506 Ungültige XML-Input-Datei
1507 Es sind keine Child-Nodes zulässig
1510 Ungültiger Parameter.
1512 Das Child wurde bereits hinzugefügt
1514 Ungültige Art an dieser Stelle

29.3.2.32.20 InsertChildBefore

Methode: InsertChildBefore (Node als XMLData, NewData als XMLData)

Beschreibung
Fügt einen neuen XML-Node (definiert im zweiten Parameter) vor dem (im ersten Parameter) angegebenen Node ein.

Errors
1500 Ungültiges Objekt.
1506 Ungültige XML-Input-Datei
1507 Es sind keine Child-Nodes zulässig
1510 Ungültiger Parameter.
1512 Das Child wurde bereits hinzugefügt
1514 Ungültige Art an dieser Stelle

29.3.2.32.21 IsSameNode

Deklaration: IsSameNode (*pNodeToCompare* als XMLData[1970]) als Boolean

Beschreibung

Gibt true zurück, wenn pNodeToComparedenselben Node wie das Objekt selbst referenziert.

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1506 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.32.22 Kind

Dellaration: Kind als SPYXMLDataKind 2033

Beschreibung
Art dieses XMLDataObjekts. Diese Eigenschaft ist schreibgeschützt.

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.32.23 MayHaveChildren

Deklaration: MayHaveChildren als Boolean

Beschreibung
Gibt an, ob Child-Elemente zu diesem XMLDataObjekt hinzugefügt werden dürfen.
Diese Eigenschaft ist schreibgeschützt.

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.32.24 Name

Deklaration: Name als String

Beschreibung
Dient zum Bearbeiten und Abrufen des Namens des XMLDataObjekts.

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.2.32.25 Parent

Deklaration: Parent als XMLData 1970

Rückgabewert
Übergeordnetes Element als XMLDataObjekt. Nichts (oder NULL), wenn es kein übergeordnetes Element gibt.

Beschreibung
Diesem Element übergeordnetes Element. Diese Eigenschaft ist schreibgeschützt.

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

© 2016-2022 Altova GmbH

29.3.2.32.26 SetTextValueXMLEncoded

Methode: SetTextValueXMLEncoded (*strVal* als String[2033])

Beschreibung
Definiert den kodierten Textwert der XML-Datei.

Fehler
1500 Ungültiges Objekt.
1513 Keine Änderung zulässig.

29.3.2.32.27 TextValue

Deklaration: TextValue als String

Beschreibung
Dient zum Bearbeitung und Abrufen des Textwerts dieses XMLDataObjekts.

Fehler
1500 Das XMLData-Objekt ist nicht mehr gültig.
1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

29.3.3 Schnittstellen (nicht mehr verwendet)

Die in diesem Teil beschriebenen Schnittstellen werden nicht mehr verwendet. Es wird empfohlen Ihre Applikationen auf die neuen Schnittstellen zu migrieren. Tipps zur Migration finden Sie unter den verschiedenen Eigenschaften und Methoden in diesem Teil.

29.3.3.1 AuthenticEvent (nicht mehr verwendet)

Wird ersetzt durch AuthenticView[1803] und AuthenticRange[1773]

Das DocEditView Objekt wird umbenannt in OldAuthenticView.
DocEditSelection wird umbenannt in AuthenticSelection.
DocEditEvent wird umbenannt in AuthenticEvent.
DocEditDataTransfer wird umbenannt in AuthenticDataTransfer.

Die Verwendung dieser Objekte wird nicht mehr empfohlen - mit Ausnahme der Verwendung für AuthenticDataTransfer. Bestehende Funktionalitäten werden auch in Zukunft auf unbestimmte Zeit unterstützt, doch es werden keine neuen Funktionen zu dieser Schnittstelle hinzugefügt. Alle bisher in DocEditView[1996], DocEditSelection[1994], DocEditEvent[1981] und DocEditDataTransfer[1769] verfügbaren

Funktionalitäten stehen nun über AuthenticView[1803], AuthenticRange[1773] und AuthenticDataTransfer[1768] zur Verfügung. Es wurden viele neue Funktionen hinzugefügt.

Beispiele zur Migration von DocEdit in Authentic finden Sie unter der Beschreibung der verschiedenen Methoden und Eigenschaften der einzelnen DocEdit Objekte.

Eigenschaften

altKey[1982]
altLeft[1983]
ctrlKey[1987]
ctrlLeft[1988]
shiftKey[1991]
shiftLeft[1992]

keyCode[1990]
repeat[1991]

button[1984]

clientX[1986]
clientY[1986]

dataTransfer[1989]

srcElement[1993]
fromElement[1990]

propertyName[1991]

cancelBubble[1985]
returnValue[1991]

type[1994]

Beschreibung
DocEditEventSchnittstelle.

29.3.3.1.1 altKey (nicht mehr verwendet)

Wird ersetzt durch Parameter für
AuthenticView.OnKeyboardEvent[1809](On_AuthenticView_KeyPressed)
AuthenticView.OnMouseEvent[1810](On_AuthenticView_MouseEvent)
AuthenticView.OnDragOver[1807](On_AuthenticView_DragOver)

© 2016-2022 Altova GmbH

Das Event-Objekt, das die Informationen des letzten Event enthält, wird nun durch Parameter für die verschiedenen Event Handler-Funktionen ersetzt, um den Datenzugriff zu erleichern. Aus Kompatibilitätsgründen wird das Event Objekt auf unbestimmte Zeit weiter unterstützt. Es sind keine Verbesserungen geplant. Wir empfehlen Ihnen, nach Möglichkeit die neuen Event Handler Funktionen zu verwenden.

```
// ----- XMLSpy scripting environment - javascript sample -----
// instead of:
// function On_DocEditKeyPressed ()
// {
//     if (Application.ActiveDocument.DocEditView.event.altKey ||
//         Application.ActiveDocument.DocEditView.event.altLeft)
//       MsgBox ("alt key is down");
// }
// use now:
function On_AuthenticView_KeyPressed (SPYKeyEvent i_eKeyEvent, long i_nKeyCode,
SPYVirtualKeyMask i_nVirtualKeyStatus)
{
   if(i_nVirtualKeyStatus & spyAltKeyMask)
      MsgBox ("alt key is down");
}
```

Deklaration: altKey als Boolean

Beschreibung
True, wenn die rechte ALT-Taste gedrückt wird.

29.3.3.1.2 altLeft (nicht mehr verwendet)

Wird ersetzt durch Parameter für
AuthenticView.OnKeyboardEvent[1809](On_AuthenticView_KeyPressed)
AuthenticView.OnMouseEvent[1810](On_AuthenticView_MouseEvent)
AuthenticView.OnDragOver[1807](On_AuthenticView_DragOver)

Das Event-Objekt, das die Informationen des letzten Event enthält, wird nun durch Parameter für die verschiedenen Event Handler-Funktionen ersetzt, um den Datenzugriff zu erleichern. Aus Kompatibilitätsgründen wird das Event Objekt auf unbestimmte Zeit weiter unterstützt. Es sind keine Verbesserungen geplant. Wir empfehlen Ihnen, nach Möglichkeit die neuen Event Handler Funktionen zu verwenden.

```
// ----- XMLSpy scripting environment - javascript sample -----
// instead of:
// function On_DocEditKeyDown ()
// {
//     if (Application.ActiveDocument.DocEditView.event.altKey ||
//         Application.ActiveDocument.DocEditView.event.altLeft)
```

```
//       MsgBox ("alt key is down");
// }
// use now:
function On_AuthenticView_KeyDown (SPYKeyEvent i_eKeyEvent, longi_nKeyCode, SPYVirtualKeyMask
i_nVirtualKeyStatus)
{
   if(i_nVirtualKeyStatus & spyAltKeyMask)
      MsgBox ("alt key is down");
}
```

Deklaration: altLeft als Boolean

Beschreibung
True, wenn die linke ALT-Taste gedrückt wird.

29.3.3.1.3 button (nicht mehr verwendet)

Wird ersetzt durch Parameter für
AuthenticView.OnMouseEvent[1810](On_AuthenticView_MouseEvent)
AuthenticView.OnDragOver[1807](On_AuthenticView_DragOver)

Das Event-Objekt, das die Informationen des letzten Event enthält, wird nun durch Parameter für die verschiedenen Event Handler-Funktionen ersetzt, um den Datenzugriff zu erleichern. Aus Kompatibilitätsgründen wird das Event Objekt auf unbestimmte Zeit weiter unterstützt. Es sind keine Verbesserungen geplant. Wir empfehlen Ihnen, nach Möglichkeit die neuen Event Handler Funktionen zu verwenden.

```
// ----- XMLSpy scripting environment - javascript sample -----
// instead of:
// function On_DocEditButtonDown ()
// {
//    if (Application.ActiveDocument.DocEditView.event.button == 1)
//       MsgBox ("left mouse button down detected");
// }
// use now:
function On_AuthenticView_MouseEvent (longi_nXPos, longi_nYPos, SPYMouseEvent i_eMouseEvent,
IAuthenticRange *i_ipRange)
{
   if(i_eMouseEvent & spyLeftButtonDownMask)
      MsgBox ("left mouse button down detected");
}
```

Deklaration: button als long

Beschreibung
Definiert, welche Maustaste gedrückt wird:

© 2016-2022 Altova GmbH

0	Es wird keine Taste gedrückt.
1	Die linke Taste wird gedrückt.
2	Die rechte Taste wird gedrückt.
3	Es werden die linke und die rechte Taste gedrückt.
4	Die mittlere Taste wird gedrückt.
5	Es werden die linke und die mittlere Taste gedrückt.
6	Es werden die rechte und die mittlere Taste gedrückt.
7	Alle drei Taste werden gedrückt.

29.3.3.1.4 cancelBubble (nicht mehr verwendet)

Wird ersetzt durch den Boolschen Rückgabewert der folgenden Event Handler-Funktionen
AuthenticView.OnKeyboardEvent[1809](On_AuthenticView_KeyPressed)
AuthenticView.OnMouseEvent[1810](On_AuthenticView_MouseEvent)
AuthenticView.OnDragOver[1807](On_AuthenticView_DragOver)

Das Event-Objekt, das die Informationen des letzten Event enthält, wird nun durch Parameter für die verschiedenen Event Handler-Funktionen ersetzt, um den Datenzugriff zu erleichern. Aus Kompatibilitätsgründen wird das Event Objekt auf unbestimmte Zeit weiter unterstützt. Es sind keine Verbesserungen geplant. Wir empfehlen Ihnen, nach Möglichkeit die neuen Event Handler Funktionen zu verwenden

Wenn von einer Event Handler-Funktion *true* zurückgegeben wird, bedeutet dies, dass das Event behandelt wurde und die normale Event-Behandlung abgebrochen werden soll.

```
// ----- XMLSpy scripting environment - javascript sample -----
// instead of:
// function On_DocEditKeyPressed ()
// {
//     if (Application.ActiveDocument.DocEditView.event.keyCode == 0x20)
//     {
//        // cancel key processing, swallow spaces :-)
//        Application.ActiveDocument.DocEditView.event.cancelBubble = true;
//     }
// }
// use now:
function On_AuthenticView_KeyPressed (SPYKeyEvent i_eKeyEvent, longi_nKeyCode,
SPYVirtualKeyMask i_nVirtualKeyStatus)
{
   if(i_nKeyCode == 0x20)
      returntrue;  // cancel key processing, swallow spaces :-)
}
```

Dellaration: cancelBubble als Boolean

Beschreibung
Setzen Sie cancelBubbleauf TRUE, wenn der Standard-Event Handler nicht aufgerufen werden soll.

29.3.3.1.5 clientX (nicht mehr verwendet)

Wird ersetzt durch Parameter für
AuthenticView.OnMouseEvent[1810](On_AuthenticView_MouseEvent)
AuthenticView.OnBeforeDrop[1805](On_AuthenticView_BeforeDrop)
AuthenticView.OnDragOver[1807](On_AuthenticView_DragOver)

Das Event-Objekt, das die Informationen des letzten Event enthält, wird nun durch Parameter für die verschiedenen Event Handler-Funktionen ersetzt, um den Datenzugriff zu erleichern. Aus Kompatibilitätsgründen wird das Event Objekt auf unbestimmte Zeit weiter unterstützt. Es sind keine Verbesserungen geplant. Wir empfehlen Ihnen, nach Möglichkeit die neuen Event Handler Funktionen zu verwenden

```
// ----- XMLSpy scripting environment - javascript sample -----
// instead of:
// function On_DocEditMouseMove ()
// {
//     MsgBox ("moving over " + Application.ActiveDocument.DocEditView.event.clientX +
//             "/" + Application.ActiveDocument.DocEditView.event.clientY);
// }
// use now:
function On_AuthenticView_MouseEvent (longi_nXPos, longi_nYPos, SPYMouseEvent i_eMouseEvent,
IAuthenticRange *i_ipRange)
{
   if(i_eMouseEvent & spyMouseMoveMask)
      MsgBox ("moving over " + i_nXPos + "/" + i_nYPos);
}
```

Deklaration: clientX als long

Beschreibung
X-Wert der aktuellen Mausposition in den Client-Koordinaten.

29.3.3.1.6 clientY (nicht mehr verwendet)

Wird ersetzt durch Parameter für
AuthenticView.OnMouseEvent[1810](On_AuthenticView_MouseEvent)
AuthenticView.OnBeforeDrop[1805](On_AuthenticView_BeforeDrop)
AuthenticView.OnDragOver[1807](On_AuthenticView_DragOver)

© 2016-2022 Altova GmbH

Das Event-Objekt, das die Informationen des letzten Event enthält, wird nun durch Parameter für die verschiedenen Event Handler-Funktionen ersetzt, um den Datenzugriff zu erleichern. Aus Kompatibilitätsgründen wird das Event Objekt auf unbestimmte Zeit weiter unterstützt. Es sind keine Verbesserungen geplant. Wir empfehlen Ihnen, nach Möglichkeit die neuen Event Handler Funktionen zu verwenden

```
// ----- XMLSpy scripting environment - javascript sample -----
// instead of:
// function On_DocEditMouseMove ()
// {
//     MsgBox ("moving over " + Application.ActiveDocument.DocEditView.event.clientX +
//             "/" + Application.ActiveDocument.DocEditView.event.clientY);
// }
// use now:
function On_AuthenticView_MouseEvent (longi_nXPos, longi_nYPos, SPYMouseEvent i_eMouseEvent,
IAuthenticRange *i_ipRange)
{
   if(i_eMouseEvent & spyMouseMoveMask)
      MsgBox ("moving over " + i_nXPos + "/" + i_nYPos);
}
```

Deklaration: clientY als long

Beschreibung
Y-Wert der aktuellen Mausposition in den Client-Koordinaten.

29.3.3.1.7 ctrlKey (nicht mehr verwendet)

Wird ersetzt durch Parameter für
AuthenticView.OnKeyboardEvent[1809](On_AuthenticView_KeyPressed)
AuthenticView.OnMouseEvent[1810](On_AuthenticView_MouseEvent)
AuthenticView.OnDragOver[1807](On_AuthenticView_DragOver)

Das Event-Objekt, das die Informationen des letzten Event enthält, wird nun durch Parameter für die verschiedenen Event Handler-Funktionen ersetzt, um den Datenzugriff zu erleichern. Aus Kompatibilitätsgründen wird das Event Objekt auf unbestimmte Zeit weiter unterstützt. Es sind keine Verbesserungen geplant. Wir empfehlen Ihnen, nach Möglichkeit die neuen Event Handler Funktionen zu verwenden.

```
// ----- XMLSpy scripting environment - javascript sample -----
// instead of:
// function On_DocEditMouseMove ()
// {
//     if (Application.ActiveDocument.DocEditView.event.ctrlKey ||
//         Application.ActiveDocument.DocEditView.event.altLeft)
//        MsgBox ("control key is down");
```

```
// }
// use now:
function On_AuthenticView_MouseEvent (long i_nXPos, long i_nYPos, SPYMouseEvent i_eMouseEvent,
IAuthenticRange *i_ipRange)
{
   if(i_eMouseEvent & spyCtrlKeyMask)
      MsgBox ("control key is down");
}
```

Deklaration: ctrlKey als Boolean

Beschreibung
True, wenn die rechte Strg-Taste gedrückt wird.

29.3.3.1.8 ctrlLeft (nicht mehr verwendet)

Wird ersetzt durch parameters to
AuthenticView.OnKeyboardEvent[1809](On_AuthenticView_KeyPressed)
AuthenticView.OnMouseEvent[1810](On_AuthenticView_MouseEvent)
AuthenticView.OnDragOver[1807](On_AuthenticView_DragOver)

Das Event-Objekt, das die Informationen des letzten Event enthält, wird nun durch Parameter für die verschiedenen Event Handler-Funktionen ersetzt, um den Datenzugriff zu erleichern. Aus Kompatibilitätsgründen wird das Event Objekt auf unbestimmte Zeit weiter unterstützt. Es sind keine Verbesserungen geplant. Wir empfehlen Ihnen, nach Möglichkeit die neuen Event Handler Funktionen zu verwenden.

```
// ----- XMLSpy scripting environment - javascript sample -----
// instead of:
// function On_DocEditMouseMove ()
// {
//     if (Application.ActiveDocument.DocEditView.event.ctrlKey ||
//         Application.ActiveDocument.DocEditView.event.altLeft)
//       MsgBox ("control key is down");
// }
// use now:
function On_AuthenticView_MouseEvent (long i_nXPos, long i_nYPos, SPYMouseEvent i_eMouseEvent,
IAuthenticRange *i_ipRange)
{
   if(i_eMouseEvent & spyCtrlKeyMask)
      MsgBox ("control key is down");
}
```

Deklaration: ctrlLeft als Boolean

Beschreibung
True, wenn die linke Strg-Taste gedrückt wird.

© 2016-2022 Altova GmbH

29.3.3.1.9 dataTransfer (nicht mehr verwendet)

Wird ersetzt durch Parameter für
AuthenticView.OnBeforeDrop[1805](On_AuthenticView_BeforeDrop)
AuthenticView.OnDragOver[1807](On_AuthenticView_DragOver)

Das Event-Objekt, das die Informationen des letzten Event enthält, wird nun durch Parameter für die verschiedenen Event Handler-Funktionen ersetzt, um den Datenzugriff zu erleichern. Aus Kompatibilitätsgründen wird das Event Objekt auf unbestimmte Zeit weiter unterstützt. Es sind keine Verbesserungen geplant. Wir empfehlen Ihnen, nach Möglichkeit die neuen Event Handler Funktionen zu verwenden.

```
// ----- XMLSpy scripting environment - javascript sample -----
// instead of:
// function On_DocEditDrop ()
// {
//     if (Application.ActiveDocument.DocEditView.event.dataTransfer != null)
//         if (! Application.ActiveDocument.DocEditView.event.dataTransfer.ownDrag)
//         {
//           // cancel key processing, don't drop foreign objects :-)
//           Application.ActiveDocument.DocEditView.event.cancelBubble = true;
//         }
// }
// use now:
function On_AuthenticView_BeforeDrop (longi_nXPos, longi_nYPos,
                                      IAuthenticRange *i_ipRange,
                                      IAuthenticDataTransfer *i_ipData)
{
   if(i_ipRange != null)
      if(! i_ipRange.ownDrag)
         returntrue;  // cancel key processing, don't drop foreign objects :-)

   returnfalse;
}
```

Deklaration: dataTransfer als Variant

Beschreibung
Eigenschaft dataTransfer.

29.3.3.1.10 fromElement (nicht mehr verwendet)

Nicht unterstützt

Deklaration: fromElement als Variant (nicht unterstützt)

Beschreibung
Derzeit wird diese Eigenschaft von keinem Event definiert.

29.3.3.1.11 keyCode (nicht mehr verwendet)

Wird ersetzt durch einen Parameter für AuthenticView.OnKeyboardEvent 1809
(On_AuthenticView_KeyPressed)

Das Event-Objekt, das die Informationen des letzten Event enthält, wird nun durch Parameter für die verschiedenen Event Handler-Funktionen ersetzt, um den Datenzugriff zu erleichern. Aus Kompatibilitätsgründen wird das Event Objekt auf unbestimmte Zeit weiter unterstützt. Es sind keine Verbesserungen geplant. Wir empfehlen Ihnen, nach Möglichkeit die neuen Event Handler Funktionen zu verwenden.

```
// ----- XMLSpy scripting environment - javascript sample -----
// instead of:
// function On_DocEditKeyPressed ()
// {
//     if (Application.ActiveDocument.DocEditView.event.keyCode == 0x20)
//     {
//        // cancel key processing, swallow spaces :-)
//        Application.ActiveDocument.DocEditView.event.cancelBubble = true;
//     }
// }
// use now:
function On_AuthenticView_KeyPressed (SPYKeyEvent i_eKeyEvent, longi_nKeyCode,
SPYVirtualKeyMask i_nVirtualKeyStatus)
{
   if(i_nKeyCode == 0x20)
      returntrue;  // cancel key processing, swallow spaces :-)
}
```

Deklaration: keyCode als long

Beschreibung
Tastaturcode der aktuell gedrückten Taste. Diese Eigenschaft ist read-write.

© 2016-2022 Altova GmbH

29.3.3.1.12 propertyName (nicht mehr verwendet)

Nicht unterstützt

Deklaration: propertyName als String (nicht unterstützt)

Beschreibung
Derzeit wird diese Eigenschaft von keinem Event definiert.

29.3.3.1.13 repeat (nicht mehr verwendet)

Nicht unterstützt

Deklaration: repeat als Boolean (nicht unterstützt)

Beschreibung
True, wenn das onkeydownEvent wiederholt wird.

29.3.3.1.14 returnValue (nicht mehr verwendet)

Nicht mehr unterstützt

Deklaration: returnValue als Variant

Beschreibung
Mit Hilfe von returnValuekönnen Sie einen Rückgabewert für Ihren Event Handler setzen.

29.3.3.1.15 shiftKey (nicht mehr verwendet)

Wird ersetzt durch Parameter für
AuthenticView.OnKeyboardEvent[1809](On_AuthenticView_KeyPressed)
AuthenticView.OnMouseEvent[1810](On_AuthenticView_MouseEvent)
AuthenticView.OnDragOver[1807](On_AuthenticView_DragOver)

Das Event-Objekt, das die Informationen des letzten Event enthält, wird nun durch Parameter für die verschiedenen Event Handler-Funktionen ersetzt, um den Datenzugriff zu erleichern. Aus Kompatibilitätsgründen wird das Event Objekt auf unbestimmte Zeit weiter unterstützt. Es sind keine Verbesserungen geplant. Wir empfehlen Ihnen, nach Möglichkeit die neuen Event Handler Funktionen zu verwenden.

```
// ----- XMLSpy scripting environment - javascript sample -----
// instead of:
// function On_DocEditDragOver ()
// {
//     if (Application.ActiveDocument.DocEditView.event.shiftKey ||
//         Application.ActiveDocument.DocEditView.event.shiftLeft)
//       MsgBox ("shift key is down");
// }
// use now:
function On_AuthenticView_DragOver (long i_nXPos, long i_nYPos,
                                    SPYMouseEvent i_eMouseEvent,
                                    IAuthenticRange *i_ipRange,
                                    IAuthenticDataTransfer *i_ipData)
{
   if(i_eMouseEvent & spyShiftKeyMask)
      MsgBox ("shift key is down");
}
```

Deklaration: shiftKey als Boolean

Beschreibung
True, wenn die rechte Umschalttaste gedrückt wird.

29.3.3.1.16 shiftLeft (nicht mehr verwendet)

Wird ersetzt durch Parameter für
AuthenticView.OnKeyboardEvent[1809](On_AuthenticView_KeyPressed)
AuthenticView.OnMouseEvent[1810](On_AuthenticView_MouseEvent)
AuthenticView.OnDragOver[1807](On_AuthenticView_DragOver)

Das Event-Objekt, das die Informationen des letzten Event enthält, wird nun durch Parameter für die verschiedenen Event Handler-Funktionen ersetzt, um den Datenzugriff zu erleichern. Aus Kompatibilitätsgründen wird das Event Objekt auf unbestimmte Zeit weiter unterstützt. Es sind keine Verbesserungen geplant. Wir empfehlen Ihnen, nach Möglichkeit die neuen Event Handler Funktionen zu verwenden.

```
// ----- XMLSpy scripting environment - javascript sample -----
// instead of:
// function On_DocEditDragOver ()
// {
//     if (Application.ActiveDocument.DocEditView.event.shiftKey ||
```

 © 2016-2022 Altova GmbH

```
//        Application.ActiveDocument.DocEditView.event.shiftLeft)
//       MsgBox ("shift key is down");
// }
// use now:
function On_AuthenticView_DragOver (long i_nXPos, long i_nYPos,
                                    SPYMouseEvent i_eMouseEvent,
                                    IAuthenticRange *i_ipRange,
                                    IAuthenticDataTransfer *i_ipData)
{
   if(i_eMouseEvent & spyShiftKeyMask)
      MsgBox ("shift key is down");
}
```

Deklaration: shiftLeft als Boolean

Beschreibung
True, wenn die linke Umschalttaste gedrückt wird.

29.3.3.1.17 srcElement (nicht mehr verwendet)

Wird ersetzt durch Parameter für
AuthenticView.OnMouseEvent[1810](On_AuthenticView_MouseEvent)
AuthenticView.OnBeforeDrop[1805](On_AuthenticView_BeforeDrop)
AuthenticView.OnDragOver[1807](On_AuthenticView_DragOver)

Das Event-Objekt, das die Informationen des letzten Event enthält, wird nun durch Parameter für die verschiedenen Event Handler-Funktionen ersetzt, um den Datenzugriff zu erleichern. Aus Kompatibilitätsgründen wird das Event Objekt auf unbestimmte Zeit weiter unterstützt. Es sind keine Verbesserungen geplant. Wir empfehlen Ihnen, nach Möglichkeit die neuen Event Handler Funktionen zu verwenden.

Mit der neuen Event Handler-Funktion wird anstelle des XMLData-Elements, das sich derzeit unterhalb des Mauscursors befindet, ein Bereichsobjekt bereitgestellt.

```
// ----- XMLSpy scripting environment - javascript sample -----
// instead of:
// function On_DocEditMouseMove ()
// {
//     var objEvent = Application.ActiveDocument.DocEditView.event;
//     if (objEvent.srcElement != null)
//         MsgBox ("moving over " + objEvent.srcElement.Parent.Name);
// }
// use now:
function On_AuthenticView_MouseEvent (longi_nXPos, longi_nYPos, SPYMouseEvent i_eMouseEvent,
IAuthenticRange *i_ipRange)
{
   if((i_eMouseEvent & spyMouseMoveMask) &&
```

```
    (i_ipRange != null))
    MsgBox ("moving over " + i_ipRange.FirstXMLData.Parent.Name);
}
```

Deklaration: srcElement als Variant

Beschreibung
Element, welches das aktuelle Event auslöst. Dies ist normalerweise ein XMLData[1970] Objekt.

29.3.3.1.18 type (nicht mehr verwendet)

Nicht unterstützt

Deklaration: type als String (nicht unterstützt)

Beschreibung
Derzeit wird diese Eigenschaft von keinem Event definiert.

29.3.3.2 AuthenticSelection (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange[1773]

Das DocEditView object wird umbenannt in OldAuthenticView.
DocEditSelection wird umbenannt in AuthenticSelection.
DocEditEvent wird umbenannt in AuthenticEvent.
DocEditDataTransfer wird umbenannt in AuthenticDataTransfer.

Die Verwendung dieser Objekte wird nicht mehr empfohlen - mit Ausnahme der Verwendung für AuthenticDataTransfer. Bestehende Funktionalitäten werden auch in Zukunft auf unbestimmte Zeit unterstützt, doch es werden keine neuen Funktionen zu dieser Schnittstelle hinzugefügt. Alle bisher in DocEditView[1996], DocEditSelection[1994], DocEditEvent[1981] und DocEditDataTransfer[1769] verfügbaren Funktionalitäten stehen nun über AuthenticView[1803], AuthenticRange[1773] und AuthenticDataTransfer[1768] zur Verfügung. Es wurden viele neue Funktionen hinzugefügt.

Beispiele zur Migration von DocEdit in Authentic finden Sie unter der Beschreibung der verschiedenen Methoden und Eigenschaften der einzelnen DocEdit Objekte.

Eigenschaften
Start[1995]
StartTextPosition[1996]
End[1995]

© 2016-2022 Altova GmbH

EndTextPosition[1995]

29.3.3.2.1 End (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.LastXMLData[1794]

```
// ----- javascript sample -----
// instead of:
// var objXMLData = Application.ActiveDocument.DocEditView.CurrentSelection.End;
// use now:
var objXMLData = Application.ActiveDocument.AuthenticView.Selection.LastXMLData;
```

Deklaration: End als XMLData[1970]

Beschreibung
XML-Element, an dem die aktuelle Auswahl endet.

29.3.3.2.2 EndTextPosition (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.LastXMLDataOffset[1794]

```
// ----- javascript sample -----
// instead of:
// var nOffset = Application.ActiveDocument.DocEditView.CurrentSelection.EndTextPosition;
// use now:
var nOffset = Application.ActiveDocument.AuthenticView.Selection.LastXMLDataOffset;
```

Deklaration: EndTextPosition als long

Beschreibung
Positioniert den Cursor in DocEditSelection.End.TextValue[1995],wo die Auswahl endet.

29.3.3.2.3 Start (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.FirstXMLData[1781]

```
// ----- javascript sample -----
```

```
// instead of:
// var objXMLData = Application.ActiveDocument.DocEditView.CurrentSelection.Start;
// use now:
var objXMLData = Application.ActiveDocument.AuthenticView.Selection.FirstXMLData;
```

Deklaration: Start als XMLData[1970]

Beschreibung
XML-Element, an dem die Cursorauswahl beginnt.

29.3.3.2.4 StartTextPosition (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.FirstXMLDataOffset[1782]

```
// ----- javascript sample -----
// instead of:
// var nOffset = Application.ActiveDocument.DocEditView.CurrentSelection.StartTextPosition;
// use now:
var nOffset = Application.ActiveDocument.AuthenticView.Selection.FirstXMLDataOffset;
```

Deklaration: StartTextPosition als long

Beschreibung
Positioniert den Cursor in DocEditSelection.Start.TextValue[1995],wo die Auswahl beginnt.

29.3.3.3 OldAuthentictView (nicht mehr verwendet)

Wird ersetzt durch AuthenticView[1803] und AuthenticRange[1773]

Das DocEditView object wird umbenannt in OldAuthenticView.
DocEditSelection wird umbenannt in AuthenticSelection.
DocEditEvent wird umbenannt in AuthenticEvent.
DocEditDataTransfer wird umbenannt in AuthenticDataTransfer.

Die Verwendung dieser Objekte wird nicht mehr empfohlen - mit Ausnahme der Verwendung für AuthenticDataTransfer. Bestehende Funktionalitäten werden auch in Zukunft auf unbestimmte Zeit unterstützt, doch es werden keine neuen Funktionen zu dieser Schnittstelle hinzugefügt. Alle bisher in DocEditView[1996], DocEditSelection[1994], DocEditEvent[1981] und DocEditDataTransfer[1769] verfügbaren Funktionalitäten stehen nun über AuthenticView[1803], AuthenticRange[1773] und AuthenticDataTransfer[1768] zur Verfügung. Es wurden viele neue Funktionen hinzugefügt.

Beispiele zur Migration von DocEdit in Authentic finden Sie unter der Beschreibung der verschiedenen Methoden und Eigenschaften der einzelnen DocEdit Objekte.

© 2016-2022 Altova GmbH

Methoden

LoadXML 2013
SaveXML 2017

EditClear 1999
EditCopy 2000
EditCut 2000
EditPaste 2001
EditRedo 2001
EditSelectAll 2001
EditUndo 2002

RowAppend 2014
RowDelete 2015
RowDuplicate 2015
RowInsert 2016
RowMoveDown 2016
RowMoveUp 2017

ApplyTextState 1998
IsTextStateApplied 2012
IsTextStateEnabled 2013

MarkUpView 2014

SelectionSet 2018
SelectionMoveTabOrder 2018

GetNextVisible 2005
GetPreviousVisible 2005

GetAllowedElements 2003

Eigenschaften
CurrentSelection 1999

event 2002

XMLRoot 2019

IsEditClearEnabled 2006
IsEditCopyEnabled 2006
IsEditCutEnabled 2007
IsEditPasteEnabled 2008
IsEditRedoEnabled 2008
IsEditUndoEnabled 2009

IsRowAppendEnabled 2009
IsRowDeleteEnabled 2010
IsRowDuplicateEnabled 2010
IsRowInsertEnabled 2011

IsRowMoveDownEnabled[2011]
IsRowMoveUpEnabled[2012]

Beschreibung
Schnittstelle für die Authentic-Ansicht.

29.3.3.3.1 ApplyTextState (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.PerformAction[1798]

Verwenden Sie spyAuthenticApply für den eAction-Parameter. Mit Hilfe der PerformAction-Methode können Sie Textstatusattribute auf jeden beliebigen Bereich des Dokuments und nicht nur auf die aktuelle Auswahl auf der Benutzeroberfläche anwenden.

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.ApplyTextState ("bold");
// use now:
if(! Application.ActiveDocument.AuthenticView.Selection.PerformAction (spyAuthenticApply, "bold"))
   MsgBox ("Error: can't set current selection to bold");
```

Deklaration: ApplyTextState (*elementName* als String)

Beschreibung
Wendet den durch den Parameter elementName definierten Textstatus an bzw. entfernt ihn. Häufige Beispiele für den Parameter elementName wären "strong" und "italic".

Ein XML-Dokument enthält Datensegmente, die Subelemente enthalten können. Nehmen wir als Beispiel den folgenden HTML-Code:

```
<b>fragment</b>
```

Aufgrund des HTML-Tags <b> wird das Wortfragment fett formatiert. Dies geschieht jedoch nur, weil der HTML-Parser weiß, dass der Tag <b> für fett steht. Bei XML gibt es viel mehr Flexibilität. Jeder beliebige XML-Tag kann beliebig nach Ihren Wünschen definiert werden. D.h. es kann mittels XML ein Textstatus angewendet werden. Der angewendete Textstatus muss jedoch Teil des Schemas sein. So ist z.B. in den Beispielen OrgChart.xml, OrgChart.sps, OrgChart.xsd der Tag derselbe wie "bold". Und um "bold" anzuwenden, wird die Methode ApplyTextState() aufgerufen. Wie bei den Operationen "row" und "edit" muss jedoch überprüft werden, ob der Textstatus angewendet werden kann.

Siehe auch IsTextStateEnabled[2013] und IsTextStateApplied[2012].

© 2016-2022 Altova GmbH

29.3.3.3.2 CurrentSelection (nicht mehr verwendet)

Wird ersetzt durch AuthenticView.Selection 1820

Das zurückgegebene Objekt AuthenticRange 1773 unterstützt die Navigation mittels XMLData-Elemente sowie die Navigation über Dokumentelemente (z.B. Zeichen, Wörter, Tags) oder Textcursorpositionen.

```
// ----- javascript sample -----
// instead of:
// var objDocEditSel = Application.ActiveDocument.DocEditView.CurrentSelection;
// use now:
   var objRange = Application.ActiveDocument.AuthenticView.Selection;
```

Deklaration: CurrentSelection als DocEditSelection 1994

Beschreibung
Die Eigenschaft bietet Zugriff auf die aktuelle Auswahl in der Authentic-Ansicht.

29.3.3.3.3 EditClear (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.Delete 1778

Mit Hilfe der Delete-Methode von AuthenticRange können Sie jeden beliebigen Bereich des Dokuments löschen, nicht nur die aktuellen Auswahl auf der Benutzeroberfläche.

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.EditClear();
// use now:
if(! Application.ActiveDocument.AuthenticView.Selection.Delete())
   MsgBox ("Error: can't delete current selection");
```

Deklaration: EditClear

Beschreibung
Löscht die aktuelle Auswahl.

29.3.3.3.4 EditCopy(nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.Copy 1777

Mit Hilfe der Copy-Methode von AuthenticRange können Sie jeden beliebigen Bereich des Dokuments kopieren, nicht nur die aktuellen Auswahl auf der Benutzeroberfläche.

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.EditCopy();
// use now:
if(! Application.ActiveDocument.AuthenticView.Selection.Copy())
   MsgBox ("Error: can't copy current selection");
```

Deklaration: EditCopy

Beschreibung
Kopiert die aktuelle Auswahl in die Zwischenablage.

29.3.3.3.5 EditCut (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.Cut 1778

Mit Hilfe der Cut-Methode von AuthenticRange können Sie jeden beliebigen Bereich des Dokuments ausschneiden, nicht nur die aktuellen Auswahl auf der Benutzeroberfläche.

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.EditCut();
// use now:
if(! Application.ActiveDocument.AuthenticView.Selection.Cut())
   MsgBox ("Error: can't cut out current selection");
```

Deklaration: EditCut

Beschreibung
Schneidet die aktuelle Auswahl aus dem Dokument aus und kopiert sie in die Zwischenablage.

© 2016-2022 Altova GmbH

29.3.3.3.6 EditPaste (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.Paste 1797

Mit Hilfe der Paste-Methode von AuthenticRange können Sie jeden beliebigen Bereich des Dokuments einfügen, nicht nur die aktuellen Auswahl auf der Benutzeroberfläche.

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.EditPaste();
// use now:
if(! Application.ActiveDocument.AuthenticView.Selection.Paste())
   MsgBox ("Error: can't paste to current selection");
```

Deklaration: EditPaste

Beschreibung
Fügt den Inhalt aus der Zwischenablage in das Dokument ein.

29.3.3.3.7 EditRedo (nicht mehr verwendet)

Wird ersetzt durch AuthenticView.Redo 1820

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.EditRedo();
// use now:
if(! Application.ActiveDocument.AuthenticView.Redo())
   MsgBox ("Error: no redo step available");
```

Deklaration: EditRedo

Beschreibung
Wiederholt den zuletzt rückgängig gemachten Schritt.

29.3.3.3.8 EditSelectAll (nicht mehr verwendet)

Wird ersetzt durch AuthenticView.WholeDocument 1822 and **AuthenticRange.Select** 1799

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.EditSelectAll();
// use now:
Application.ActiveDocument.AuthenticView.WholeDocument.Select();
```

Deklaration: EditSelectAll

Beschreibung
Die Methode wählt das gesamte Dokument aus.

29.3.3.3.9 EditUndo (nicht mehr verwendet)

Wird ersetzt durch AuthenticView.Undo [1821]

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.EditUndo();
// use now:
if(! Application.ActiveDocument.AuthenticView.Undo())
   MsgBox ("Error: no undo step available");
```

Deklaration: EditUndo

Beschreibung
Macht die letzte Aktion rückgängig.

29.3.3.3.10 event (nicht mehr verwendet)

Wird ersetzt durch Parameter für AuthenticView events [1803]**.**

Dkclaration: event als DocEditEvent [1981]

Beschreibung
Die Event-Eigenschaft enthält ein DocEditEventObjekt, das Informationen über das aktuelle Event enthält.

© 2016-2022 Altova GmbH

29.3.3.3.11 GetAllowedElements (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.CanPerformActionWith[1776]

AuthenticRange unterstützt nun alle Funktionalitäten der Elementeingabehilfen. Neben der Abfrage, welche Elemente eingefügt, angehängt usw. werden können, können Sie auch die Aktion selbst aufrufen. Nähere Informationen dazu finden Sie unter AuthenticRange.PerformAction[1798].

```
// ----- javascript sample -----
// instead of:
// var arrElements = New Array();
// var objDocEditView = Application.ActiveDocument.DocEditView;
// var objStartElement = objDocEditView.CurrentSelection.Start;
// var objEndElement = objDocEditView.CurrentSelection.End;
// objDocEditView.GetAllowedElements(k_ActionInsertBefore, objStartElement, objEndElement,
arrElements);
// use now:
var arrElements = New Array();
Application.ActiveDocument.AuthenticView.Selection.CanPerformActionWith (spyAuthenticInsertBefore,
arrElements);
```

Deklaration: GetAllowedElements (*nAction* als SpyAuthenticElementActions[2021], *pStartElement* als XMLData[1970], *pEndElement* als XMLData[1970], *pElements* als Variant)

Beschreibung
GetAllowedElements()gibt die zulässigen Elemente für die verschiedenen von nActiondefinierten Aktionen zurück.

JavaScript-Beispiel:

```
FunctionGetAllowed()
{
	var objView = Application.ActiveDocument.DocEditView;

	var arrElements = NewArray(1);

	var objStart = objView.CurrentSelection.Start;
	var objEnd = objView.CurrentSelection.End;

	var strText;
	strText = "valid elements at current selection:\n\n";

	For(var i = 1;i <= 4;i++)  {
		objPlugIn.GetAllowedElements(i,objStart,objEnd,arrElements);
		strText = strText + ListArray(arrElements) + "----------\n";
	}

	ReturnstrText;
}

FunctionListArray(arrIn)
```

```
{
        var strText = "";

        If(TypeOf(arrIn) == "object")        {
                For(var i = 0;i <= (arrIn.length - 1);i++)
                        strText = strText + arrIn[i] + "\n";
        }

        ReturnstrText;
}
```

VBScript-Beispiel:

```
SubDisplayAllowed
        DimobjView
        SetobjView = Application.ActiveDocument.DocEditView

        DimarrElements()

        DimobjStart
        DimobjEnd
        SetobjStart = objView.CurrentSelection.Start
        SetobjEnd = objView.CurrentSelection.End

        DimstrText
        strText = "valid elements at current selection:" & chr(13) & chr(13)

        Dimi

        Fori = 1 To4
                objView.GetAllowedElements i,objStart,objEnd,arrElements
                strText = strText & ListArray(arrElements) & "----------" & chr(13)
        Next

        msgbox strText
EndSub

FunctionListArray(arrIn)
        DimstrText

        IfIsArray(arrIn) Then
                Dimi

                Fori = 0 ToUBound(arrIn)
                        strText = strText & arrIn(i) & chr(13)
                Next
        EndIf

        ListArray = strText
EndFunction
```


© 2016-2022 Altova GmbH

29.3.3.3.12 GetNextVisible (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.SelectNext[1799]

AuthenticRange unterstützt nun eine breite Palette an Elementnavigationsmethoden basierend auf Dokumentelementen wie Zeichen, Wörter, Tags und vieles mehr. Die Auswahl der Textpassage, die den Inhalt des nächsten XML-Elements darstellt, ist nur eine davon.

```
// ----- javascript sample -----
// instead of:
// var objCurrXMLData = ...
// var objXMLData = Application.ActiveDocument.DocEditView.GetNextVisible(objCurrXMLData);
// Application.ActiveDocument.DocEditView.SelectionSet (objXMLData, 0, objXMLData, -1);
// use now:
var objRange = ...
try
   { objRange.SelectNext (spyAuthenticTag).Select(); }
catch(err)
{
   if((err.number & 0xffff) == 2003)
      MsgBox ("end of document reached");
   else
      throw(err);
}
```

Deklaration: GetNextVisible (*pElement* als XMLData[1970]) als XMLData[1970]

Beschreibung
Die Methode ruft das nächste sichtbare XML-Element im Dokument ab.

29.3.3.3.13 GetPreviousVisible (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.SelectPrevious[1800]

AuthenticRange unterstützt nun eine breite Palette an Elementnavigationsmethoden basierend auf Dokumentelementen wie Zeichen, Wörter, Tags und vieles mehr. Die Auswahl der Textpassage, die den Inhalt des vorherigen XML-Elements darstellt, ist nur eine davon.

```
// ----- javascript sample -----
// instead of:
// var objCurrXMLData = ...
// var objXMLData = Application.ActiveDocument.DocEditView.GetPreviousVisible(objCurrXMLData);
```

```
// Application.ActiveDocument.DocEditView.SelectionSet (objXMLData, 0, objXMLData, -1);
// use now:
var objRange = ...
try
   { objRange.SelectPrevious (spyAuthenticTag).Select(); }
catch(err)
{
   if((err.number & 0xffff) == 2004)
      MsgBox ("begin of document reached");
   else
      throw(err);
}
```

Deklaration: GetPreviousVisible (*pElement* als XMLData[1970]) als XMLData[1970]

Beschreibung
Die Methode ruft das vorhergehende sichtbare XML-Element im Dokument ab.

29.3.3.3.14 IsEditClearEnabled (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.IsDeleteEnabled[1790]

Die IsDeleteEnabled-Eigenschaft wird nun für jeden Bereich des Dokuments und nicht nur für die aktuelle Auswahl auf der Benutzeroberfläche unterstützt.

```
// ----- javascript sample -----
// instead of:
// if (Application.ActiveDocument.DocEditView.IsEditClearEnabled)
//    Application.ActiveDocument.DocEditView.EditClear();
// use now:
var objCurrSelection = Application.ActiveDocument.AuthenticView.Selection;
if(objCurrSelection.IsDeleteEnabled)
   objCurrSelection.Delete();
```

Deklaration: IsEditClearEnabled als Boolean

Beschreibung
True, wenn EditClear[1999] möglich ist. Siehe auch Bearbeitungsoperationen.[1744]

29.3.3.3.15 IsEditCopyEnabled (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.IsCopyEnabled[1790]

© 2016-2022 Altova GmbH

Die IlsCopyEnabled-Eigenschaft wird nun für jeden Bereich des Dokuments und nicht nur für die aktuelle Auswahl auf der Benutzeroberfläche unterstützt.

```
// ----- javascript sample -----
// instead of:
// if (Application.ActiveDocument.DocEditView.IsEditCopyEnabled)
//    Application.ActiveDocument.DocEditView.EditCopy();
// use now:
var objCurrSelection = Application.ActiveDocument.AuthenticView.Selection;
if(objCurrSelection.IsCopyEnabled)
   objCurrSelection.Copy();
```

Deklaration: IsEditCopyEnabled als Boolean

Beschreibung
True, wenn das Kopieren in die Zwischenablage möglich ist. Siehe auch EditCopy[2000] und Bearbeitungsoperationen[1744].

29.3.3.3.16 IsEditCutEnabled (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.IsCutEnabled[1790]

Die IsCutEnabled-Eigenschaft wird nun für jeden Bereich des Dokuments und nicht nur für die aktuelle Auswahl auf der Benutzeroberfläche unterstützt.

```
// ----- javascript sample -----
// instead of:
// if (Application.ActiveDocument.DocEditView.IsEditCutEnabled)
//    Application.ActiveDocument.DocEditView.EditCut();
// use now:
var objCurrSelection = Application.ActiveDocument.AuthenticView.Selection;
if(objCurrSelection.IsCutEnabled)
   objCurrSelection.Cut();
```

Deklaration: IsEditCutEnabled als Boolean

Beschreibung
True, wenn EditCut[2000] derzeit möglich ist. Siehe auch Bearbeitungsoperationen[1744].

29.3.3.3.17 IsEditPasteEnabled (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.IsPasteEnabled[1792]

Die IsPasteEnabled-Eigenschaft wird nun für jeden Bereich des Dokuments und nicht nur für die aktuelle Auswahl auf der Benutzeroberfläche unterstützt.

```
// ----- javascript sample -----
// instead of:
// if (Application.ActiveDocument.DocEditView.IsEditPasteEnabled)
//    Application.ActiveDocument.DocEditView.EditPaste();
// use now:
var objCurrSelection = Application.ActiveDocument.AuthenticView.Selection;
if(objCurrSelection.IsPasteEnabled)
   objCurrSelection.Paste();
```

Deklaration: IsEditPasteEnabled als Boolean

Beschreibung
True, wenn EditCut[2000] derzeit möglich ist. Siehe auch Bearbeitungsoperationen[1744].

29.3.3.3.18 IsEditRedoEnabled (nicht mehr verwendet)

Wird ersetzt durch AuthenticView.IsRedoEnabled[1818]

```
// ----- javascript sample -----
// instead of:
// if (Application.ActiveDocument.DocEditView.IsEditRedoEnabled)
//    Application.ActiveDocument.DocEditView.EditRedo();
// use now:
if(Application.ActiveDocument.AuthenticView.IsRedoEnabled)
   Application.ActiveDocument.AuthenticView.Redo();
```

Deklaration: IsEditRedoEnabled als Boolean

Beschreibung
True, wenn EditRedo[2001] derzeit möglich ist. Siehe auch Bearbeitungsoperationen[1744].

© 2016-2022 Altova GmbH

29.3.3.3.19 IsEditUndoEnabled (nicht mehr verwendet)

Wird ersetzt durch AuthenticView.IsUndoEnabled 1818

```
// ----- javascript sample -----
// instead of:
// if (Application.ActiveDocument.DocEditView.IsEditUndoEnabled)
//    Application.ActiveDocument.DocEditView.EditUndo();
// use now:
if(Application.ActiveDocument.AuthenticView.IsUndoEnabled)
   Application.ActiveDocument.AuthenticView.Undo();
```

Deklaration: IsEditUndoEnabled als Boolean

Beschreibung
True, wenn EditUndo 2002 derzeit möglich ist. Siehe auch Bearbeitungsoperationen 1744.

29.3.3.3.20 IsRowAppendEnabled (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.IsInDynamicTable 1791

Die Operation zum Einfügen, Anhängen, Löschen und Kopieren einer Zeile ('insert', 'append', 'delete' und 'duplicate' row) stehen zur Verfügung, wenn sich die Auswahl innerhalb einer dynamischen Tabelle befindet.

```
// ----- javascript sample -----
// instead of:
// if (Application.ActiveDocument.DocEditView.IsRowAppendEnabled)
//    Application.ActiveDocument.DocEditView.RowAppend();
// use now:
if(Application.ActiveDocument.AuthenticView.Selection.IsInDynamicTable())
   Application.ActiveDocument.AuthenticView.Selection.AppendRow();
```

Deklaration: IsRowAppendEnabled als Boolean

Beschreibung
True, wenn RowAppend 2014 möglich ist. Siehe auch Zeilenoperationen 1743.

29.3.3.3.21 IsRowDeleteEnabled (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.IsInDynamicTable 1791

Die Operation zum Einfügen, Anhängen, Löschen und Kopieren einer Zeile ('insert', 'append', 'delete' und 'duplicate' row) stehen zur Verfügung, wenn sich die Auswahl innerhalb einer dynamischen Tabelle befindet.

```
// ----- javascript sample -----
// instead of:
// if (Application.ActiveDocument.DocEditView.IsRowDeleteEnabled)
//    Application.ActiveDocument.DocEditView.Rowdelete();
// use now:
if(Application.ActiveDocument.AuthenticView.Selection.IsInDynamicTable())
   Application.ActiveDocument.AuthenticView.Selection.DeleteRow();
```

Deklaration: IsRowDeleteEnabled als Boolean

Beschreibung
True, wenn RowDelete 2015 möglich ist. Siehe auch Zeilenoperationen 1743.

29.3.3.3.22 IsRowDuplicateEnabled (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.IsInDynamicTable 1791

Die Operation zum Einfügen, Anhängen, Löschen und Kopieren einer Zeile ('insert', 'append', 'delete' und 'duplicate' row) stehen zur Verfügung, wenn sich die Auswahl innerhalb einer dynamischen Tabelle befindet.

```
// ----- javascript sample -----
// instead of:
// if (Application.ActiveDocument.DocEditView.IsRowDuplicateEnabled)
//    Application.ActiveDocument.DocEditView.RowDuplicate();
// use now:
if(Application.ActiveDocument.AuthenticView.Selection.IsInDynamicTable())
   Application.ActiveDocument.AuthenticView.Selection.DuplicateRow();
```

Deklaration: IsRowDuplicateEnabled als Boolean

Beschreibung
True, wenn RowDuplicate 2015 möglich ist. Siehe auch Zeilenoperationen 1743.

© 2016-2022 Altova GmbH

29.3.3.3.23 IsRowInsertEnabled (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.IsInDynamicTable 1791

Die Operation zum Einfügen, Anhängen, Löschen und Kopieren einer Zeile ('insert', 'append', 'delete' und 'duplicate' row) stehen zur Verfügung, wenn sich die Auswahl innerhalb einer dynamischen Tabelle befindet.

```
// ----- javascript sample -----
// instead of:
// if (Application.ActiveDocument.DocEditView.IsRowInsertEnabled)
//    Application.ActiveDocument.DocEditView.RowInsert();
// use now:
if(Application.ActiveDocument.AuthenticView.Selection.IsInDynamicTable())
   Application.ActiveDocument.AuthenticView.Selection.InsertRow();
```

Deklaration: IsRowInsertEnabled als Boolean

Beschreibung
True, wenn RowInsert 2016 möglich ist. Siehe auch Zeilenoperationen 1743.

29.3.3.3.24 IsRowMoveDownEnabled (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.IsLastRow 1792

```
// ----- javascript sample -----
// instead of:
// if (Application.ActiveDocument.OldAuthenticView.IsRowMoveDownEnabled)
//    Application.ActiveDocument.DocEditView.RowMoveDown();
// use now:
if(!Application.ActiveDocument.AuthenticView.Selection.IsLastRow)
   Application.ActiveDocument.AuthenticView.Selection.MoveRowDown();
```

Deklaration: IsRowMoveDownEnabled als Boolean

Beschreibung
True, wenn RowMoveDown 2016 möglich ist. Siehe auch Zeilenoperationen 1743.

29.3.3.3.25 IsRowMoveUpEnabled (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.IsFirstRow[1791]

```
// ----- javascript sample -----
// instead of:
// if (Application.ActiveDocument.DocEditView.IsRowMoveUpEnabled)
//    Application.ActiveDocument.DocEditView.RowMoveUp();
// use now:
if(!Application.ActiveDocument.AuthenticView.Selection.IsFirstRow)
   Application.ActiveDocument.AuthenticView.Selection.MoveRowUp();
```

Deklaration: IsRowMoveUpEnabled als Boolean

Beschreibung
True, wenn RowMoveUp[2017] möglich ist. Siehe auch Zeilenoperationen[1743].

29.3.3.3.26 IsTextStateApplied (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.IsTextStateApplied[1793]

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.IsTextStateApplied ("bold");
// use now:
if(Application.ActiveDocument.AuthenticView.Selection.IsTextStateApplied ("bold"))
   MsgBox ("bold on");
else
   MsgBox ("bold off");
```

Deklaration: IsTextStateApplied (*elementName* als String) als Boolean

Beschreibung
Überprüft, ob der Textstatus bereits angewendet wurde. Typische Beispiele für den Parameter elementName sind "strong" und "italic".

© 2016-2022 Altova GmbH

29.3.3.3.27 IsTextStateEnabled (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.CanPerformAction 1776

Verwenden Sie spyAuthenticApply für den eAction-Parameter. Mit Hilfe der CanPerformAction Methode können Sie an jedem Bereich des Dokuments arbeiten und nicht nur an dem aktuell auf der Benutzeroberfläche ausgewählten.

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.IsTextStateEnabled ("bold");
// use now:
if(Application.ActiveDocument.AuthenticView.Selection.CanPerformAction (spyAuthenticApply, "bold"))
   ... // e.g. enable 'bold' button
```

Deklaration: IsTextStateEnabled (*i_strElementName* als String) als Boolean

Beschreibung
Überprüft, ob ein Textstatus angewendet werden kann. Typische Beispiele für den Parameter elementName sind "strong" und "italic".

29.3.3.3.28 LoadXML (nicht mehr verwendet)

Wird ersetzt durch AuthenticView.AsXMLString 1813

AuthenticView unterstützt nun die Eigenschaft AsXMLString, über die Sie den Dokumentinhalt direkt als XMLString aufrufen und ersetzen können.

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.LoadXML (strDocAsXMLString);
// use now:
try
   { Application.ActiveDocument.AuthenticView.AsXMLString = strDocAsXMLString; }
catch (err)
   { MsgBox ("Error: invalid XML string"); }
```

Deklaration: LoadXML (*xmlString* als String)

Beschreibung
Lädt das aktuelle XML-Dokument, wobei der XML-String angewendet wird. Der neue Inhalt wird sofort angezeigt.

Der Parameter *xmlString* muss mit der XML-Deklaration beginnen, z.B.,
objPlugIn.LoadXML("<?xml version='1.0' encoding='UTF-8'?><root></root>");

29.3.3.3.29 MarkUpView (nicht mehr verwendet)

Wird ersetzt durch AuthenticView.MarkupVisibility 1819

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.MarkuUpView = 2;
// use now:
Application.ActiveDocument.AuthenticView.MarkupVisibility = spyAuthenticMarkupLarge;
```

Deklaration: MarkUpView (*kind* als long)

Beschreibung
Standardmäßig wird das Dokument mittels HTML-Verfahren angezeigt. Manchmal ist es aber sinnvoll, wenn die Bearbeitungstags angezeigt werden. Mit dieser Methode können drei unterschiedliche Arten von Markup Tags angezeigt werden:

0	Markup Tags ausblenden
2	Große Markup Tags einblenden
3	Gemischte Markup Tags einblenden

29.3.3.3.30 RowAppend (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.AppendRow 1775

Mit Hilfe der Tabellenoperationen von AuthenticRange kann nun jede Tabelle im aktuellen Dokument unabhängig von der aktuellen Auswahl auf der Benutzeroberfläche bearbeitet werden.

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.RowAppend();
// use now:
if(! Application.ActiveDocument.AuthenticView.Selection.AppendRow())
   MsgBox ("Error: can't append row");
```

Deklaration: RowAppend

Beschreibung

© 2016-2022 Altova GmbH

Hängt eine Zeile an der aktuellen Position an.

Siehe auch Zeilenoperationen. 1743

29.3.3.3.31 RowDelete (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.DeleteRow 1778

Mit Hilfe der Tabellenoperationen von AuthenticRange kann nun jede Tabelle im aktuellen Dokument unabhängig von der aktuellen Auswahl auf der Benutzeroberfläche bearbeitet werden.

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.RowDelete();
// use now:
if(! Application.ActiveDocument.AuthenticView.Selection.DeleteRow())
   MsgBox ("Error: can't delete row");
```

Deklaration: RowDelete

Beschreibung
Löscht die aktuell ausgewählte(n) Zeile(n).

Siehe auch Zeileoperationen. 1743

29.3.3.3.32 RowDuplicate (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.DuplicateRow 1779

Mit Hilfe der Tabellenoperationen von AuthenticRange kann nun jede Tabelle im aktuellen Dokument unabhängig von der aktuellen Auswahl auf der Benutzeroberfläche bearbeitet werden.

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.RowDuplicate();
// use now:
if(! Application.ActiveDocument.AuthenticView.Selection.DuplicateRow())
   MsgBox ("Error: can't duplicate row");
```

Deklaration: RowDuplicate

Beschreibung

Die Methode kopiert die aktuell ausgewählten Zeilen.

Siehe auch Zeilenoperationen[1743].

29.3.3.3.33 RowInsert (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.InsertRow[1789]

Mit Hilfe der Tabellenoperationen von AuthenticRange kann nun jede Tabelle im aktuellen Dokument unabhängig von der aktuellen Auswahl auf der Benutzeroberfläche bearbeitet werden.

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.RowInsert();
// use now:
if(! Application.ActiveDocument.AuthenticView.Selection.InsertRow())
   MsgBox ("Error: can't insert row");
```

Deklaration: RowInsert

Beschreibung
Fügt eine neue Zeile unmittelbar oberhalb der aktuellen Auswahl ein.

Siehe auch Zeilenoperationen.[1743]

29.3.3.3.34 RowMoveDown (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.MoveRowDown[1796]

Mit Hilfe der Tabellenoperationen von AuthenticRange kann nun jede Tabelle im aktuellen Dokument unabhängig von der aktuellen Auswahl auf der Benutzeroberfläche bearbeitet werden.

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.RowMoveDown();
// use now:
if(! Application.ActiveDocument.AuthenticView.Selection.MoveRowDown())
   MsgBox ("Error: can't move row down");
```

Deklaration: RowMoveDown

Beschreibung

© 2016-2022 Altova GmbH

Verschiebt die aktuelle Zeile um eine Position nach unten.

Siehe auch Zeilenoperationen. 1743

29.3.3.3.35 RowMoveUp (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.MoveRowUp 1797

Mit Hilfe der Tabellenoperationen von AuthenticRange kann nun jede Tabelle im aktuellen Dokument unabhängig von der aktuellen Auswahl auf der Benutzeroberfläche bearbeitet werden.

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.RowAppend();
// use now:
if(! Application.ActiveDocument.AuthenticView.Selection.MoveRowUp())
   MsgBox ("Error: can't move row up");
```

Deklaration: RowMoveUp

Beschreibung
Verschiebt die aktuelle Zeile um eine Position nach oben.

Siehe auch Zeilenoperationen. 1743

29.3.3.3.36 SaveXML (nicht mehr verwendet)

Wird ersetzt durch AuthenticView.AsXMLString 1813

AuthenticView unterstützt nun die Eigenschaft XMLString, mit deren Hilfe Sie den Dokumentinhalt direkt aufrufen und als XMLString ersetzen können.

```
// ----- javascript sample -----
// instead of:
// var strDocAsXMLString = Application.ActiveDocument.DocEditView.SaveXML();
// use now:
try
{
   var strDocAsXMLString = Application.ActiveDocument.AuthenticView.AsXMLString;
   ... // do something here
}
catch (err)
```

```
{ MsgBox ("Error: invalid XML string"); }
```

Deklaration: SaveXML als String

Rückgabewert
XML-Struktur als String.

Beschreibung
Speichert die aktuellen XML-Daten als String, der an den Aufrufenden zurückgegeben wird.

29.3.3.3.37 SelectionMoveTabOrder (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.SelectNext[1799]

AuthenticRange unterstützt nun eine breite Palette an Elementnavigationsmethoden basierend auf Dokumentelementen wie Zeichen, Wörter, Tags und vieles mehr. Das Auswählen des nächsten Absatzes ist nur eine davon und die Navigation muss nicht notwendigerweise auf die aktuelle Auswahl auf der Benutzeroberfläche beschränkt sein.

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.DocEditView.SelectionMoveTabOrder(true, true);
// use now:
Application.ActiveDocument.AuthenticView.Selection.SelectNext (spyAuthenticParagraph).Select();
// to append a row to a table use AuthenticRange.AppendRow
```

Dkclaration: SelectionMoveTabOrder (*bForward* als Boolean, *bTag* als Boolean)

Beschreibung
SelectionMoveTabOrder()verschiebt die aktuelle Auswahl nach vorne oder nach hinten.

Wenn bTag "false" ist und sich die aktuelle Auswahl in der letzten Zelle einer Tabelle befindet, wird eine neue Zeile hinzugefügt.

29.3.3.3.38 SelectionSet (nicht mehr verwendet)

Wird ersetzt durch AuthenticRange.FirstXMLData[1781] und damit in Zusammenhang stehenden Eigenschaften

AuthenticRange unterstützt sowohl die Navigation über XMLData-Elemente als auch die Navigation nach Dokumentelementen (z.B. Zeichen, Wörter, Tags) oder Textcursorpositionen.

© 2016-2022 Altova GmbH

```
// ----- javascript sample -----
// instead of:
// if (! Application.ActiveDocument.DocEditView.SelectionSet(varXMLData1, 0, varXMLData2, -1))
//    MsgBox ("Error: invalid data position");
// use now:
try
{
   var objSelection = Application.ActiveDocument.AuthenticView.Selection;
   objSelection.FirstXMLData = varXMLData1;
   objSelection.FirstXMLdataOffset = 0;
   objSelection.LastXMLData = varXMLData2;
   objSelection.LastXMLDataOffset = -1;
   objSelection.Select();
}
catch(err)
   { MsgBox ("Error: invalid data position"); }
// to select all text between varXMLData1 and varXMLdata2, inclusive
```

Deklaration: SelectionSet (*pStartElement* als XMLData[1970], *nStartPos* als long, *pEndElement* als XMLData[1970], *nEndPos* als long) als Boolean

Beschreibung
Verwenden Sie SelectionSet()um in der Authentic-Ansicht eine neue Auswahl zu definieren. Sie können pEndElementauf Null (nichts) setzen, wenn die Auswahl nur ein XML-Element (pStartElement) umfassen soll.

29.3.3.3.39 XMLRoot (nicht mehr verwendet)

Wird ersetzt durch AuthenticView.XMLDataRoot[1822]

```
// ----- javascript sample -----
// instead of:
// var objXMLData = Application.ActiveDocument.DocEditView.XMLRoot;
// use now:
var objXMLData = Application.ActiveDocument.AuthenticView.XMLDataRoot;
```

Deklaration: XMLRoot als XMLData[1970]

Beschreibung
XMLRootist das übergeordnete Element der aktuell angezeigten XML-Struktur. Über die XMLData[1970]-Schnittstelle haben Sie vollen Zugriff auf den kompletten Dateiinhalt.

29.3.4 Enumerationen

Dies ist eine Liste aller von der XMLSpy API verwendeten Enumerationen. Wenn Ihre Skripting-Umgebung Enumerationen nicht unterstützt, verwenden Sie stattdessen die Zahlenwerte.

29.3.4.1 ENUMApplicationStatus

Beschreibung
Enumeration zur Definition des aktuellen Applikationsstatus.

Mögliche Werte:

eApplicationRunning	= 0
eApplicationAfterLicenseCheck	= 1
eApplicationBeforeLicenseCheck	= 2
eApplicationConcurrentLicenseCheckFailed	= 3
eApplicationProcessingCommandLine	= 4

29.3.4.2 SPYAttributeTypeDefinition

Beschreibung
Attributtypdefinition, die zur Generierung einer XML-Beispieldatei ausgewählt werden kann.
Dieser Typ wird mit den Methoden GenerateDTDOrSchema[1854] und GenerateDTDOrSchemaEx[1855] verwendet.

Mögliche Werte:

spyMergedGlobal	= 0
spyDistinctGlobal	= 1
spyLocal	= 2

29.3.4.3 SPYAuthenticActions

Beschreibung
Aktionen, die an AuthenticRange[1773] Objekten durchgeführt werden können.

Mögliche Werte:

spyAuthenticInsertAt	= 0
spyAuthenticApply	= 1
spyAuthenticClearSurr	= 2
spyAuthenticAppend	= 3
spyAuthenticInsertBefore	= 4
spyAuthenticRemove	= 5

© 2016-2022 Altova GmbH

29.3.4.4 SPYAuthenticDocumentPosition

Beschreibung
Relative und absolute Position für die Navigation mit AuthenticRange[1773] Objekten.

Mögliche Werte:

spyAuthenticDocumentBegin	= 0
spyAuthenticDocumentEnd	= 1
spyAuthenticRangeBegin	= 2
spyAuthenticRangeEnd	= 3

29.3.4.5 SPYAuthenticElementActions

Beschreibung
Aktionen die mit dem nicht mehr verwendeten Objekt GetAllowedElements verwendet werden können (Wird ersetzt durch AuthenticRange.CanPerformActionWith[1776]).

Mögliche Werte:

k_ActionInsertAt	= 0
k_ActionApply	= 1
k_ActionClearSurr	= 2
k_ActionAppend	= 3
k_ActionInsertBefore	= 4
k_ActionRemove	= 5

29.3.4.6 SPYAuthenticElementKind

Beschreibung
Enumeration der verschiedenen Arten von Elementen, die für die Navigation und Auswahl innerhalb der AuthenticRange[1773] und AuthenticView[1803] Objekte verwendet werden.

Mögliche Werte:

spyAuthenticChar	= 0
spyAuthenticWord	= 1
spyAuthenticLine	= 3
spyAuthenticParagraph	= 4
spyAuthenticTag	= 6
spyAuthenticDocument	= 8
spyAuthenticTable	= 9
spyAuthenticTableRow	= 10
spyAuthenticTableColumn	= 11

29.3.4.7 SPYAuthenticMarkupVisibility

Beschreibung
Enumerationswerte zum Anpassen der Sichtbarkeit von Markup-Tags mit MarkupVisibility[1819].

Mögliche Werte:

spyAuthenticMarkupHidden	= 0
spyAuthenticMarkupSmall	= 1
spyAuthenticMarkupLarge	= 2
spyAuthenticMarkupMixed	= 3

29.3.4.8 SPYAuthenticToolbarButtonState

Beschreibung
Die Status der Authentic-Symbolleisten-Schaltflächen werden durch die folgende Enumeration definiert:

Possible values:

authenticToolbarButtonDefault	= 0
authenticToolbarButtonEnabled	= 1
authenticToolbarButtonDisabled	= 2

29.3.4.9 SPYDatabaseKind

Beschreibung
Were zur Auswahl verschiedener Datenbankarten für den Import. Informationen zur Verwendung finden Sie unter DatabaseConnection.DatabaseKind[1831].

Mögliche Werte:

spyDB_Access	= 0
spyDB_SQLServer	= 1
spyDB_Oracle	= 2
spyDB_Sybase	= 3
spyDB_MySQL	= 4
spyDB_DB2	= 5
spyDB_Other	= 6
spyDB_Unspecified	= 7
spyDB_PostgreSQL	= 8
spyDB_iSeries	= 9

© 2016-2022 Altova GmbH

29.3.4.10 SPYDialogAction

Beschreibung
Werte zur Simulation verschiedener Interaktionen in Dialogfeldern. Informationen dazu, welche Dialogfelder zur Verfügung stehen, finden Sie unter Dialogs 1835.

Mögliche Werte:

spyDialogOK	= 0	// simulate click on OK button
spyDialogCancel	= 1	// simulate click on Cancel button
spyDialogUserInput	= 2	//show dialog and allow user interaction

29.3.4.11 SPYDOMType

Beschreibung
Enumerationswerte zum Parametrisieren der Generierung von C++ Code anhand von Schemadefinitionen.

Mögliche Werte:

spyDOMType_msxml4	= 0	wird nicht mehr verwendet
spyDOMType_xerces	= 1	
spyDOMType_xerces3	= 2	
spyDOMType_msxml6	= 3	

`spyDOMType_xerces` bezeichnet die Verwendung von Xerces 2.x; `spyDOMType_xerces3` bezeichnet die Verwendung von Xerces 3.x.

29.3.4.12 SPYDTDSchemaFormat

Beschreibung
Enumeration zum Identifizieren der verschiedenen Schemaformate.

Mögliche Werte:

spyDTD	= 0
spyW3C	= 1

29.3.4.13 SPYEncodingByteOrder

Beschreibung
Enumerationswerte zur Definition der Codierungsbytefolge für den Textimport und -export.

Mögliche Werte:

spyNONE	= 0
spyLITTLE_ENDIAN	= 1
spyBIG_ENDIAN	= 2

29.3.4.14 SPYExportNamespace

Beschreibung
Enumerationstyp, um die Behandlung von Namespace Identifiers beim Export zu konfigurieren.

Mögliche Werte:

spyNoNamespace	= 0
spyReplaceColonWithUnderscore	= 1

29.3.4.15 SPYFindInFilesSearchLocation

Beschreibung
Die verschiedenen Optionen, wo eine Suche durchgeführt werden kann. Dieser Typ wird mit dem Dialogfeld FindInFilesDlg[1885] verwendet.

Mögliche Werte:

spyFindInFiles_Documents	= 0
spyFindInFiles_Project	= 1
spyFindInFiles_Folder	= 2

29.3.4.16 SPYFrequentElements

Beschreibung
Enumerationswert zum Parametrisieren der Schemagenerierung.

Mögliche Werte:

spyGlobalElements	= 0
spyGlobalComplexType	= 1

29.3.4.17 SPYImageKind

Beschreibung
Enumerationswerte zum Parametrisieren des Bildtyps der generierten Dokumentation. Diese Werte werden in SchemaDocumentationDialog.DiagramFormat[1908] und WSDLDocumentationDlg.DiagramFormat[1936] verwendet.

Mögliche Werte:

spyImageType_PNG	= 0
spyImageType_EMF	= 1

29.3.4.18 SPYImportColumnsType

Beschreibung
Enumeration zur Definition der verschiedenen Importspaltenarten.

Mögliche Werte:

© 2016-2022 Altova GmbH

spyImportColumns_Element = 0
spyImportColumns_Attribute = 1

29.3.4.19 SPYKeyEvent

Beschreibung
Enumerationstyp zum Identifizieren der verschiedenen Tastatur-Events. Diese Events entsprechen den gleichnamigen Windows-Messages.

Mögliche Werte:
spyKeyDown = 0
spyKeyUp = 1
spyKeyPressed = 2

29.3.4.20 SPYKeyStatus

Beschreibung
Enumerationstyp zum Identifizieren des Tastenstatus.

Mögliche Werte:
spyLeftShiftKeyMask = 1
spyRightShiftKeyMask = 2
spyLeftCtrlKeyMask = 4
spyRightCtrlKeyMask = 8
spyLeftAltKeyMask = 16
spyRightAltKeyMask = 32

29.3.4.21 SPYLibType

Beschreibung
Enumerationswerte zum Parametrisieren der Generierung von C++ Code anhand von Schemadefinitionen.

Mögliche Werte:
spyLibType_static = 0
spyLibType_dll = 1

29.3.4.22 SPYLoading

Beschreibung
Enumerationswerte zum Definieren des Ladeverhaltens von URL-Dateien.

Mögliche Werte:
spyUseCacheProxy = 0
spyReload = 1

29.3.4.23 SPYMouseEvent

Beschreibung
Enumerationstyp, der die Mausstatus während eines Maus-Events definiert. Verwenden Sie diese Enumerationswerte als Bitmasken anstatt direkte Vergleiche damit durchzuführen.

Beispiele

```
' to check for ctrl-leftbutton-down in VB
If(i_eMouseEvent = (XMLSpyLib.spyLeftButtonDownMask OrXMLSpyLib.spyCtrlKeyDownMask)) Then
        ' react on ctrl-leftbutton-down
EndIf

' to check for double-click with any button in VBScript
If(((i_eMouseEvent AndspyDoubleClickMask) <> 0) Then
        ' react on double-click
EndIf
```

Mögliche Werte:

spyNoButtonMask	= 0	
spyMouseMoveMask	= 1	
spyLeftButtonMask	= 2	
spyMiddleButtonMask	= 4	
spyRightButtonMask	= 8	
spyButtonUpMask	= 16	
spyButtonDownMask	= 32	
spyDoubleClickMask	= 64	
spyShiftKeyDownMask	= 128	
spyCtrlKeyDownMask	= 256	
spyLeftButtonDownMask	= 34	// spyLeftButtonMask | spyButtonDownMask
spyMiddleButtonDownMask	= 36	// spyMiddleButtonMask | spyButtonDownMask
spyRightButtonDownMask	= 40	// spyRightButtonMask | spyButtonDownMask
spyLeftButtonUpMask	= 18	// spyLeftButtonMask | spyButtonUpMask
spyMiddleButtonUpMask	= 20	// spyMiddleButtonMask | spyButtonUpMask
spyRightButtonUpMask	= 24	// spyRightButtonMask | spyButtonUpMask
spyLeftDoubleClickMask	= 66	// spyRightButtonMask | spyButtonUpMask
spyMiddleDoubleClickMask	= 68	// spyMiddleButtonMask | spyDoubleClickMask
spyRightDoubleClickMask	= 72	// spyRightButtonMask | spyDoubleClickMask

29.3.4.24 SPYNumberDateTimeFormat

Beschreibung
Enumerationswert zum Konfigurieren von Datenbankverbindungen.

Mögliche Werte:

spySystemLocale	= 0
spySchemaCompatible	= 1

© 2016-2022 Altova GmbH

29.3.4.25 SPYProgrammingLanguage

Beschreibung
Enumerationswerte zur Auswahl der Programmiersprache für die Codegenerierung von Schemadefinitionen.

Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Mögliche Werte:

spyUndefinedLanguage	= -1
spyJava	= 0
spyCpp	= 1
spyCSharp	= 2

29.3.4.26 SPYProjectItemTypes

Beschreibung
Enumerationswerte zum Identifizieren der verschiedenen Elemente in Projektelementlisten. Siehe SpyProjectItem.ItemType[1922].

Mögliche Werte:

spyUnknownItem	= 0
spyFileItem	= 1
spyFolderItem	= 2
spyURLItem	= 3

29.3.4.27 SPYProjectType

Beschreibung
Enumerationswerte zum Parametrisieren der Generierung von C# Code anhand von Schemadefinitionen.

Mögliche Werte:

spyVisualStudioProject	= 0	Wird nicht mehr verwendet
spyVisualStudio2003Project	= 1	Wird nicht mehr verwendet
spyBorlandProject	= 2	Wird nicht mehr verwendet
spyMonoMakefile	= 3	Wird nicht mehr verwendet
spyVisualStudio2005Project	= 4	Wird nicht mehr verwendet
spyVisualStudio2008Project	= 5	Wird nicht mehr verwendet
spyVisualStudio2010Project	= 6	Auch für C++-Code
spyVisualStudio2013Project	= 7	Auch für C++-Code
spyVisualStudio2015Project	= 8	Auch für C++-Code

spyVisualStudio2017Project = 9 Auch für C++-Code
spyVisualStudio2019Project =10 Auch für C++-Code

29.3.4.28 SpySampleXMLGenerationChoiceMode

Beschreibung
Diese Enumeration wird in `GenerateSampleXMLDlg.ChoiceMode`[1897] verwendet:

spySampleXMLGen_FirstBranch = 0
spySampleXMLGen_AllBranches = 1
spySampleXMLGen_ShortestBranch = 2

29.3.4.29 SPYSampleXMLGenerationOptimization (nicht mehr verwendet)

Diese Enumeration wird seit v2014 NICHT MEHR VERWENDET.

Beschreibung
Definiert die Elemente, die in der XML-Beispieldatei generiert werden.
Diese Enumeration wird in GenerateSampleXMLDlg[1896] verwendet.

Mögliche Werte:
spySampleXMLGen_Optimized = 0
spySampleXMLGen_NonMandatoryElements = 1
spySampleXMLGen_Everything = 2

29.3.4.30 SpySampleXMLGenerationSampleValueHints

Beschreibung
Diese Enumeration wird in `GenerateSampleXMLDlg.SampleValueHints`[1901] verwendet.

spySampleXMLGen_FirstFit = 0
spySampleXMLGen_RandomFit = 1
spySampleXMLGen_CycleThrough = 2

© 2016-2022 Altova GmbH

29.3.4.31 SPYSampleXMLGenerationSchemaOrDTDAssignment

Beschreibung
Definiert, welche Art von Referenz zum Schema/zur DTD zur generierten XML-Beispieldatei hinzugefügt werden soll.
Diese Enumeration wird in GenerateSampleXMLDlg[1896] verwendet.

Mögliche Werte:

spySampleXMLGen_AssignRelatively	= 0
spySampleXMLGen_AssignAbsolutely	= 1
spySampleXMLGen_DoNotAssign	= 2

29.3.4.32 SPYSchemaDefKind

Beschreibung
Enumerationstyp zur Auswahl von Schemadiagrammtypen.

Mögliche Werte:

spyKindElement	= 0
spyKindComplexType	= 1
spyKindSimpleType	= 2
spyKindGroup	= 3
spyKindModel	= 4
spyKindAny	= 5
spyKindAttr	= 6
spyKindAttrGroup	= 7
spyKindAttrAny	= 8
spyKindIdentityUnique	= 9
spyKindIdentityKey	= 10
spyKindIdentityKeyRef	= 11
spyKindIdentitySelector	= 12
spyKindIdentityField	= 13
spyKindNotation	= 14
spyKindInclude	= 15
spyKindImport	= 16
spyKindRedefine	= 17
spyKindFacet	= 18
spyKindSchema	= 19
spyKindCount	= 20

29.3.4.33 SPYSchemaDocumentationFormat

Beschreibung
Enumerationswerte zum Parametrisieren der Generierung von Schemadokumentation. Diese Werte werden in SchemaDocumentationDialog.OutputFormat[1914] und WSDLDocumentationDlg.OutputFormat[1941] verwendet.

Mögliche Werte:

spySchemaDoc_HTML	= 0
spySchemaDoc_MSWord	= 1
spySchemaDoc_RTF	= 2
spxSchemaDoc_PDF	=3

29.3.4.34 SPYSchemaExtensionType

Beschreibung
Enumeration zur Definition der verschiedenen Schema Extension-Typen.

Mögliche Werte:

spySchemaExtension_None	= 0
spySchemaExtension_SQL_XML	= 1
spySchemaExtension_MS_SQL_Server	= 2
spySchemaExtension_Oracle	= 3

29.3.4.35 SPYSchemaFormat

Beschreibung
Enumeration zur Definition der unterschiedlichen Schemaformattypen.

Mögliche Werte:

spySchemaFormat_Hierarchical	= 0
spySchemaFormat_Flat	= 1

29.3.4.36 SPYTextDelimiters

Beschreibung
Enumerationswerte zum Definieren der Texttrennzeichen für den Textexport.

Mögliche Werte:

spyTabulator	= 0
spySemicolon	= 1
spyComma	= 2
spySpace	= 3

© 2016-2022 Altova GmbH

29.3.4.37 SPYTextEnclosing

Beschreibung
Enumerationswert zum Definieren der den Text einschließenden Zeichen beim Textimport und -export.

Mögliche Werte:

spyNoEnclosing	= 0
spySingleQuote	= 1
spyDoubleQuote	= 2

29.3.4.38 SPYTypeDetection

Beschreibung
Enumeration, um auszuwählen, wie die Typerkennung bei `GenerateDTDOrSchema`[1854] und `GenerateDTDOrSchemaEx`[1855] funktioniert.

Mögliche Werte:

spyBestPossible	= 0
spyNumbersOnly	= 1
spyNoDetection	= 2

29.3.4.39 SPYURLTypes

Beschreibung
Enumeration zum Definieren der verschiedenen URL-Typen.

Mögliche Werte:

spyURLTypeAuto	= -1
spyURLTypeXML	= 0
spyURLTypeDTD	= 1

29.3.4.40 SPYValidateXSDVersion

Beschreibung
Enumerationswerte zur Auswahl der zu verwendenden XSD-Version. Welche XSD-Version ausgewählt wird, hängt (i) vom Vorhandensein/Fehlen - und falls vorhanden, vom Wert - des `/xs:schema/@vc:minVersion` Attributs des XSD-Dokuments ab und (ii) dem Wert dieser Enumeration.

`spyValidateXSDVersion_1_0` wählt `XSD 1.0` aus, wenn `vc:minVersion` fehlt oder mit einem beliebigen Wert vorhanden ist.
`spyValidateXSDVersion_1_1` wählt `XSD 1.1` aus, wenn `vc:minVersion` fehlt oder mit einem beliebigen Wert vorhanden ist.
`spyValidateXSDVersion_AutoDetect` wählt `XSD 1.1` aus, wenn `vc:minVersion=1.1`. Wenn das `vc:minVersion` Attribut fehlt oder vorhanden ist und einen beliebigen anderen Wert als `1.1`, hat, so wird `XSD 1.0` ausgewählt.

Mögliche Werte

```
spyValidateXSDVersion_AutoDetect      = 0
spyValidateXSDVersion_1_1             = 1
spyValidateXSDVersion_1_0             = 2
```

29.3.4.41 SPYValidateErrorFormat

Beschreibung
Enumerationswerte zur Auswahl des Formats der Fehlermeldung.

Mögliche Werte

```
spyValidateErrorFormat_Text           = 0
spyValidateErrorFormat_ShortXML       = 1
spyValidateErrorFormat_LongXML        = 2
```

29.3.4.42 SPYViewModes

Beschreibung
Enumerationswerte, die die verschiedenen Ansichtsmodi für XML-Dokumente definieren. Der Modus *spyViewAuthentic(4)* identifiziert den Modus, der zwischenzeitlich DocEdit-Modus genannt wurde und nun Authentic-Modus heißt. Der Modus *spyViewJsonSchema* identifiziert einen Modus, der auf der Benutzeroberfläche der Schema-Design-Ansicht zugeordnet ist, intern aber unterschiedlich behandelt wird.

Mögliche Werte:

```
spyViewGrid          = 0
spyViewText          = 1
spyViewBrowser       = 2
spyViewSchema        = 3
spyViewContent       = 4     // nicht mehr verwendet
spyViewAuthentic     = 4
spyViewWSDL          = 5
spyViewZIP           = 6
spyViewEditionInfo   = 7
spyViewXBRL          = 8
spyViewJsonSchema    = 9
```


© 2016-2022 Altova GmbH

29.3.4.43 SPYVirtualKeyMask

Beschreibung
Enumeratiostyp für die am häufigsten verwendeten Tastaturmasken, die den Status der virtuellen Tasten identifizieren. Verwenden Sie diese Werte als Bitmasken, anstatt direkte Vergleiche damit durchzuführen. Wenn nötig können Sie mit Hilfe des 'logical or' Operators weitere Masken erstellen.

Beispiele

```
'  VBScript sample: check if ctrl-key is pressed
If((i_nVirtualKeyStatus AndspyCtrlKeyMask) <> 0)) Then
        ' ctrl-key is pressed
EndIf

'  VBScript sample: check if ONLY ctrl-key is pressed
If(i_nVirtualKeyStatus == spyCtrlKeyMask) Then
        ' exactly ctrl-key is pressed
EndIf

// JScript sample: check if any of the right virtual keys is pressed
if((i_nVirtualKeyStatus & (spyRightShiftKeyMask | spyRightCtrlKeyMask | spyRightAltKeyMask)) != 0)
{
        ; ' right virtual key is pressed
}
```

Mögliche Werte:

spyNoVirtualKeyMask	= 0	
spyLeftShiftKeyMask	= 1	
spyRightShiftKeyMask	= 2	
spyLeftCtrlKeyMask	= 4	
spyRightCtrlKeyMask	= 8	
spyLeftAltKeyMask	= 16	
spyRightAltKeyMask	= 32	
spyShiftKeyMask	= 3	// spyLeftShiftKeyMask | spyRightShiftKeyMask
spyCtrlKeyMask	= 12	// spyLeftCtrlKeyMask | spyRightCtrlKeyMask
spyAltKeyMask	= 48	// spyLeftAltKeyMask | spyRightAltKeyMask

29.3.4.44 SPYXMLDataKind

Beschreibung
Die verschiedenen Typen von XMLData-Elementen, die für XML-Dokumente zur Verfügung stehen.

Mögliche Werte:

spyXMLDataXMLDocStruct	= 0
spyXMLDataXMLEntityDocStruct	= 1
spyXMLDataDTDDocStruct	= 2
spyXMLDataXML	= 3
spyXMLDataElement	= 4
spyXMLDataAttr	= 5

spyXMLDataText	= 6
spyXMLDataCData	= 7
spyXMLDataComment	= 8
spyXMLDataPI	= 9
spyXMLDataDefDoctype	= 10
spyXMLDataDefExternalID	= 11
spyXMLDataDefElement	= 12
spyXMLDataDefAttlist	= 13
spyXMLDataDefEntity	= 14
spyXMLDataDefNotation	= 15
spyXMLDataKindsCount	= 16

29.3.5 Applikations-API für Java (nicht mehr verwendet)

Die in diesem Abschnitt (Applikation API für Java) beschriebenen Objekte werden ab v2012 nicht mehr verwendet.

Informationen zum Aufrufen der Application API von Java-Code aus finden Sie im Abschnitt: Programmiersprachen | Java[1732].

Die Applikations-API in Java hat eine Benutzeroberfläche, die auf Java-Klassen aufbaut, wobei jede dieser Klassen einem Objekt in der Applikations-API[1706] entspricht. Entwickler können mit Hilfe dieser Java-Klassen mit der COM API interagieren. Diese Klassen sind unten aufgelistet und in den Unterabschnitten näher beschrieben. Eine Beschreibung der Applikations-API[1706]-Objekte selbst finden Sie in der Dokumentation zur Applikations-API[1706]. Beachten Sie, dass einige API-Funktionen nur in Skripting-Umgebungen zur Verfügung stehen; diese wurden daher nicht nach Java portiert.

Java-Klassen

- SpyApplication[2037]
  - SpyProject[2048]
    - SpyProjectItems[2048]
      - SpyProjectItem[2048]
  - SpyDocuments[2042]
    - SpyDoc[2041]
      - SpyAuthenticView[2060]
        - SpyAuthenticRange[2059]
      - SpyDocEditView[2061]
        - SpyDocEditSelection[2061]
      - SpyGridView[2048]
      - SpyTextView[2052]
      - SpyXMLData[2058]
  - SpyDialogs[2040]
    - SpyCodeGeneratorDlg[2038]
    - SpyDTDSchemaGeneratorDlg[2043]
    - SpyFileSelectionDlg[2045]
    - SpyFindInFilesDlg[2045]

© 2016-2022 Altova GmbH

SpyGenerateSampleXMLDlg 2047
SpySchemaDocumentationDlg 2049
SpyWSDL20DocumentationDlg 2052
SpyWSDLDocumentationDlg 2054
SpyXBRLDocumentationDlg 2056
SpyDatabaseConnection 2039
SpyElementList 2044
SpyElementListItem 2044
SpyExportSettings 2044
SpyFindInFilesResults 2047
SpyFindInFilesResult 2046
SpyFindInFilesMatch 2046
SpyTextImportExportSettings 2051

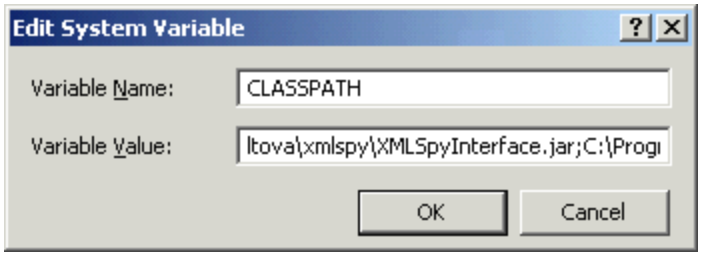
Implementierung von COM-Eigenschaften in Java
Die Eigenschaften in Java wurden so definiert, dass Sie sowohl eine **`set`** als auch eine **`get`** Methode enthalten (`set` wenn dies die COM-Implementierung zulässt). So enthält z.B. die COM-Klasse `Document` die Eigenschaft **`GridView`**. In Java heißt die Methode `SpyDoc` und die Eigenschaft ist als eine **`GetGridView`** Methode definiert.

Überprüfen Sie bei Kompilierproblemen die folgenden Punkte:

- Die `xmlspylib.dll` muss in `..\windows\system32` verfügbar sein.
- Die Datei `XMLSpyInterface.jar` muss in die ClassPath-Umgebungsvariable eingesetzt werden.

Setzen der `ClassPath` Variable in Windows XP

1. Klicken Sie auf **Start | Einstellungen | Systemsteuerung | System | Erweitert | Umgebungsvariablen**. Daraufhin wird das Dialogfeld "Umgebungsvariablen" geöffnet.
2. Wenn in der Gruppe der Systemvariablen bereits ein `ClassPath` Eintrag vorhanden ist, wählen Sie den Eintrag `ClassPath` aus und klicken Sie auf die Schaltfläche **Bearbeiten.** Bearbeiten Sie den Pfad zu: `"C:\Program Files\Altova\xmlspy\XMLSpyInterface.jar"`.



Wenn in der Gruppe der Systemvarialblen noch kein Eintrag `ClassPath` vorhanden ist, klicken Sie auf die Schaltfläche **Neu**. Daraufhin erscheint das Dialogfeld "Neue Systemvariable". Geben Sie als Name der Variable `CLASSPATH` ein und `"C:\Program Files\Altova\xmlspy\XMLSpyInterface.jar"` als die `ClassPath` Variable (ändern Sie gegebenenfalls den Pfad zu ihrer Installation).

29.3.5.1 Beispielquellcode (nicht mehr verwendet)

Der Befehlsparameter "`SpyDoc doc = app.GetDocuments().OpenFile(...)`" muss für Ihre Umgebung angepasst werden.

Das Beispiel führt folgende Schritte durch:

- Es startet eine neue XMLSpy-Instanz
- Es öffnet die Datei Datasheet.xml (ändern Sie den Pfad hier...)
- Es wechselt in die Grid-Ansicht
- Es hängt ein neues Child-Element namens "NewChild" mit dem Textwertelement "NewValuE" an das Root-Element an
- Es überprüft, ob das Dokument gültig ist und gibt eine Meldung an die Java-Konsole aus
- Es schließt die XMLSpy-Applikation und gibt diese frei

```
import XMLSpyInterface.*;

public class TestSpyInterface
{
    public TestSpyInterface() {}

    public static void main(String[] args)
    {
        SpyApplication app = null;
        SpyDoc oDoc = null;
        SpyXMLData oData = null;
        SpyXMLData oNewChild = null;

        try
        {
            app = new SpyApplication();
            app.ShowApplication( true );

            oDoc = app.GetDocuments().OpenFile("C:\\FilePath\\OrgChart.xml", true );

            // OrgChart.xml is in the folder C:\Dokumente und Einstellungen\<Benutzer>\Eigene
            Dokumente\Altova\XMLSpy2023. The filepath should be in
            // the form: C:\\Documents and Settings\\Username\\Folder\\Filename.xml

            if ( oDoc != null )
            {
                oDoc.SwitchViewMode(SPYViewModes.spyViewGrid);
                oData = oDoc.GetRootElement();
                oNewChild = oDoc.CreateChild(SPYXMLDataKind.spyXMLDataElement);

                oNewChild.SetName( "NewChild" );
                oNewChild.SetTextValue("newVaLuE");
                oData.AppendChild(oNewChild);

                if ( oDoc.IsValid() == false )
                {
                    // is to be expected after above insertion
```

 © 2016-2022 Altova GmbH

```
                System.out.println( "!!!!!!validation error: " + oDoc.GetErrorString() );
                System.out.println( "!!!!!!validation error: " + oDoc.GetErrorPos() );
                System.out.println( "!!!!!!validation error: " + oDoc.GetBadData() );

            }
        }

        app.Quit();
    }
    finally
    {
        // Free any allocated resources by calling ReleaseInstance().
        if ( oNewChild != null )
            oNewChild.ReleaseInstance();

        if ( oData != null )
            oData.ReleaseInstance();

        if ( oDoc != null )
            oDoc.ReleaseInstance();

        if ( app != null )
            app.ReleaseInstance();
    }
  }
}
```

Wenn Sie Probleme haben, dieses Beispiel zu kompilieren, versuchen Sie bitte in der Befehlszeile (**Start | Ausführen | cmd**) die folgenden Befehle. Stellen Sie vorher sicher, dass Sie sich gerade in dem Ordner befinden, der die Java-Beispieldatei enthält.

Kompilierung
javac -classpath c:\IhrPfadhier\XMLSpyInterface.jar *testspyinterface.java*

Ausführung
java -classpath c:\IhrPfadhier\XMLSpyInterface.jar *testspyinterface*

29.3.5.2 SpyApplication (nicht mehr verwendet)

```
public class SpyApplication
{
  public void ReleaseInstance2036();
  public void ShowApplication1763( boolean bShow );
  public void Quit1761();
  public void AddMacroMenuItem1750( String sMacro, String sDisplayText );
  public void ClearMacroMenu1751();
  public SpyDoc2041 GetActiveDocument1749();
  public SpyProject2048 GetCurrentProject1751();
  public SpyDocuments2042 GetDocuments1752();
  public SpyElementList2044 GetDatabaseImportElementList1753( SpyDatabaseConnection2039
  oImportSettings );
  public SpyDatabaseConnection2039 GetDatabaseSettings1753();
```

```
  public SpyElementList GetDatabaseTables( SpyDatabaseConnection oImportSettings );
  public SpyExportSettings GetExportSettings();
  public SpyElementList GetTextImportElementList( SpyTextImportExportSettings
  oImportSettings );
  public SpyTextImportExportSettings GetTextImportExportSettings();
  public SpyDoc ImportFromDatabase( SpyDatabaseConnection oImportSettings,
  SpyElementList oElementList );
  public SpyDoc ImportFromSchema( SpyDatabaseConnection oImportSettings, String
  strTable, SpyDoc oSchemaDoc );
  public SpyDoc ImportFromText( SpyTextImportExportSettings oImportSettings,
  SpyElementList oElementList );
  public SpyDoc ImportFromWord( String sFile );
  public void NewProject( String sPath, boolean bDiscardCurrent );
  public void OpenProject(String sPath , boolean bDiscardCurrent, boolean bDialog );
  public long ShowForm( String sName );
  public void URLDelete( String sURL, String sUser, String sPassword );
  public void URLMakeDirectory( String sURL, String sUser, String sPassword );
  public int GetWarningNumber();
  public String GetWarningText();

  // since Version 2004R4
  public SpyApplication GetApplication();
  public SpyApplication GetParent();
  public SpyDialogs GetDialogs();
  public boolean GetVisible();
  public void SetVisible( boolean i_bVisibility );
  public long GetWindowHandle();

  public void ReloadSettings();
  public SpyFindInFilesResults FindInFiles( SpyFindInFilesDlg dlgSettings );
  public boolean ShowFindInFiles( SpyFindInFilesDlg dlgSettings );
  public void Selection( String sVal );

  public long Status();
  public int MajorVersion();
  public int MinorVersion();
  public String Edition();
  public boolean IsAPISupported();
  public long ServicePackVersion();
  public void CreateXMLSchemaFromDBStructure( SpyDatabaseConnection oConnection,
SpyElementList oTables );
}
```

29.3.5.3 SpyCodeGeneratorDlg (nicht mehr verwendet)

Steht nur in der Enterprise Edition zur Verfügung. Bei Aufruf über eine andere Version wird eine Fehlermeldung ausgegeben.

```
// since version 2004R4
public class SpyCodeGeneratorDlg
{
  public void ReleaseInstance();
```

```
  public SpyApplication[2037] GetApplication[1823]();
  public SpyDialogs[2040] GetParent[1827]();
  public long[2067] GetProgrammingLanguage[1827]();
  public void SetProgrammingLanguage[1827]( long[2067] i_eVal );
  public String GetTemplateFileName[1828]();
  public void SetTemplateFileName[1828]( String i_strVal );
  public String GetOutputPath[1826]();
  public void SetOutputPath[1826]( String i_strVal );
  public long[2065] GetOutputPathDialogAction[1826]();
  public void SetOutputPathDialogAction[1826]( long[2065] i_eVal );
  public long[2065] GetPropertySheetDialogAction[1828]();
  public void SetPropertySheetDialogAction[1828]( long[2065] i_eVal );
  public long[2065] GetOutputResultDialogAction[1827]();
  public void SetOutputResultDialogAction[1827]( long[2065] i_eVal );
  public long[2065] GetCPPSettings_DOMType[1824]();
  public void SetCPPSettings_DOMType[1824]( long[2065] i_eVal );
  public long[2067] GetCPPSettings_LibraryType[1825]();
  public void SetCPPSettings_LibraryType[1825]( long[2067] i_eVal );
  public boolean GetCPPSettings_UseMFC[1825]();
  public void SetCPPSettings_UseMFC[1825]( boolean i_bVal );
  public long[2068] GetCSharpSettings_ProjectType[1826]();
  public void SetCSharpSettings_ProjectType[1826]( long[2068] i_eVal );
}
```

29.3.5.4 SpyDatabaseConnection (nicht mehr verwendet)

```
public class SpyDatabaseConnection
{
  public void ReleaseInstance[2036]();
  public String GetADOConnection[1829]();
  public void SetADOConnection[1829]( String sValue );
  public boolean GetAsAttributes[1830]();
  public void SetAsAttributes[1830]( boolean bValue );
  public boolean GetCreateMissingTables[1830]();
  public void SetCreateMissingTables[1830]( boolean bValue );
  public boolean GetCreateNew[1830]();
  public void SetCreateNew[1830]( boolean bValue );
  public boolean GetExcludeKeys[1831]();
  public void SetExcludeKeys[1831]( boolean bValue );
  public String GetFile[1831]();
  public void SetFile[1831]( String sValue );
  public boolean GetIncludeEmptyElements[1832]();
  public void SetIncludeEmptyElements[1832]( boolean bValue );
  public long[2067] GetNumberDateTimeFormat[1833]();
  public void SetNumberDateTimeFormat[1833]( long[2067] nValue );
  public String GetODBCConnection[1833]();
  public void SetODBCConnection[1833]( String sValue );
```

```
  public String GetSQLSelect();
  public void SetSQLSelect( String sValue );
  public long GetTextFieldLen();
  public void SetTextFieldLen( long nValue );

  // since version 2004R4
  public long GetDatabaseKind();
  public void SetDatabaseKind( long nValue );

  // since version 2008R2
  public boolean GetCommentIncluded();
  public void SetCommentIncluded( boolean bValue );
  public String GetNullReplacement();
  public void SetNullReplacement( String sValue );
  public String GetDatabaseSchema();
  public void SetDatabaseSchema( String sValue );

  // since version 2010r3
  public boolean GetPrimaryKeys()
  public void SetPrimaryKeys( boolean bValue )
  public boolean GetForeignKeys()
  public void SetForeignKeys( boolean bValue )
  public boolean GetUniqueKeys()
  public void SetUniqueKeys( boolean bValue )
  public long GetSchemaExtensionType()
  public void SetSchemaExtensionType( long nValue )
  public long GetSchemaFormat()
  public void SetSchemaFormat( long nValue )
  public long GetImportColumnsType()
  public void SetImportColumnsType( long nValue )
}
```

29.3.5.5 SpyDialogs (nicht mehr verwendet)

```
// Since version 2004R4
public class SpyDialogs
{
  public SpyApplication GetApplication();
  public SpyApplication GetParent();
  public SpyCodeGeneratorDlg GetCodeGeneratorDlg();
  public SpyFileSelectionDlg GetFileSelectionDlg();
  public SpySchemaDocumentationDlg GetSchemaDocumentationDlg();
  public SpyGenerateSampleXMLDlg GetGenerateSampleXMLDlg();
  public SpyDTDSchemaGeneratorDlg GetDTDSchemaGeneratorDlg();
  public SpyFindInFilesDlg GetFindInFilesDlg();
  public SpyWSDLDocumentationDlg GetWSDLDocumentationDlg();

  // Since version 2010
  public SpyWSDL20DocumentationDlg GetWSDL20DocumentationDlg();
  public SpyXBRLDocumentationDlg GetXBRLDocumentationDlg();
```

```
}
```

29.3.5.6 SpyDoc (nicht mehr verwendet)

```
public class SpyDoc
{
  public void ReleaseInstance();
  public void SetEncoding( String strEncoding );
  public void SetPathName( String strPath );
  public String GetPathName();
  public String GetTitle();
  public boolean IsModified();
  public void Save();
  public void Close( boolean bDiscardChanges );
  public void UpdateViews();
  public long GetCurrentViewMode();
  public boolean SwitchViewMode( long nMode );
  public SpyGridView GetGridView();
  public void SetActiveDocument();
  public void StartChanges();
  public void EndChanges();
  public void TransformXSL();
  public void AssignDTD( String sDTDFile, boolean bDialog );
  public void AssignSchema( String sSchemaFile, boolean bDialog );
  public void AssignXSL( String sXSLFile, boolean bDialog );
  public void ConvertDTDOrSchema( long nFormat, long nFrequentElements );
  public SpyXMLData CreateChild( long nKind );
  public void CreateSchemaDiagram( long nKind, String sName, String sFile );
  public SpyDocEditView GetDocEditView();
  public void ExportToDatabase( SpyXMLData oFromChild, SpyExportSettings
  oExportSettings, SpyDatabaseConnection oDatabaseConnection );
  public void ExportToText( SpyXMLData oFromChild, SpyExportSettings
  oExportSettings, SpyTextImportExportSettings oTextSettings );
  public void GenerateDTDOrSchema( long nFormat, int nValuesList, long nDetection,
  long nFrequentElements );
  public SpyElementList GetExportElementList( SpyXMLData oFromChild,
  SpyExportSettings oExportSettings );
  public SpyXMLData GetRootElement();
  public String SaveInString( SpyXMLData oData, boolean bMarked );
  public void SaveToURL( String sUrl, String sUser, String sPassword );
  public String GetErrorString(); // See IsValid() or IsWellFormed()
  public int GetErrorPos(); // See IsValid() or IsWellFormed()
  public SpyXMLData GetBadData(); // See IsValid() or IsWellFormed()
  public boolean IsValid();
  public boolean IsWellFormed( SpyXMLData oData, boolean bWithChildren );

  // Since version 2004R3
  public SpyAuthenticView GetAuthenticView()

  // Since version 2004R4
```

```
  public SpyApplication[2037] GetApplication[1843]();
  public SpyDocuments[2042] GetParent[1862]();
  public String GetFullName[1854]();
  public void SetFullName[1854]( String i_strName );
  public String GetName[1862]();
  public String GetPath[1863]();
  public boolean GetSaved[1864]();
  public void SaveAs[1864]( String i_strFileNameOrPath );
  public String GetEncoding[1850]();
  public SpyXMLData[2058] GetDataRoot[1849]();
  public void GenerateProgramCode[1855]( SpyCodeGeneratorDlg[2038] i_dlg );
  public void AssignXSLFO[1844]( String i_strFile, boolean i_bUseDialog );
  public void TransformXSLFO[1868]();
  public void GenerateSchemaDocumentation[1856]( SpySchemaDocumentationDlg[2049] i_dlg );

  public void ExecuteXQuery[1851]( String i_strXMLSourceFile );
  public void SetExternalIsValid[1866]( boolean bIsValid );
  public SpyDoc[2041] GenerateSampleXML[1855]( SpyGenerateSampleXMLDlg[2047] ipGenerateXMLDlg );
  public boolean UpdateXMLData[1869]();
  public String GetAsXMLString[1845]();
  public void SetAsXMLString[1845]( String newVal );
  public SpyDoc[2041] GenerateDTDOrSchemaEx[1855]( SpyDTDSchemaGeneratorDlg[2043]
  ipDTDSchemaGeneratorDlg );
  public SpyDoc[2041] ConvertDTDOrSchemaEx[1847]( long[2065] nFormat, long[2066] nFrequentElements,
  String sOutputPath, long[2065] nOutputPathDialogAction );
  public SpyTextView[2052] GetTextView[1867]();
  public String[] GetSuggestions[1867]();
  public void SetSuggestions[1867]( String[] aList );
  public void SetSelection( String sVal );

  // Since version 2009
  public void GenerateWSDLDocumentation[1857]( SpyWSDLDocumentationDlg[2054] ipWSDLDocumenationDlg
  );
  public void TransformXSLEx[1868]( long[2065] nDialogAction );

  // Since version 2010
  public void GenerateWSDL20Documentation[1856]( SpyWSDL20DocumentationDlg[2052]
  ipWSD20DocumenationDlg );
  public void GenerateXBRLDocumentation[1857]( SpyXBRLDocumentationDlg[2056]
  ipXBRLDocumentationDlg );
  public SpyDoc[2041] ConvertToWSDL20[1847]( String sFilePath, boolean bShowDialogs );

  // Since version 2010r3
  public String CreateDBStructureFromXMLSchema[1848]( SpyDatabaseConnection[2039] oConnection,
  SpyElementList[2044] oTables, boolean bDropTableWithExistingName );
  public SpyElementList[2044] GetDBStructureList[1858]( SpyDatabaseConnection[2039] oConnection );
}
```

29.3.5.7 SpyDocuments (nicht mehr verwendet)

```
public class SpyDocuments
```

```
{
  public void ReleaseInstance();
  public long Count();
  public SpyDoc GetItem( long nNo );
  public SpyDoc NewFile( String strFile, String strType );
  public SpyDoc NewFileFromText( String nSource, String strType );
  public SpyDoc OpenFile( String sPath, boolean bDialog );
  public SpyDoc OpenURL( String sUrl, long nURLType, long nLoading, String
  sUser, String sPassword );
  public SpyDoc OpenURLDialog(String sURL, long nURLType, long nLoading, String
  sUser, String sPassword );
  // Since version 2011r2
  public SpyDoc NewAuthenticFile( String strSPSPath, String strXMLPath );
  public SpyDoc OpenAuthenticFile( String strSPSPath, String strXMLPath );
}
```

29.3.5.8 SpyDTDSchemaGeneratorDlg (nicht mehr verwendet)

```
public class SpyDTDSchemaGeneratorDlg
{
  public void ReleaseInstance();
  public SpyApplication GetApplication();
  public long GetDTDSchemaFormat();
  public void SetDTDSchemaFormat( long newVal );
  public short GetValueList();
  public void SetValueList( short newVal );
  public long GetTypeDetection();
  public void SetTypeDetection( long newVal );
  public long GetFrequentElements();
  public void SetFrequentElements( long newVal );
  public boolean GetMergeAllEqualNamed();
  public void SetMergeAllEqualNamed( boolean newVal );
  public boolean GetResolveEntities();
  public void SetResolveEntities( boolean newVal );
  public long GetAttributeTypeDefinition();
  public void SetAttributeTypeDefinition( long newVal );
  public boolean GetGlobalAttributes();
  public void SetGlobalAttributes( boolean newVal );
  public boolean GetOnlyStringEnums();
  public void SetOnlyStringEnums( boolean newVal );
  public long GetMaxEnumLength();
  public void SetMaxEnumLength( long newVal );
  public String GetOutputPath();
  public void SetOutputPath( String newVal );
  public long GetOutputPathDialogAction();
  public void SetOutputPathDialogAction( long newVal );
}
```

29.3.5.9 SpyElementList (nicht mehr verwendet)

```
public class SpyElementList
{
  public void ReleaseInstance();
  public long GetCount();
  public SpyElementListItem GetItem( long nIndex );
  public void RemoveElement( long nIndex );
}
```

29.3.5.10 SpyElementListItem (nicht mehr verwendet)

```
public class SpyElementListItem
{
  public void ReleaseInstance();
  public long GetElementKind();
  public void SetElementKind( long nKind );
  public long GetFieldCount();
  public String GetName();
  public long GetRecordCount();
}
```

29.3.5.11 SpyExportSettings (nicht mehr verwendet)

```
public class SpyExportSettings
{
  public void ReleaseInstance();
  public boolean GetCreateKeys();
  public void SetCreateKeys( boolean bValue );
  public SpyElementList GetElementList();
  public void SetElementList( SpyElementList obj );
  public boolean GetEntitiesToText();
  public void SetEntitiesToText( boolean bValue );
  public boolean GetExportAllElements();
  public void SetExportAllElements( boolean bValue );
  public boolean GetFromAttributes();
  public void SetFromAttributes( boolean bValue );
  public boolean GetFromSingleSubElements();
  public void SetFromSingleSubElements( boolean bValue );
  public boolean GetFromTextValues();
  public void SetFromTextValues( boolean bValue );
  public boolean GetIndependentPrimaryKey();
  public void SetIndependentPrimaryKey( boolean bValue );
  public long GetNamespace();
  public void SetNamespace( long nValue );
  public int GetSubLevelLimit();
```


© 2016-2022 Altova GmbH

```
  public void SetSubLevelLimit( int nValue );
}
```

29.3.5.12 SpyFileSelectionDlg (nicht mehr verwendet)

```
// Since version 2004R4
public class SpyFileSelectionDlg
{
  public void ReleaseInstance();
  public SpyApplication GetApplication();
  public SpyDialogs GetParent();
  public String GetFullName();
  public void SetFullName( String i_strName );
  public long GetDialogAction();
  public void SetDialogAction( long i_eAction );
}
```

29.3.5.13 SpyFindInFilesDlg (nicht mehr verwendet)

```
public class SpyFindInFilesDlg
{
  public void ReleaseInstance();
  public SpyApplication GetApplication();
  public String GetFind();
  public void SetFind( String sNewVal );
  public boolean GetRegularExpression();
  public void SetRegularExpression( boolean bNewVal );
  public String GetReplace();
  public void SetReplace( String sNewVal );
  public boolean GetReplaceOnDisk();
  public void SetReplaceOnDisk( boolean bNewVal );
  public boolean GetDoReplace();
  public void SetDoReplace( boolean bNewVal );
  public boolean GetMatchWholeWord();
  public void SetMatchWholeWord( boolean bNewVal );
  public boolean GetMatchCase();
  public void SetMatchCase( boolean bNewVal );
  public long GetSearchLocation();
  public void SetSearchLocation( long nPosition );
  public String GetStartFolder();
  public void SetStartFolder( String sNewVal );
  public boolean GetIncludeSubfolders();
  public void SetIncludeSubfolders( boolean bNewVal );
  public boolean GetSearchInProjectFilesDoExternal();
  public void SetSearchInProjectFilesDoExternal( boolean bNewVal );
  public String GetFileExtension();
  public void SetFileExtension( String sNewVal );
  public boolean GetAdvancedXMLSearch();
```

```
  public void SetAdvancedXMLSearch( boolean bNewVal );
  public boolean GetXMLElementNames();
  public void SetXMLElementNames( boolean bNewVal );
  public boolean GetXMLElementContents();
  public void SetXMLElementContents( boolean bNewVal );
  public boolean GetXMLAttributeNames();
  public void SetXMLAttributeNames( boolean bNewVal );
  public boolean GetXMLAttributeContents();
  public void SetXMLAttributeContents( boolean bNewVal );
  public boolean GetXMLComments();
  public void SetXMLComments( boolean bNewVal );
  public boolean GetXMLCData();
  public void SetXMLCData( boolean bNewVal );
  public boolean GetXMLPI();
  public void SetXMLPI( boolean bNewVal );
  public boolean GetXMLRest();
  public void SetXMLRest( boolean bNewVal );
  public boolean GetShowResult();
  public void SetShowResult( boolean bNewVal );
}
```

29.3.5.14 SpyFindInFilesMatch (nicht mehr verwendet)

```
public class SpyFindInFilesMatch
{
  public void ReleaseInstance();
  public long Line();
  public long Position();
  public long Length();
  public String LineText();
  public boolean Replaced();
}
```

29.3.5.15 SpyFindInFilesResult (nicht mehr verwendet)

```
public class SpyFindInFilesResult
{
  public void ReleaseInstance();
  public long Count();
  public SpyFindInFilesMatch GetItem( long nNo );
  public String GetPath();
  public SpyDoc GetDocument();
}
```


© 2016-2022 Altova GmbH

29.3.5.16 SpyFindInFilesResults (nicht mehr verwendet)

```
public class SpyFindInFilesResults
{
  public void ReleaseInstance();
  public long Count();
  public SpyFindInFilesResult GetItem( long nNo );
}
```

29.3.5.17 SpyGenerateSampleXMLDlg (nicht mehr verwendet)

```
public class SpyGenerateSampleXMLDlg
{
  public void ReleaseInstance();
  public SpyApplication GetApplication();
  public boolean GetNonMandatoryAttributes();
  public void SetNonMandatoryAttributes( boolean newVal );
  public boolean GetNonMandatoryElements();
  public void SetNonMandatoryElements( boolean newVal );
  public boolean GetTakeFirstChoice();
  public void SetTakeFirstChoice( boolean newVal );
  public long GetRepeatCount();
  public void SetRepeatCount( long newVal );
  public boolean GetFillWithSampleData();
  public void SetFillWithSampleData( boolean newVal );
  public boolean GetFillElementsWithSampleData();
  public void SetFillElementsWithSampleData( boolean newVal );
  public boolean GetFillAttributesWithSampleData();
  public void SetFillAttributesWithSampleData( boolean newVal );
  public boolean GetContentOfNillableElementsIsNonMandatory();
  public void SetContentOfNillableElementsIsNonMandatory( boolean newVal );
  public boolean GetTryToUseNonAbstractTypes();
  public void SetTryToUseNonAbstractTypes( boolean newVal );
  public long GetOptimization();
  public void SetOptimization( long newVal );
  public long GetSchemaOrDTDAssignment();
  public void SetSchemaOrDTDAssignment( long newVal );
  public String GetLocalNameOfRootElement();
  public void SetLocalNameOfRootElement( String newVal );
  public String GetNamespaceURIOfRootElement();
  public void SetNamespaceURIOfRootElement( String newVal );
  public long GetOptionsDialogAction();
  public void SetOptionsDialogAction( long newVal );
}
```

29.3.5.18 SpyGridView (nicht mehr verwendet)

```
public class SpyGridView
{
  public void ReleaseInstance();
  public SpyXMLData GetCurrentFocus();
  public void Deselect( SpyXMLData oData );
  public boolean GetIsVisible();
  public void Select( SpyXMLData oData );
  public void SetFocus( SpyXMLData oData );
}
```

29.3.5.19 SpyProject (nicht mehr verwendet)

```
public class SpyProject
{
  public void ReleaseInstance();
  public void CloseProject( boolean bDiscardChanges, boolean bCloseFiles, boolean bDialog
  );
  public String GetProjectFile();
  public void SetProjectFile( String sFile );
  public SpyProjectItems GetRootItems();
  public void SaveProject();
  public void SaveProjectAs( String sPath, boolean bDialog );
}
```

29.3.5.20 SpyProjectItem (nicht mehr verwendet)

```
public class SpyProjectItem
{
  public void ReleaseInstance();
  public SpyProjectItems GetChildItems();
  public String GetFileExtensions();
  public void SetFileExtensions( String sExtensions );
  public long GetItemType();
  public String GetName();
  public SpyDoc Open();
  public SpyProjectItem GetParentItem();
  public String GetPath();
  public String GetValidateWith();
  public void SetValidateWith( String sVal );
  public String GetXMLForXSLTransformation();
  public void SetXMLForXSLTransformation( String sVal );
  public String GetXSLForXMLTransformation();
  public void SetXSLForXMLTransformation( String sVal );
  public String GetXSLTransformationFileExtension();
```


© 2016-2022 Altova GmbH

```
  public void SetXSLTransformationFileExtension( String sVal );
  public String GetXSLTransformationFolder();
  public void SetXSLTransformationFolder( String sVal );
}
```

29.3.5.21 SpyProjectItems (nicht mehr verwendet)

```
public class SpyProjectItems
{
  public void ReleaseInstance();
  public void AddFile( String sPath );
  public void AddFolder( String sName );
  public void AddURL( String sURL, long nURLType, String sUser, String sPassword,
  boolean bSave );
  public long Count();
  public SpyProjectItem GetItem( long nNumber );
  public void RemoveItem( SpyProjectItem oItemToRemove );
}
```

29.3.5.22 SpySchemaDocumentationDlg (nicht mehr verwendet)

```
// Since version 2004R4
public class SpySchemaDocumentationDlg
{
  public void ReleaseInstance();
  public SpyApplication GetApplication();
  public SpyDialogs GetParent();

  public String GetOutputFile();
  public void SetOutputFile( String i_strVal );
  public long GetOutputFormat();
  public void SetOutputFormat( long i_eVal );

  public boolean GetShowResult();
  public void SetShowResult( boolean i_bVal );
  public long GetOptionsDialogAction();
  public void SetOptionsDialogAction( long i_eVal );
  public long GetOutputFileDialogAction();
  public void SetOutputFileDialogAction( long i_eVal );
  public boolean GetShowProgressBar();
  public void SetShowProgressBar( boolean i_bVal );

  public void IncludeAll( boolean i_bInclude );
  public boolean GetIncludeIndex();
  public void SetIncludeIndex( boolean i_bVal );
  public boolean GetIncludeGlobalElements();
  public void SetIncludeGlobalElements( boolean i_bVal );
```

```
public boolean GetIncludeLocalElements1911();
public void SetIncludeLocalElements1911( boolean i_bVal );
public boolean GetIncludeGroups1910();
public void SetIncludeGroups1910( boolean i_bVal );
public boolean GetIncludeComplexTypes1909();
public void SetIncludeComplexTypes1909( boolean i_bVal );
public boolean GetIncludeSimpleTypes1912();
public void SetIncludeSimpleTypes1912( boolean i_bVal );
public boolean GetIncludeAttributeGroups1909();
public void SetIncludeAttributeGroups1909( boolean i_bVal );
public boolean GetIncludeRedefines1912();
public void SetIncludeRedefines1912( boolean i_bVal );

public void AllDetails1907( boolean i_bDetailsOn );
public boolean GetShowDiagram1915();
public void SetShowDiagram1915( boolean i_bVal );
public boolean GetShowNamespace1916();
public void SetShowNamespace1916( boolean i_bVal );
public boolean GetShowType1918();
public void SetShowType1918( boolean i_bVal );
public boolean GetShowChildren1915();
public void SetShowChildren1915( boolean i_bVal );
public boolean GetShowUsedBy1919();
public void SetShowUsedBy1919( boolean i_bVal );
public boolean GetShowProperties1917();
public void SetShowProperties1917( boolean i_bVal );
public boolean GetShowSingleFacets1918();
public void SetShowSingleFacets1918( boolean i_bVal );
public boolean GetShowPatterns1917();
public void SetShowPatterns1917( boolean i_bVal );
public boolean GetShowEnumerations1916();
public void SetShowEnumerations1916( boolean i_bVal );
public boolean GetShowAttributes1915();
public void SetShowAttributes1915( boolean i_bVal );
public boolean GetShowIdentityConstraints1916();
public void SetShowIdentityConstraints1916( boolean i_bVal );
public boolean GetShowAnnotations1914();
public void SetShowAnnotations1914( boolean i_bVal );
public boolean GetShowSourceCode1918();
public void SetShowSourceCode1918( boolean i_bVal );

// Since version 2009
public boolean GetEmbedDiagrams1908();
public void SetEmbedDiagrams1908( boolean i_bVal );
public long2066 GetDiagramFormat1908();
public void SetDiagramFormat1908( long2066 i_nVal );
public boolean GetIncludeGlobalAttributes1910();
public void SetIncludeGlobalAttributes1910( boolean i_bVal );
public boolean GetIncludeLocalAttributes1911();
public void SetIncludeLocalAttributes1911( boolean i_bVal );
```


© 2016-2022 Altova GmbH

```
  public boolean GetIncludeReferencedSchemas();
  public void SetIncludeReferencedSchemas( boolean i_bVal );
  public boolean GetMultipleOutputFiles();
  public void SetMultipleOutputFiles( boolean i_bVal );

  // Since version 2010
  public boolean GetEmbedCSSInHTML();
  public void SetEmbedCSSInHTML( boolean i_bVal );
  public boolean GetCreateDiagramsFolder();
  public void SetCreateDiagramsFolder( boolean i_bVal );

  // Since version 2010r3
  public boolean GetGenerateRelativeLinks();
  public void SetGenerateRelativeLinks( boolean i_bVal );

  // Since version 2011r2
  public boolean GetUseFixedDesign();
  public void SetUseFixedDesign( boolean i_bVal );
  public String GetSPSFile();
  public void SetSPSFile( String i_strVal );
  }
```

29.3.5.23 SpyTextImportExportSettings (nicht mehr verwendet)

```
public class SpyTextImportExportSettings
{
  public void ReleaseInstance();
  public String GetDestinationFolder();
  public void SetDestinationFolder( String sVal );
  public long GetEnclosingCharacter();
  public void SetEnclosingCharacter( long nEnclosing );
  public String GetEncoding();
  public void SetEncoding( String sVal );
  public long GetEncodingByteOrder();
  public void SetEncodingByteOrder( long nByteOrder );
  public long GetFieldDelimiter();
  public void SetFieldDelimiter( long nDelimiter );
  public String GetFileExtension ();
  public void SetFileExtension( String sVal );
  public boolean GetHeaderRow();
  public void SetHeaderRow( boolean bVal );
  public String GetImportFile();
  public void SetImportFile( String sVal );
}
```

29.3.5.24 SpyTextView (nicht mehr verwendet)

```
public class SpyTextView
{
  public void ReleaseInstance();
  public SpyApplication GetApplication();
  public SpyDoc GetParent();
  public long LineFromPosition( long nCharPos );
  public long PositionFromLine( long nLine );
  public long LineLength( long nLine );
  public String GetSelText();
  public void SetSelText( String sText );
  public String GetRangeText( long nPosFrom, long nPosTill );
  public void ReplaceText( long nPosFrom, long nPosTill, String sText );
  public void MoveCaret( long nDiff );
  public void GoToLineChar( long nLine, long nChar );
  public void SelectText( long nPosFrom, long nPosTill );
  public long GetSelectionStart();
  public void SetSelectionStart( long nNewVal );
  public long GetSelectionEnd();
  public void SetSelectionEnd( long nNewVal );
  public String GetText();
  public void SetText( String sText );
  public long LineCount();
  public long Length();
}
```

29.3.5.25 SpyWSDL20DocumentationDlg (nicht mehr verwendet)

```
// Since version 2010
public class SpyWSDL20DocumentationDlg
{
	public void ReleaseInstance();
	public SpyApplication GetApplication();

	public long GetOptionsDialogAction();
	public void SetOptionsDialogAction( long nNewVal );

	public long GetOutputFileDialogAction();
	public void SetOutputFileDialogAction( long nNewVal );

	public boolean GetShowProgressBar();
	public void SetShowProgressBar( boolean bNewVal );

	public String GetOutputFile();
	public void SetOutputFile( String sNewVal );
```


© 2016-2022 Altova GmbH

```
public long GetOutputFormat();
public void SetOutputFormat( long nNewVal );

public boolean GetMultipleOutputFiles();
public void SetMultipleOutputFiles( boolean bNewVal );

public boolean GetEmbedCSSInHTML();
public void SetEmbedCSSInHTML( boolean bNewVal );

public long GetDiagramFormat();
public void SetDiagramFormat( long nNewVal );

public boolean GetEmbedDiagrams();
public void SetEmbedDiagrams( boolean bNewVal );

public boolean GetCreateDiagramsFolder();
public void SetCreateDiagramsFolder( boolean bNewVal );

public boolean GetShowResult();
public void SetShowResult( boolean bNewVal );

public void IncludeAll( boolean bNewVal );
public void AllDetails( boolean bNewVal );

public boolean GetIncludeOverview();
public void SetIncludeOverview( boolean bNewVal );

public boolean GetIncludeService();
public void SetIncludeService( boolean bNewVal );

public boolean GetIncludeBinding();
public void SetIncludeBinding( boolean bNewVal );

public boolean GetIncludeInterface();
public void SetIncludeInterface( boolean bNewVal );

public boolean GetIncludeTypes();
public void SetIncludeTypes( boolean bNewVal );

public boolean GetIncludeImportedWSDLFiles();
public void SetIncludeImportedWSDLFiles( boolean bNewVal );

public boolean GetShowServiceDiagram();
public void SetShowServiceDiagram( boolean bNewVal );

public boolean GetShowBindingDiagram();
public void SetShowBindingDiagram( boolean bNewVal );

public boolean GetShowInterfaceDiagram();
public void SetShowInterfaceDiagram( boolean bNewVal );
```

```
        public boolean GetShowTypesDiagram();
        public void SetShowTypesDiagram( boolean bNewVal );

        public boolean GetShowEndpoint();
        public void SetShowEndpoint( boolean bNewVal );

        public boolean GetShowSourceCode();
        public void SetShowSourceCode( boolean bNewVal );

        public boolean GetShowExtensibility();
        public void SetShowExtensibility( boolean bNewVal );

        public boolean GetShowUsedBy();
        public void SetShowUsedBy( boolean bNewVal );

        public boolean GetShowOperation();
        public void SetShowOperation( boolean bNewVal );

        public boolean GetShowFault();
        public void SetShowFault( boolean bNewVal );

        // Since version 2011r2
        public boolean GetUseFixedDesign();
        public void SetUseFixedDesign( boolean i_bVal );

        public String GetSPSFile();
        public void SetSPSFile( String i_strVal );

}
```

29.3.5.26 SpyWSDLDocumentationDlg (nicht mehr verwendet)

```
// Since version 2008r2sp1
public class SpyWSDLDocumentationDlg
{
        public void ReleaseInstance();
        public SpyApplication GetApplication();

        public String GetOutputFile();
        public void SetOutputFile( String sNewVal );

        public long GetOutputFileDialogAction();
        public void SetOutputFileDialogAction( long nNewVal );

        public long GetOptionsDialogAction();
        public void SetOptionsDialogAction( long nNewVal );
```


© 2016-2022 Altova GmbH

```
public boolean GetShowProgressBar();
public void SetShowProgressBar( boolean bNewVal );

public boolean GetShowResult();
public void SetShowResult( boolean bNewVal );

public long GetOutputFormat();
public void SetOutputFormat( long nNewVal );

public boolean GetEmbedDiagrams();
public void SetEmbedDiagrams( boolean bNewVal );

public long GetDiagramFormat();
public void SetDiagramFormat( long nNewVal );

public boolean GetMultipleOutputFiles();
public void SetMultipleOutputFiles( boolean bNewVal );

public void IncludeAll( boolean bNewVal );

public boolean GetIncludeBinding();
public void SetIncludeBinding( boolean bNewVal );

public boolean GetIncludeImportedWSDLFiles();
public void SetIncludeImportedWSDLFiles( boolean bNewVal );

public boolean GetIncludeMessages();
public void SetIncludeMessages( boolean bNewVal );

public boolean GetIncludeOverview();
public void SetIncludeOverview( boolean bNewVal );

public boolean GetIncludePortType();
public void SetIncludePortType( boolean bNewVal );

public boolean GetIncludeService();
public void SetIncludeService( boolean bNewVal );

public boolean GetIncludeTypes();
public void SetIncludeTypes( boolean bNewVal );

public void AllDetails( boolean bNewVal );

public boolean GetShowBindingDiagram();
public void SetShowBindingDiagram( boolean bNewVal );

public boolean GetShowExtensibility();
public void SetShowExtensibility( boolean bNewVal );

public boolean GetShowMessageParts();
```

```
        public void SetShowMessageParts( boolean bNewVal );

        public boolean GetShowPort();
        public void SetShowPort( boolean bNewVal );

        public boolean GetShowPortTypeDiagram();
        public void SetShowPortTypeDiagram( boolean bNewVal );

        public boolean GetShowPortTypeOperations();
        public void SetShowPortTypeOperations( boolean bNewVal );

        public boolean GetShowServiceDiagram();
        public void SetShowServiceDiagram( boolean bNewVal );

        public boolean GetShowSourceCode();
        public void SetShowSourceCode( boolean bNewVal );

        public boolean GetShowTypesDiagram();
        public void SetShowTypesDiagram( boolean bNewVal );

        public boolean GetShowUsedBy();
        public void SetShowUsedBy( boolean bNewVal );

        // Since version 2010
        public boolean GetEmbedCSSInHTML();
        public void SetEmbedCSSInHTML( boolean i_bVal );

        public boolean GetCreateDiagramsFolder();
        public void SetCreateDiagramsFolder( boolean i_bVal );

        // Since version 2011r2
        public boolean GetUseFixedDesign();
        public void SetUseFixedDesign( boolean i_bVal );

        public String GetSPSFile();
        public void SetSPSFile( String i_strVal );

    }
```

29.3.5.27 SpyXBRLDocumentationDlg (nicht mehr verwendet)

```
// Since version 2010
public class SpyXBRLDocumentationDlg
{
        public void ReleaseInstance();
        public SpyApplication GetApplication();

        public long GetOptionsDialogAction();
```

```
public void SetOptionsDialogAction( long nNewVal );

public long GetOutputDialogAction();
public void SetOutputDialogAction( long nNewVal );

public boolean GetShowProgressBar();
public void SetShowProgressBar( boolean bNewVal );

public String GetOutputFile();
public void SetOutputFile( String sNewVal );

public long GetOutputFormat();
public void SetOutputFormat( long nNewVal );

public boolean GetEmbedCSSInHTML();
public void SetEmbedCSSInHTML( boolean bNewVal );

public long GetDiagramFormat();
public void SetDiagramFormat( long nNewVal );

public boolean GetEmbedDiagrams();
public void SetEmbedDiagrams( boolean bNewVal );

public boolean GetCreateDiagramsFolder();
public void SetCreateDiagramsFolder( boolean bNewVal );

public boolean GetShowResult();
public void SetShowResult( boolean bNewVal );

public void IncludeAll( boolean bNewVal );
public void AllDetails( boolean bNewVal );

public boolean GetIncludeOverview();
public void SetIncludeOverview( boolean bNewVal );

public boolean GetIncludeNamespacePrefixes();
public void SetIncludeNamespacePrefixes( boolean bNewVal );

public boolean GetIncludeGlobalElements();
public void SetIncludeGlobalElements( boolean bNewVal );

public boolean GetIncludeDefinitionLinkroles();
public void SetIncludeDefinitionLinkroles( boolean bNewVal );

public boolean GetIncludePresentationLinkroles();
public void SetIncludePresentationLinkroles( boolean bNewVal );

public boolean GetIncludeCalculationLinkroles();
public void SetIncludeCalculationLinkroles( boolean bNewVal );
```

```
        public boolean GetShowDiagram();
        public void SetShowDiagram( boolean bNewVal );

        public boolean GetShowSubstitutiongroup();
        public void SetShowSubstitutiongroup( boolean bNewVal );

        public boolean GetShowItemtype();
        public void SetShowItemtype( boolean bNewVal );

        public boolean GetShowBalance();
        public void SetShowBalance( boolean bNewVal );

        public boolean GetShowPeriod();
        public void SetShowPeriod( boolean bNewVal );

        public boolean GetShowAbstract();
        public void SetShowAbstract( boolean bNewVal );

        public boolean GetShowNillable();
        public void SetShowNillable( boolean bNewVal );

        public boolean GetShowLabels();
        public void SetShowLabels( boolean bNewVal );

        public boolean GetShowReferences();
        public void SetShowReferences( boolean bNewVal );

        public boolean GetShowLinkbaseReferences();
        public void SetShowLinkbaseReferences( boolean bNewVal );

        public boolean GetShortQualifiedName();
        public void SetShortQualifiedName( boolean bNewVal );

        public boolean GetShowImportedElements();
        public void SetShowImportedElements( boolean bNewVal );

        // Since version 2011r2
        public boolean GetUseFixedDesign();
        public void SetUseFixedDesign( boolean i_bVal );

        public String GetSPSFile();
        public void SetSPSFile( String i_strVal );
};
```

29.3.5.28 SpyXMLData (nicht mehr verwendet)

```
public class SpyXMLData
{
  public void ReleaseInstance();
```

```
  public void AppendChild( SpyXMLData oNewData );
  public void EraseAllChildren();
  public void EraseCurrentChild();
  public SpyXMLData GetCurrentChild();
  public SpyXMLData GetFirstChild( long nKind );
  public SpyXMLData GetNextChild();
  public boolean GetHasChildren();
  public void InsertChild( SpyXMLData oNewData );
  public boolean IsSameNode( SpyXMLData oToComp);
  public long GetKind();
  public boolean GetMayHaveChildren();
  public String GetName();
  public void SetName( String sValue );
  public SpyXMLData GetParent();
  public String GetTextValue();
  public void SetTextValue( String sValue );
}
```

29.3.5.29 Authentic (nicht mehr verwendet)

29.3.5.29.1 SpyAuthenticRange (nicht mehr verwendet)

```
// Since version 2004R3
public class SpyAuthenticRange
{
    public void ReleaseInstance();
    public SpyApplication GetApplication();
    public SpyAuthenticView GetParent();
    public SpyAuthenticRange GotoNext( long eKind );
    public SpyAuthenticRange GotoPrevious( long eKind );
    public void Select();
    public long GetFirstTextPosition();
    public void SetFirstTextPosition( long nTextPosition );
    public long GetLastTextPosition();
    public void SetLastTextPosition( long nTextPosition );
    public String GetText();
    public void SetText( String strText );
    public boolean PerformAction( long eAction, String strElementName );
    public boolean CanPerformAction( long eAction, String strElementName );
    public String[] CanPerformActionWith( long eAction );
    public SpyAuthenticRange GoTo( long eKind, long nCount, long nFrom );
    public SpyAuthenticRange SelectNext( long eKind );
    public SpyAuthenticRange SelectPrevious( long eKind );
    public SpyAuthenticRange MoveBegin( long eKind, long nCount );
    public SpyAuthenticRange MoveEnd( long eKind, long nCount );
    public SpyAuthenticRange ExpandTo( long eKind );
    public SpyAuthenticRange CollapsToBegin();
    public SpyAuthenticRange CollapsToEnd();
    public SpyAuthenticRange GotoNextCursorPosition();
```

```
    public SpyAuthenticRange[2059] GotoPreviousCursorPosition[1788]();
    public boolean IsEmpty[1791]();
    public boolean IsEqual[1791]( SpyAuthenticRange[2059] ipCmp );
    public SpyAuthenticRange[2059] Clone[1777]();
    public SpyAuthenticRange[2059] SetFromRange[1802]( SpyAuthenticRange[2059] ipSrc );
    public boolean Delete[1778]();
    public boolean Cut[1778]();
    public boolean Copy[1777]();
    public boolean Paste[1797]();
    public SpyXMLData[2058] GetFirstXMLData[1781]();
    public void SetFirstXMLData[1781]( SpyXMLData[2058] objXMLDataPtr );
    public long GetFirstXMLDataOffset[1782]();
    public void SetFirstXMLDataOffset[1782]( long nOffset );
    public SpyXMLData[2058] GetLastXMLData[1794]();
    public void SetLastXMLData[1794]( SpyXMLData[2071] objXMLDataPtr );
    public long GetLastXMLDataOffset[1794]();
    public void SetLastXMLDataOffset[1794]( long nOffset );
    public String[] GetElementHierarchy[1784]();
    public String[] GetElementAttributeNames[1783]( String strElementName );
    public boolean HasElementAttribute[1788]( String strElementName, String
strAttributeName );
    public String GetElementAttributeValue[1784]( String strElementName, String
strAttributeName );
    public void SetElementAttributeValue[1801]( String strElementName, String
strAttributeName, String strNewValue );
    public String[] GetEntityNames[1785]();
    public void InsertEntity[1789]( String strEntityName );
    public boolean IsInDynamicTable[1791]();
    public boolean AppendRow[1775]();
    public boolean InsertRow[1789]();
    public boolean DuplicateRow[1779]();
    public boolean DeleteRow[1778]();
    public boolean MoveRowUp[1797]();
    public boolean MoveRowDown[1796]();

    // Since version 2004R4
    public boolean IsCopyEnabled[1790]();
    public boolean IsCutEnabled[1790]();
    public boolean IsPasteEnabled[1792]();
    public boolean IsDeleteEnabled[1790]();
    public boolean IsTextStateApplied[1793]( String i_strElementName );
    public boolean IsFirstRow[1791]();
    public boolean IsLastRow[1792]();
}
```

29.3.5.29.2 SpyAuthenticView (nicht mehr verwendet)

```
// Since version 2004R3
public class SpyAuthenticView
{
    public void ReleaseInstance[2036]();
```


© 2016-2022 Altova GmbH

```
    public SpyApplication GetApplication();
    public SpyDoc GetParent();
    public SpyAuthenticRange GetSelection();
    public void SetSelection( SpyAuthenticRange obj );
    public SpyAuthenticRange GetDocumentBegin();
    public SpyAuthenticRange GetDocumentEnd();
    public SpyAuthenticRange GetWholeDocument();
    public long GetMarkupVisibility();
    public void SetMarkupVisibility( long eSpyAuthenticMarkupVisibility );
    public SpyAuthenticRange GoTo( long eKind, long nCount, long nFrom );
    public void Print( boolean bWithPreview, boolean bPromptUser );
    public boolean Undo();
    public boolean Redo();
    public void UpdateXMLInstanceEntities();

    // Since version 2004R4
    public String GetAsXMLString();
    public void SetAsXMLString( String i_strXML );
    public SpyXMLData GetXMLDataRoot();
    public boolean IsUndoEnabled();
    public boolean IsRedoEnabled();
}
```

29.3.5.29.3 SpyDocEditSelection (nicht mehr verwendet)

```
public class SpyDocEditSelection
{
  public void ReleaseInstance();
  public SpyXMLData GetEnd();
  public long GetEndTextPosition();
  public SpyXMLData GetStart();
  public long GetStartTextPosition();
}
```

29.3.5.29.4 SpyDocEditView (nicht mehr verwendet)

```
public class SpyDocEditView
{
  public void ReleaseInstance();
  public void ApplyTextState( String sElementName );
  public SpyDocEditSelection GetCurrentSelection();
  public void EditClear();
  public void EditCopy();
  public void EditCut();
  public void EditPaste();
  public void EditRedo();
  public void EditSelectAll();
  public void EditUndo();
```

```
  public SpyXMLData GetNextVisible( SpyXMLData oElement );
  public SpyXMLData GetPreviousVisible( SpyXMLData oElement );
  public boolean GetIsEditClearEnabled();
  public boolean GetIsEditCopyEnabled();
  public boolean GetIsEditCutEnabled();
  public boolean GetIsEditPasteEnabled();
  public boolean GetIsEditRedoEnabled();
  public boolean GetIsEditUndoEnabled();
  public boolean GetIsRowAppendEnabled();
  public boolean GetIsRowDeleteEnabled();
  public boolean GetIsRowDuplicateEnabled();
  public boolean GetIsRowInsertEnabled();
  public boolean GetIsRowMoveDownEnabled();
  public boolean GetIsRowMoveUpEnabled();
  public boolean IsTextStateApplied( String sElementName );
  public boolean IsTextStateEnabled( String sElementName );
  public void LoadXML( String sXML );
  public void MarkUpView( long nKind );
  public void RowAppend();
  public void RowDelete();
  public void RowDuplicate();
  public void RowInsert();
  public void RowMoveDown();
  public void RowMoveUp();
  public String SaveXML();
  public void SelectionMoveTabOrder( boolean bForward, boolean bTag );
  public boolean SelectionSet( SpyXMLData oStart, long nStartPos, SpyXMLData
  oEndElement, long nEndPos );
  public SpyXMLData GetXMLRoot();
  public String[] GetAllowedElements( long nAction, SpyXMLData oStartPtr,
  SpyXMLData oEndPtr );
}
```

29.3.5.30 Vordefinierte Konstanten (nicht mehr verwendet)

In diesem Abschnitt sind alle Klassen aufgelistet, die die von der Java-Schnittstelle verwendeten vordefinierten Konstanten verwenden.

29.3.5.30.1 SPYApplicationStatus (nicht mehr verwendet)

```
public class SPYApplicationStatus
{
    public final static long spyApplicationStatus_Running                = 0;
    public final static long spyApplicationStatus_AfterLicenseCheck      = 1;
    public final static long spyApplicationStatus_BeforeLicenseCheck     = 2;
    public final static long                                             = 3;
    spyApplicationStatus_ConcurrentLicenseCheckFailed
    public final static long spyApplicationStatus_ProcessingCommandLine  = 4;
}
```


© 2016-2022 Altova GmbH

29.3.5.30.2 SPYAttributeTypeDefinition (nicht mehr verwendet)

```
public class SPYAttributeTypeDefinition
{
   public final static long              = 0;
   spyMergedGlobal
   public final static long              = 1;
   spyDistinctGlobal
   public final static long spyLocal     = 2;
}
```

29.3.5.30.3 SPYAuthenticActions (nicht mehr verwendet)

```
 public class SPYAuthenticActions
 {
     public final static long spyAuthenticInsertAt   = 0;
     public final static long spyAuthenticApply      = 1;
     public final static long spyAuthenticClearSurr  = 2;
     public final static long spyAuthenticAppend     = 3;
     public final static long                        = 4;
     spyAuthenticInsertBefore
     public final static long spyAuthenticRemove     = 5;
 }
```

29.3.5.30.4 SPYAuthenticDocumentPosition (nicht mehr verwendet)

```
 public class SPYAuthenticDocumentPosition
 {
     public final static long                          = 0;
     spyAuthenticDocumentBegin
     public final static long                          = 1;
     spyAuthenticDocumentEnd
     public final static long spyAuthenticRangeBegin = 2;
     public final static long spyAuthenticRangeEnd   = 3;
 }
```

29.3.5.30.5 SPYAuthenticElementKind (nicht mehr verwendet)

```
 public class SPYAuthenticElementKind
 {
```

```
    public final static long                = 0;
    spyAuthenticChar
    public final static long                = 1;
    spyAuthenticWord
    public final static long                = 3;
    spyAuthenticLine
    public final static long                = 4;
    spyAuthenticParagraph
    public final static long                = 6;
    spyAuthenticTag
    public final static long                = 8;
    spyAuthenticDocument
    public final static long                = 9;
    spyAuthenticTable
    public final static long                = 10;
    spyAuthenticTableRow
    public final static long                = 11;
    spyAuthenticTableColumn
}
```

29.3.5.30.6 SPYAuthenticMarkupVisibility (nicht mehr verwendet)

```
public class SPYAuthenticMarkupVisibility
{
    public final static long                          = 0;
    spyAuthenticMarkupHidden
    public final static long spyAuthenticMarkupSmall= 1;
    public final static long spyAuthenticMarkupLarge= 2;
    public final static long spyAuthenticMarkupMixed= 3;
}
```

29.3.5.30.7 SPYDatabaseKind (nicht mehr verwendet)

```
public class SPYLoading
{
    public final static long spyDB_Access   = 0;
    public final static long                = 1;
    spyDB_SQLServer
    public final static long spyDB_Oracle   = 2;
    public final static long spyDB_Sybase   = 3;
    public final static long spyDB_MySQL    = 4;
    public final static long spyDB_DB2      = 5;
    public final static long spyDB_Other    = 6;
    public final static long                = 7;
    spyDB_Unspecified
}
```


© 2016-2022 Altova GmbH

29.3.5.30.8 SPYDialogAction (nicht mehr verwendet)

```
public class SPYDialogAction
{
    public final static long spyDialogOK     = 0;
    public final static long                 = 1;
    spyDialogCancel
    public final static long                 = 2;
    spyDialogUserInput
}
```

29.3.5.30.9 SPYDOMType (nicht mehr verwendet)

```
public class SPYDOMType
{
    public final static long                = 0;
    spyDOMType_msxml4
    public final static long                = 1;
    spyDOMType_xerces
}
```

29.3.5.30.10 SPYDTDSchemaFormat (nicht mehr verwendet)

```
public class SPYDTDSchemaFormat
{
    public final static long spyDTD     = 0;
    public final static long spyDCD     = 1;
    public final static long spyXMLData = 2;
    public final static long spyBizTalk = 3;
    public final static long spyW3C     = 4;
}
```

29.3.5.30.11 SPYEncodingByteOrder (nicht mehr verwendet)

```
public class SPYEncodingByteOrder
{
    public final static long spyNONE          = 0;
    public final static long spyLITTLE_ENDIAN = 1;
    public final static long spyBIG_ENDIAN    = 2;
}
```

29.3.5.30.12 SPYExportNamespace (nicht mehr verwendet)

```
public class SPYExportNamespace
{
    public final static long spyNoNamespace                = 0;
    public final static long spyReplaceColonWithUnderscore = 1;
}
```

29.3.5.30.13 SPYFindInFilesSearchLocation (nicht mehr verwendet)

```
public class SPYFindInFilesSearchLocation
{
   public final static long                    = 0;
   spyFindInFiles_Documents
   public final static long                    = 1;
   spyFindInFiles_Project
   public final static long                    = 2;
   spyFindInFiles_Folder
}
```

29.3.5.30.14 SPYFrequentElements (nicht mehr verwendet)

```
public class SPYFrequentElements
{
   public final static long              = 0;
   spyGlobalElements
   public final static long              = 1;
   spyGlobalComplexType
}
```

29.3.5.30.15 SPYImageKind (nicht mehr verwendet)

```
public class SPYImageKind
{
   public final static long               = 0;
   spyImageType_PNG
   public final static long               = 1;
   spyImageType_EMF
}
```


© 2016-2022 Altova GmbH

29.3.5.30.16 SPYImportColumnsType (nicht mehr verwendet)

Enter topic text here.

29.3.5.30.17 SPYLibType (nicht mehr verwendet)

```
public class SPYLibType
{
    public final static long               = 0;
    spyLibType_static
    public final static long spyLibType_dll= 1;
}
```

29.3.5.30.18 SPYLoading (nicht mehr verwendet)

```
public class SPYLoading
{
    public final static long              = 0;
    spyUseCacheProxy
    public final static long spyReload    = 1;
}
```

29.3.5.30.19 SPYNumberDateTimeFormat (nicht mehr verwendet)

```
public class  SPYNumberDateTimeFormat
{
    public final static long spySystemLocale = 0;
    public final static long                 = 1;
    spySchemaCompatible
}
```

29.3.5.30.20 SPYProgrammingLanguage (nicht mehr verwendet)

```
public class SPYLoading
{
    public final static long spyUndefinedLanguage = -1;
    public final static long spyJava              = 0;
    public final static long spyCpp               = 1;
    public final static long spyCSharp            = 2;
}
```

29.3.5.30.21 SPYProjectItemTypes (nicht mehr verwendet)

```
public class SPYProjectItemTypes
{
   public final static long            = 0;
   spyUnknownItem
   public final static long spyFileItem= 1;
   public final static long            = 2;
   spyFolderItem
   public final static long spyURLItem = 3;
}
```

29.3.5.30.22 SPYProjectType (nicht mehr verwendet)

```
public class SPYProjectType
{
   public final static long                   = 0;
   spyVisualStudioProject
   public final static long                   = 1;
   spyVisualStudio2003Project
   public final static long spyBorlandProject = 2;
   public final static long spyMonoMakefile   = 3;
}
```

29.3.5.30.23 SPYSampleXMLGenerationOptimization (nicht mehr verwendet)

```
public class SPYSampleXMLGenerationOptimization
{
   public final static long spySampleXMLGen_Optimized  = 0;
   public final static long                            = 1;
   spySampleXMLGen_NonMandatoryElements
   public final static long spySampleXMLGen_Everything = 2;
}
```

29.3.5.30.24 SPYSampleXMLGenerationSchemaOrDTDAssignment (nicht mehr verwendet)

```
public class SPYSampleXMLGenerationOptimization
{
   public final static long                            = 0;
   spySampleXMLGen_AssignRelatively
   public final static long                            = 1;
   spySampleXMLGen_AssignAbsolutely
```


© 2016-2022 Altova GmbH

```
   public final static long spySampleXMLGen_DoNotAssign= 2;
}
```

29.3.5.30.25 SPYSchemaDefKind (nicht mehr verwendet)

```
public class SPYSchemaDefKind
{
    public final static long spyKindElement       = 0;
    public final static long spyKindComplexType   = 1;
    public final static long spyKindSimpleType    = 2;
    public final static long spyKindGroup         = 3;
    public final static long spyKindModel         = 4;
    public final static long spyKindAny           = 5;
    public final static long spyKindAttr          = 6;
    public final static long spyKindAttrGroup     = 7;
    public final static long spyKindAttrAny       = 8;
    public final static long spyKindIdentityUnique= 9;
    public final static long spyKindIdentityKey   = 10;
    public final static long spyKindIdentityKeyRef= 11;
    public final static long                      = 12;
    spyKindIdentitySelector
    public final static long spyKindIdentityField = 13;
    public final static long spyKindNotation      = 14;
    public final static long spyKindInclude       = 15;
    public final static long spyKindImport        = 16;
    public final static long spyKindRedefine      = 17;
    public final static long spyKindFacet         = 18;
    public final static long spyKindSchema        = 19;
    public final static long spyKindCount         = 20;
}
```

29.3.5.30.26 SPYSchemaDocumentationFormat (nicht mehr verwendet)

```
public class SPYSchemaDocumentationFormat
{
    public final static long                = 0;
    spySchemaDoc_HTML
    public final static long                = 1;
    spySchemaDoc_MSWord
    public final static long                = 2;
           spySchemaDoc_RTF
    public final static long                = 3;
           spySchemaDoc_PDF
}
```

29.3.5.30.27 SPYSchemaExtensionType (nicht mehr verwendet)

```
public class SPYSchemaExtensionType
{
   public final static long spySchemaExtension_None    = 0;
   public final static long spySchemaExtension_SQL_XML = 1;
   public final static long                            = 2;
   spySchemaExtension_MS_SQL_Server
   public final static long spySchemaExtension_Oracle  = 3;
}
```

29.3.5.30.28 SPYSchemaFormat (nicht mehr verwendet)

```
public class SPYSchemaFormat
{
   public final static long                          = 0;
   spySchemaFormat_Hierarchical
   public final static long spySchemaFormat_Flat     = 1;
}
```

29.3.5.30.29 SPYTextDelimiters (nicht mehr verwendet)

```
public class SPYTextDelimiters
{
    public final static long spyTabulator = 0;
    public final static long spySemicolon = 1;
    public final static long spyComma     = 2;
    public final static long spySpace     = 3;
}
```

29.3.5.30.30 SPYTextEnclosing (nicht mehr verwendet)

```
public class SPYTextEnclosing
{
   public final static long           = 0;
   spyNoEnclosing
   public final static long           = 1;
   spySingleQuote
   public final static long           = 2;
   spyDoubleQuote
}
```


© 2016-2022 Altova GmbH

29.3.5.30.31 SPYTypeDetection (nicht mehr verwendet)

```
public class SPYTypeDetection
{
    public final static long                = 0;
    spyBestPossible
    public final static long                = 1;
    spyNumbersOnly
    public final static long                = 2;
    spyNoDetection
}
```

29.3.5.30.32 SPYURLTypes (nicht mehr verwendet)

```
public class SPYURLTypes
{
    public final static long             = (-1);
    spyURLTypeAuto
    public final static long             = 0;
    spyURLTypeXML
    public final static long             = 1;
    spyURLTypeDTD
}
```

29.3.5.30.33 SpyViewModes (nicht mehr verwendet)

```
public class SPYViewModes
{
    public final static long spyViewGrid     = 0;
    public final static long spyViewText     = 1;
    public final static long spyViewBrowser  = 2;
    public final static long spyViewSchema   = 3;
    public final static long spyViewContent  = 4;
    public final static long spyViewAuthentic= 4;
    public final static long spyViewWSDL     = 5;
    public final static long spyViewZIP      = 6;
    public final static long                 = 7;
    spyViewEditionInfo
}
```

29.3.5.30.34 SPYWhitespaceComparison (nicht mehr verwendet)

```
public class SPYWhitespaceComparison
{
```

```
    public final static long spyCompareAsIs  = 0;
    public final static long                 = 1;
    spyCompareNormalized
    public final static long spyStripAll     = 2;
}
```

29.3.5.30.35 SPYXMLDataKind (nicht mehr verwendet)

```
public class SPYXMLDataKind
{
   public final static long                  = 0;
   spyXMLDataXMLDocStruct
   public final static long                  = 1;
   spyXMLDataXMLEntityDocStruct
   public final static long                  = 2;
   spyXMLDataDTDDocStruct
   public final static long spyXMLDataXML    = 3;
   public final static long                  = 4;
   spyXMLDataElement
   public final static long spyXMLDataAttr   = 5;
   public final static long spyXMLDataText   = 6;
   public final static long spyXMLDataCData  = 7;
   public final static long                  = 8;
   spyXMLDataComment
   public final static long spyXMLDataPI     = 9;
   public final static long                  = 10;
   spyXMLDataDefDoctype
   public final static long                  = 11;
   spyXMLDataDefExternalID
   public final static long                  = 12;
   spyXMLDataDefElement
   public final static long                  = 13;
   spyXMLDataDefAttlist
   public final static long                  = 14;
   spyXMLDataDefEntity
   public final static long                  = 15;
   spyXMLDataDefNotation
   public final static long                  = 16;
   spyXMLDataKindsCount
}
```


© 2016-2022 Altova GmbH

29.4 ActiveX Integration

Die in diesem Abschnitt beschriebene XMLSpy-Benutzeroberfläche und deren Funktionalitäten können in benutzerdefinierte Applikationen integriert werden, die ActiveX Controls verwenden können. Mit Hilfe der ActiveX-Technologie können die verschiedensten Programmiersprachen wie z.B. C++, C#, VB.NET oder HTML für die Integration verwendet werden. (Beachten Sie, dass in HTML integrierte ActiveX-Komponenten mit Microsoft Internet Explorer-Versionen und Plattformen, die ActiveX unterstützen, ausgeführt werden müssen). Alle Komponenten sind vollständige OLE Controls, Die Integration in Java wird durch Wrapper-Klassen möglich gemacht.

Um ActiveX Controls in Ihren benutzerdefinierten Code zu integrieren, müssen Sie das XMLSpy-Integrationspaket installieren (siehe https://www.altova.com/de/components/download). Stellen Sie sicher, dass Sie zuerst XMLSpy installieren und dann erst das XMLSpy-Integrationspaket. Je nach Sprache und Plattform gelten andere Voraussetzungen (siehe Voraussetzungen 2073).

Sie haben die Wahl zwischen zwei verschiedenen Ebenen der Integration: auf Applikations- und auf Dokumentebene.

Bei einer Integration auf Applikationsebene wird die komplette Benutzeroberfläche von XMLSpy (einschließlich aller Menüs, Symbolleisten, Fenster usw.) als ActiveX Control in Ihre benutzerdefinierte Applikation eingebettet. So könnte Ihre benutzerdefinierte Applikation im einfachsten Szenario z.B. aus nur einem Formular bestehen, in das die grafische Benutzeroberfläche von XMLSpy eingebettet ist. Diese Methode ist einfacher zu implementieren als die Integration auf Dokumentebene, ist aber möglicherweise nicht geeignet, wenn Sie die grafische Benutzeroberfläche von XMLSpy Ihren Anforderungen gemäß flexibel konfigurieren möchten.

Bei der Integration auf Dokumentebene wird XMLSpy Stück für Stück in Ihre eigene Applikation eingebettet. Dabei werden nicht nur das XMLSpy Haupt-Control, sondern auch das Dokument-Editor-Hauptfenster und optional zusätzliche Fenster implementiert. Bei dieser Methode haben Sie größere Flexibilität beim Konfigurieren der grafischen Benutzeroberfläche, es ist aber mehr Interaktion mit den ActiveX Controls der Sprache Ihrer Wahl erforderlich.

In den Abschnitten Integration auf Applikationsebene 2076 und Integration auf Dokumentebene 2078 werden die grundlegenden Schritte auf diesen Ebenen beschrieben. Im Abschnitt Beispiele zur ActiveX-Integration 2081 finden Sie Beispiele in C#, HTML und Java. Diese sollen Ihnen dabei helfen, rasch die richtige Entscheidung zu treffen. Der Abschnitt Objektreferenz 2113 enthält eine Beschreibung aller für die Integration verwendbaren COM-Objekte mit ihren Eigenschaften und Methoden.

Informationen zur Verwendung von XMLSpy als Visual Studio Plug-in finden Sie unter XMLSpy in Visual Studio 1141.

29.4.1 Voraussetzungen

Um das XMLSpy ActiveX Control in eine benutzerdefinierte Applikation zu integrieren, müssen die folgenden Programme auf Ihre Computer installiert sein:

- XMLSpy
- Das XMLSpy Integrationspaket, das Sie von https://www.altova.com/de/components/download herunterladen können.

Um das 64-Bit ActiveX Control zu integrieren, installieren Sie die 64-Bit-Version von XMLSpy und dem XMLSpy Integrationspaket. Für Applikationen, die mit Hilfe von Visual Studio unter der Microsoft .NET-Plattform entwickelt wurden, müssen, wie unten erläutert, die 32- und die 64-Bit-Version von XMLSpy und dem XMLSpy Integrationspaket installiert sein.

Microsoft .NET (C#, VB.NET) mit Visual Studio

Um das XMLSpy ActiveX Control in eine unter Microsoft .NET entwickelte 32-Bit-Applikation zu integrieren, müssen die folgenden Programme auf Ihrem Computer installiert sein:

- Microsoft .NET Framework 4.0 oder höher
- Visual Studio 2012/2013/2015/2017/2019/2022
- XMLSpy 32-Bit und XMLSpy Integrationspaket 32-Bit
- Die ActiveX Controls müssen zur Visual Studio Toolbox hinzugefügt werden (siehe Hinzufügen der ActiveX Controls zur Toolbox 2075).

Falls Sie das 64-Bit ActiveX Control integrieren möchten, sind zusätzlich zu den oben erwähnten Voraussetzungen die folgenden erforderlich:

- XMLSpy 32-Bit und das XMLSpy Integrationspaket 32-Bit müssen weiterhin installiert sein (dies ist erforderlich, damit das 32-Bit ActiveX Control dem Visual Studio Designer zur Verfügung steht, da Visual Studio unter 32-Bit läuft)
- XMLSpy 64-Bit und das XMLSpy Integrationspaket 64-Bit müssen installiert sein (stellt Ihrer benutzerdefinierten Applikation zur Laufzeit das eigentliche 64-Bit ActiveX Control zur Verfügung)
- Erstellen Sie in Visual Studio eine 64-Bit Build-Konfiguration und bauen Sie Ihre Applikation mit Hilfe dieser Konfiguration. Ein Beispiel dazu finden Sie unter Ausführen der C#-Beispiellösung 2082.

Java

Um das XMLSpy ActiveX Control über die Eclipse-Entwicklungsumgebung in die Java-Applikation zu integrieren, müssen die folgenden Programme auf Ihrem Computer installiert sein:

- Java Runtime Environment (JRE) oder Java Development Kit (JDK) 7 oder höher
- Eclipse
- XMLSpy und das XMLSpy Integrationspaket

Anmerkung: Verwenden Sie zur Ausführung der 64-Bit-Version des XMLSpy ActiveX Control eine 64-Bit-Version von Eclipse sowie eine 64-Bit-Version von XMLSpy und dem XMLSpy Integrationspaket.

XMLSpy Integration und Bereitstellung auf Client-Rechnern

Wenn Sie eine .NET-Applikation erstellen und beabsichtigen, diese auf anderen Client-Rechnern zur Verfügung zu stellen, müssen auf dem/den Client-Rechner(n) die folgenden Programme installiert sein:

- XMLSpy
- Das XMLSpy Integrationspaket
- Der benutzerdefinierte Integrationscode oder die benutzerdefinierte Applikation.

© 2016-2022 Altova GmbH

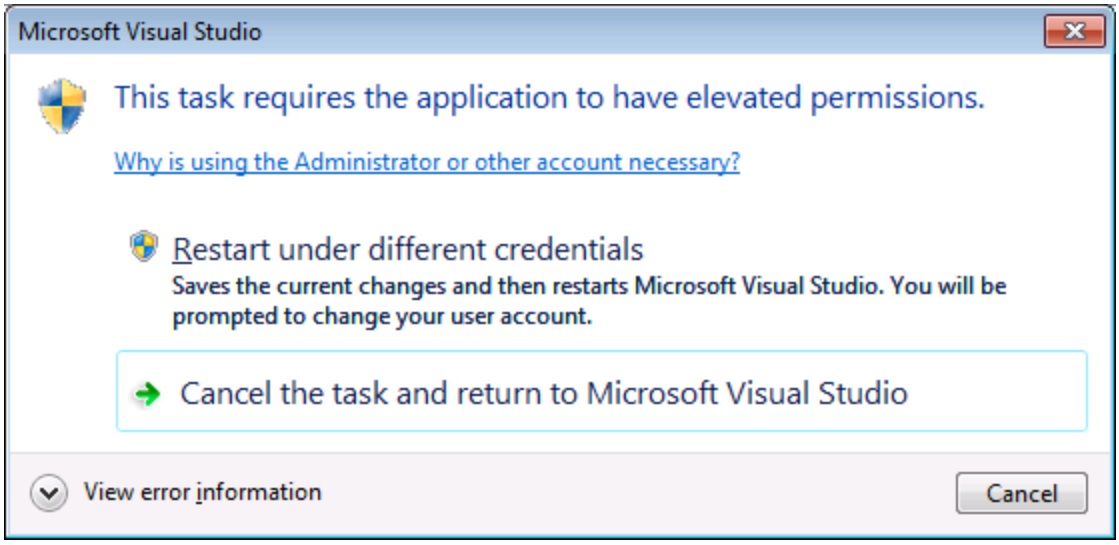
29.4.2 Hinzufügen der ActiveX Controls zur Toolbox

Um die XMLSpy ActiveX Controls in einer mit Visual Studio entwickelten Applikation verwenden zu können, fügen Sie diese folgendermaßen zu Ihrer Visual Studio Toolbox hinzu:

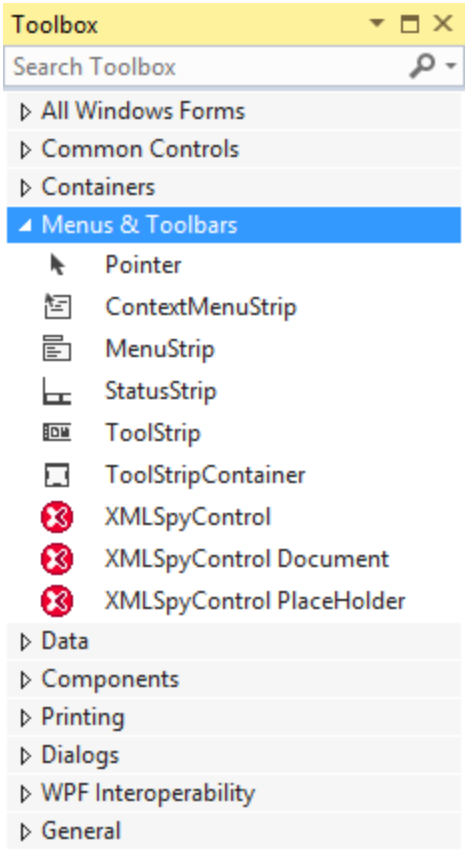
1. Klicken Sie im Menü **Tools** von Visual Studio auf **Choose Toolbox Items**.
2. Aktivieren Sie auf dem Register **COM Components** die Kontrollkästchen neben dem XMLSpyControl, dem XMLSpyControl-Dokument und dem XMLSpyControl-Platzhalter.

Falls die oben erwähnten Controls nicht zur Verfügung stehen, gehen Sie folgendermaßen vor:

1. Klicken Sie auf dem Register **COM Components** auf **Browse** und wählen Sie die Datei **XMLSpyControl.ocx** aus dem XMLSpy-Installationsordner aus. Beachten Sie, dass das XMLSpy-Integrationspaket installiert sein muss, da die Datei sonst nicht zur Verfügung steht (siehe Voraussetzungen 2073).
3. Sobald Sie aufgefordert werden, Visual Studio mit erweiterten Berechtigungen zu starten, klicken Sie auf **Restart under different credentials**.



Wenn die obigen Schritte erfolgreich ausgeführt wurden, stehen die XMLSpy ActiveX Controls in der Visual Studio Toolbox zur Verfügung.



Anmerkung: Bei einer Integration auf Applikationsebene wird nur das **XMLSpyControl** ActiveX Control verwendet (siehe Integration auf Applikationsebene[2076]). Die Controls **XMLSpyControl Document** und **XMLSpyControl Placeholder** werden für die Integration auf Dokumentebene verwendet (siehe Integration auf Dokumentebene[2078]).

29.4.3 Integration auf Applikationsebene

Bei der Integration auf Applikationsebene können Sie die gesamte Benutzeroberfläche von XMLSpy in ein Fenster Ihrer Applikation einbetten. Bei dieser Art von Integration steht Ihnen die gesamte Benutzeroberfläche von XMLSpy einschließlich aller Menüs, Symbolleisten, der Statusleiste, der Dokumentfenster und Eingabehilfen zur Verfügung. Die Anpassung der Benutzeroberfläche der Applikation ist auf die Optionen eingeschränkt, die XMLSpy bietet. Dazu gehören die Neuanordnung und Anpassung der Größe der Eingabehilfen und die Anpassung von Menüs und Symbolleisten.

Das einzige ActiveX Control, das Sie integrieren müssen, ist `XMLSpyControl`[2117]. Bei Integration auf Applikationsebene dürfen `XMLSpyControlDocument`[2125] oder `XMLSpyControlPlaceHolder`[2132] ActiveX Controls nicht instantiiert oder aufgerufen werden.

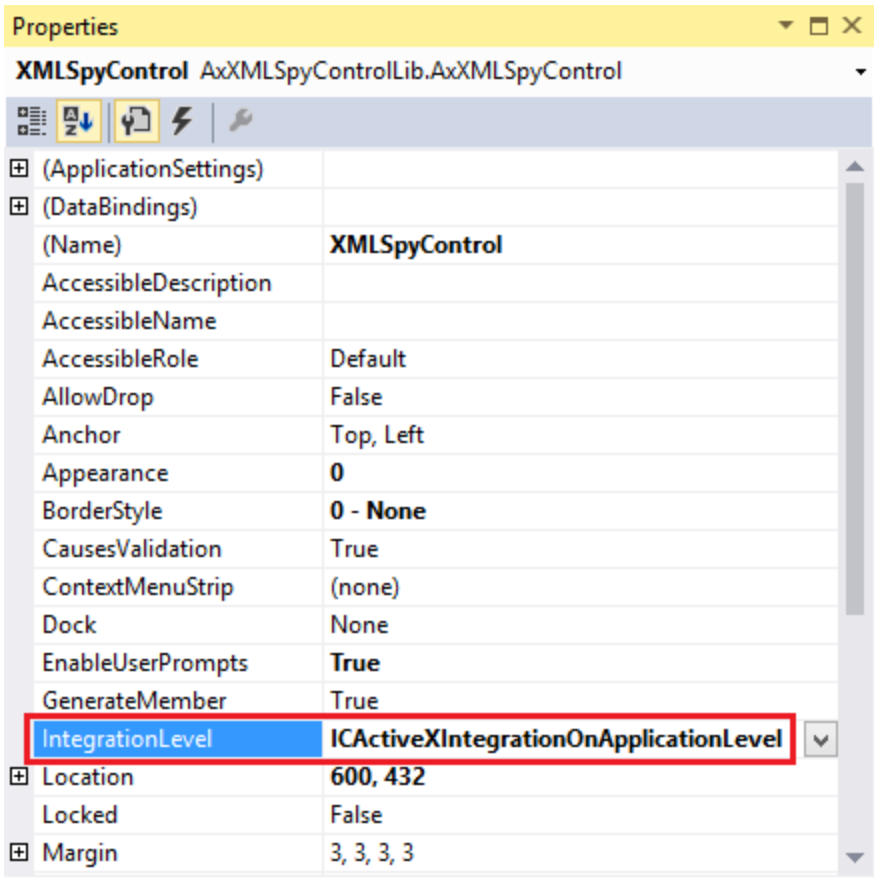
Wenn Sie Initialisierungen vornehmen müssen oder ein bestimmtes Verhalten von XMLSpy automatisieren wollen, verwenden Sie die für `XMLSpyControl`[2117] beschriebenen Eigenschaften, Methoden und Events. Um

 © 2016-2022 Altova GmbH

komplexere XMLSpy Funktionsaufrufe auszuführen, sollten Sie eventuell `XMLSpyControl.Application`[2118] verwenden.

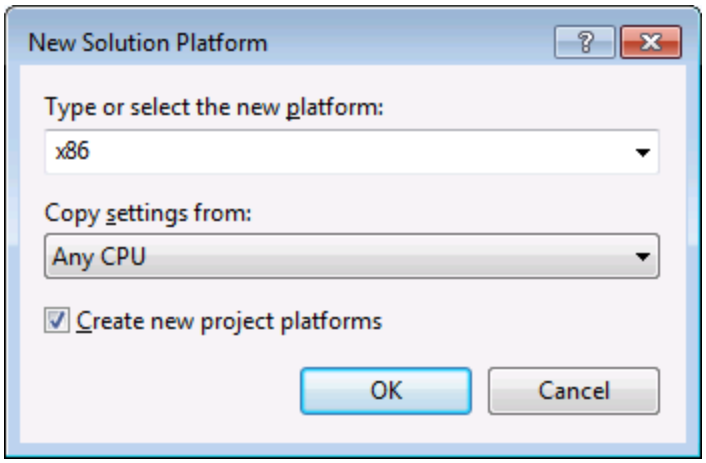
Gehen Sie in C# oder VB.NET bei Verwendung von Visual Studio folgendermaßen vor, um eine aus einem Formular bestehende Applikation zu erstellen, in der die XMLSpy ActiveX Controls auf Applikationsebene integriert sind:

1. Überprüfen Sie, ob alle Voraussetzungen erfüllt werden (siehe Voraussetzungen[2073]).
2. Erstellen Sie ein Visual Studio Windows Forms-Projekt mit einem neuen leeren Formular.
3. Fügen Sie die ActiveX Controls zur Toolbox hinzu, falls Sie das noch nicht getan haben (siehe Hinzufügen der ActiveX Controls zur Toolbox[2075]).
4. Ziehen Sie das **XMLSpyControl** aus der Toolbox in Ihr neues Formular.
5. Wählen Sie das **XMLSpyControl** im Formular aus und definieren Sie im Fenster "Properties" für die Eigenschaft **IntegrationLevel** den Wert **ICActiveXIntegrationOnApplicationLevel**.



6. Erstellen Sie eine Build-Plattform-Konfiguration für die Plattform, unter der Sie die Projektmappe erstellen möchten (x86, x64). So erstellen Sie die Build-Konfiguration:

   a. Klicken Sie mit der rechten Maustaste in Visual Studio auf die Projektmappen und wählen Sie **Configuration Manager**.
   b. Wählen Sie unter **Active solution platform** den Befehl **New...** und wählen Sie anschließend die x86- oder x64-Konfiguration aus (in diesem Beispiel **x86**).



Sie sind nun fertig und können die Projektmappe in Visual Studio erstellen und ausführen. Denken Sie daran, die Projektmappe mit der richtigen Konfiguration für Ihre Zielplattform (x86, x64) zu erstellen.

29.4.4 Integration auf Dokumentebene

Im Vergleich zur Integration auf Applikationsebene ist die Integration auf Dokumentebene komplexer, bietet aber dafür mehr Flexibilität beim Einbetten von XMLSpy-Funktionalitäten in Ihre Applikation mit Hilfe von ActiveX Controls. Auf diese Art haben Sie selektiven Zugriff auf die folgenden Teile der XMLSpy-Benutzeroberfläche:

- Dokument-Bearbeitungsfenster
- Projektfenster
- Eingabehilfenfenster
- Validierungsausgabefenster
- XPath Profiler-Fenster
- XPath-Dialogfeld-Fenster
- XSLT/XQuery Debugger-Fenster
- SOAP-Debugger-Fenster

Wie bereits im Abschnitt Integration auf Applikationsebene[2076] erwähnt, ist für eine ActiveX-Integration auf Applikationsebene nur ein Control, nämlich das **XMLSpyControl**, erforderlich. Für eine ActiveX-Integration auf Dokumentebene werden die XMLSpy-Funktionalitäten jedoch durch die folgenden ActiveX Controls zur Verfügung gestellt:

- XMLSpyControl[2117]
- XMLSpyControl Document[2125]
- XMLSpyControl Placeholder[2132]

Diese Controls werden durch die Datei **XMLSpyControl.ocx** aus dem Applikationsinstallationsordner von XMLSpy zur Verfügung gestellt. Wenn Sie die ActiveX-Integration mit Hilfe von Visual Studio erstellen, benötigen Sie über die Visual Studio Toolbox Zugriff auf diese Controls (siehe Hinzufügen der ActiveX Controls zur Toolbox[2075]).

Die grundlegenden Schritte, um die ActiveX Controls auf Dokumentebene in Ihre Applikation zu integrieren sind die folgenden:

© 2016-2022 Altova GmbH

1. Instantiieren Sie in Ihrer Applikation zuerst **XMLSpyControl**. Die Instantiierung dieses Control ist obligatorisch; Es ermöglicht die Unterstützung für die oben erwähnten Controls **XMLSpyControl Document** und **XMLSpyControl Placeholder**. Die Eigenschaft `IntegrationLevel`[2119] muss auf **ICActiveXIntegrationOnDocumentLevel** (oder "1") gesetzt werden. Um das Control für den Benutzer auszublenden, setzen Sie seine Eigenschaft **Visible** auf **False**.

Anmerkung: Verwenden Sie für die Integration auf Dokumentebene nicht die **Open**-Methode des **XMLSpyControl**, da dies zu unerwünschten Ergebnissen führen könnte. Verwenden Sie stattdessen die entsprechenden open-Methoden von **XMLSpyControl Document** und **XMLSpyControl PlaceHolder**.

2. Erstellen Sie mindestens eine Instanz von **XMLSpyControl Document** in Ihrer Applikation. Dieses Control stellt Ihrer Applikation das Dokumentbearbeitungsfenster von XMLSpy zur Verfügung und kann bei Bedarf mehrmals instantiiert werden.

   Verwenden Sie die Methode **Open,** um eine vorhandene Datei zu laden. Um Funktionen im Zusammenhang mit einem Dokument aufzurufen, verwenden Sie die Methoden **Path** und **Save** oder Methoden und Eigenschaften, die über die Eigenschaft **Document** zur Verfügung stehen.

Anmerkung: Das Control unterstützt keinen schreibgeschützten Modus. Der Wert der Eigenschaft **ReadOnly** wird ignoriert.

3. Fügen Sie optional für jedes weitere Fenster (bei dem es sich nicht um das Dokumentfenster handelt) und das Ihrer Applikation zur Verfügung stehen soll, den **XMLSpyControl Placeholder** zur Ihrer Applikation hinzu.

   Mit Hilfe von Instanzen von **XMLSpyControl PlaceHolder** können Sie selektiv zusätzliche Fenster von XMLSpy in Ihre Applikation einbetten. Die Art des Fensters (z.B. Projektfenster) wird durch die Eigenschaft **PlaceholderWindowID** definiert. Um daher die Art des Fensters zu definieren, definieren Sie die Eigenschaft **PlaceholderWindowID**. Gültige Fenster-IDs finden Sie unter XMLSpyControlPlaceholderWindow[2135].

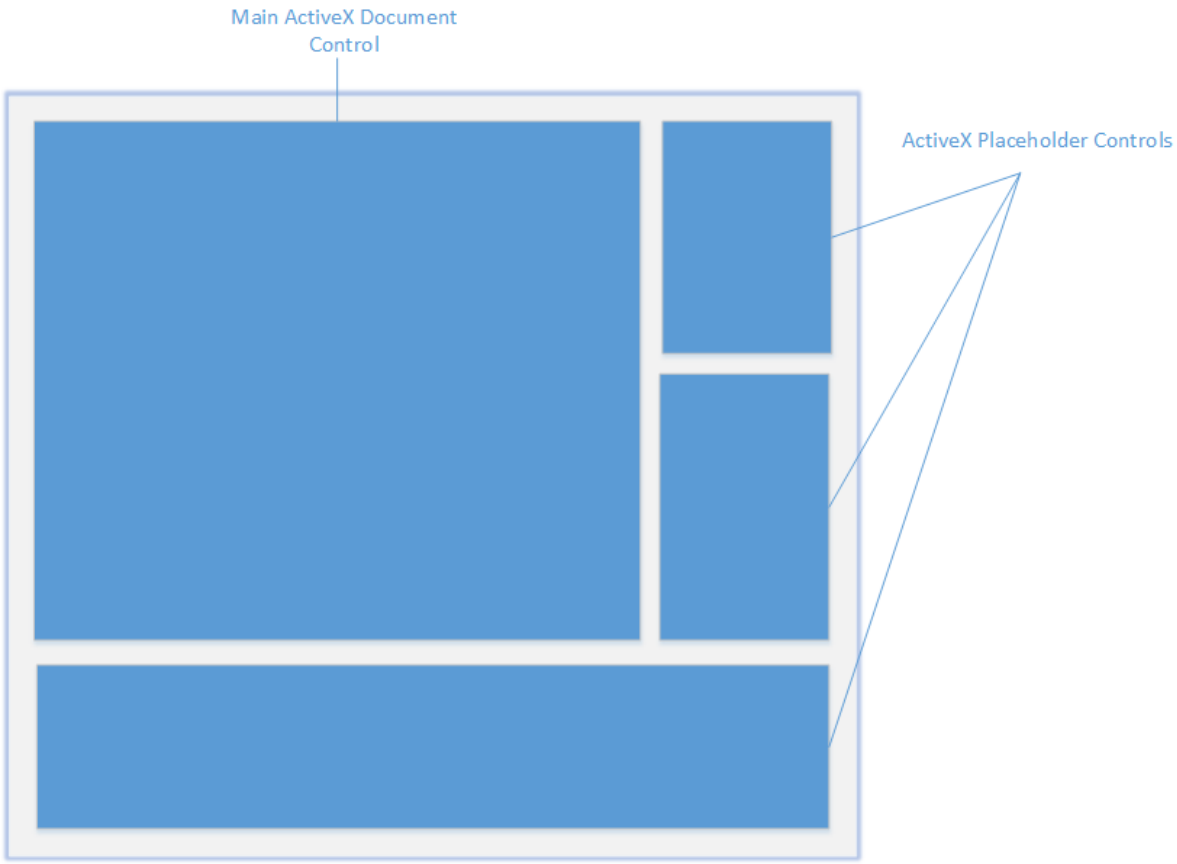
Anmerkung: Verwenden Sie für jeden Windows Identifier nur einen **XMLSpyControl PlaceHolder**.

   Für Placeholder Controls, die das XMLSpy-Projektfenster auswählen, stehen zusätzliche Methoden zur Verfügung. Mit Hilfe von **OpenProject** können Sie ein XMLSpy-Projekt laden. Mit Hilfe der Eigenschaft "Project" und der Methoden und Eigenschaften aus der XMLSpy Automation-Schnittstelle können Sie andere Operationen im Zusammenhang mit einem Projekt durchführen.

Ein Beispiel, wie Sie in C# oder VB.NET mit Visual Studio eine einfache, aus einem Formular bestehende Applikation, in der die XMLSpy ActiveX Controls auf Dokumentebene integriert sind, erstellen, sehen Sie in der Beschreibung unten. Beachten Sie, dass Ihre Applikation bei Bedarf auch komplexer sein kann. In der Anleitung unten sehen Sie jedoch, welche Voraussetzungen mindestens gegeben sein müssen, damit eine ActiveX-Integration auf Dokumentebene erfolgen kann.

1. Erstellen Sie ein neues Visual Studio Windows Forms-Projekt mit einem neuen leeren Formular.
2. Fügen Sie die ActiveX Controls zur Toolbox hinzu, falls dies noch nicht geschehen ist (siehe Hinzufügen der ActiveX Controls zur Toolbox[2075]).
3. Ziehen Sie das **XMLSpyControl**[2117] aus der Toolbox in Ihr neues Formular.
4. Setzen Sie die Eigenschaft **IntegrationLevel** des **XMLSpyControl** auf **ICActiveXIntegrationOnDocumentLevel** und die Eigenschaft **Visible** auf **False**. Sie können dies entweder über den Code oder über das Fenster **Properties** tun.

5. Ziehen Sie das **XMLSpyControl Document**[2125] aus der Toolbox in das Formular. Dieses Control stellt das Dokument-Hauptfenster von XMLSpy für Ihre Applikation zur Verfügung. Eventuell müssen Sie die Größe des Fensters an die Ihres Dokuments anpassen.
6. Fügen Sie optional ein oder mehrere **XMLSpyControl Placeholder**[2132] Controls zum Formular hinzu (eines für jeden zusätzlichen Fenstertyp, der für Ihre Applikation benötigt wird, z.B. das Projektfenster). Solche zusätzlichen Placeholder Controls werden normalerweise entweder unterhalb oder rechts oder links vom Dokument-Haupt-Control platziert z.B.:



7. Setzen Sie die Eigenschaft **PlaceholderWindowID** jedes **XMLSpyControl Placeholder** auf einen gültigen Fenster-Identifier. Eine Liste der gültigen Werte finden Sie unter XMLSpyControlPlaceholderWindow[2135].
8. Fügen Sie, wie unten gezeigt, Befehle zu Ihrer Applikation hinzu (Sie benötigen zumindest Befehle zum Öffnen, Speichern und Schließen von Dokumenten).

Abfragen von XMLSpy-Befehlen

Bei einer Integration auf Dokumentebene stehen in Ihrer Applikation kein XMLSpy-Menü- und keine XMLSpy-Symbolleiste zur Verfügung. Sie können statt dessen die benötigten Befehle aufrufen, deren Status anzeigen und diese programmatisch ausführen, wie folgt:

- Um alle verfügbaren Befehle abzurufen, verwenden Sie die Eigenschaft `CommandsList`[2118] des **XMLSpyControl**.
- Um die Befehle, geordnet nach ihrer Menüstruktur abzurufen, verwenden Sie die Eigenschaft `MainMenu`[2119].

© 2016-2022 Altova GmbH

- Um die Befehle, geordnet nach der Symbolleiste, in der sie vorkommen, abzurufen, verwenden Sie die Eigenschaft `Toolbars`[2120].
- Um Befehle an XMLSpy zu senden, verwenden Sie die Methode `Exec`[2121].
- Um abzufragen, ob ein Befehl gerade aktiviert oder deaktiviert ist, verwenden Sie die Methode `QueryStatus`[2122].

Auf diese Art können Sie XMLSpy-Befehle flexibel in die Menüs und Symbolleisten Ihrer Applikation integrieren.

Über Ihre Installation von XMLSpy stehen Ihnen auch Symbole für in XMLSpy verwendete Befehle zur Verfügung. Im Ordner **<Applicationsordner>\Examples\ActiveX\Images** Ihrer XMLSpy-Installation finden Sie Symbole im GIF-Format. Die Dateinamen entsprechen den im Abschnitt Befehlsreferenz[2094] aufgelisteten Befehlsnamen.

Allgemeines

Um das Verhalten von XMLSpy zu automatisieren, verwenden Sie die für **XMLSpyControl**[2117], **XMLSpyControl Document**[2125] und **XMLSpyControl Placeholder**[2132] beschriebenen Eigenschaften, Methoden und Ereignisse.

Für einen komplexeren Zugriff auf XMLSpy-Funktionalitäten sollten Sie die folgenden Eigenschaften verwenden:

- `XMLSpyControl.Application`[2118]
- `XMLSpyControlDocument.Document`[2126]
- `XMLSpyControlPlaceHolder.Project`[2133]

Über diese Eigenschaften erhalten Sie Zugriff auf die XMLSpy Automation-Schnittstelle (<%APPAPI%)

Anmerkung: Verwenden zum Öffnen eines Dokuments immer `XMLSpyControlDocument.Open`[2128] oder `XMLSpyControlDocument.New`[2128] für das entsprechenden Dokument-Control. Verwenden Sie zum Öffnen eines Projekts immer `XMLSpyControlPlaceHolder.OpenProject`[2133] in einem Placeholder Control, das ein XMLSpy-Projektfenster einbettet.

Beispiele, wie Sie die erforderlichen Controls in unterschiedlichen Programmierumgebungen instantiieren und aufrufen finden Sie unter Beispiele zur ActiveX-Integration[2081].

29.4.5 Beispiele zur ActiveX-Integration

Dieser Abschnitt enthält Beispiele für die Integration von XMLSpy auf Dokumentebene über unterschiedliche Container-Umgebungen und Programmiersprachen. Der Quellcode für alle Beispiele steht im Ordner `<ApplicationFolder>\Examples`\ActiveX Ihrer XMLSpy Installation zur Verfügung.

29.4.5.1 C#

Im Ordner `<ApplicationFolder>\Examples\ActiveX\C#` finden Sie ein einfaches Beispiel für eine ActiveX-Integration mittels C# und Visual Studio. Bevor Sie den Quellcode kompilieren und das Beispiel ausführen, stellen Sie sicher, dass alle Voraussetzungen erfüllt werden (siehe Ausführen der C#-Beispiellösung[2082]).

29.4.5.1.1 Ausführen der C#-Beispiellösung

In der Visual Studio-Beispielprojektmappe im Ordner **<ApplicationFolder>\Examples\ActiveX\C#** wird gezeigt, wie die XMLSpy ActiveX Controls verwendet werden. Beachten Sie die folgenden Schritte, bevor Sie versuchen, diese Projektmappen zu erstellen und auszuführen:

Schritt 1: Überprüfen Sie die Voraussetzungen

Um die Beispielprojektmappen öffnen zu können, benötigen Sie Visual Studio 2010 oder höher. Eine Liste der Voraussetzungen finden Sie unter Voraussetzungen[2073].

Schritt 2: Kopieren Sie das Beispiel in ein Verzeichnis, auf das Sie Schreibzugriff haben

Um Visual Studio nicht als Administrator ausführen zu müssen, kopieren Sie den Quellcode in ein Verzeichnis, auf das Sie Schreibzugriff haben, anstatt ihn vom Standardverzeichnis aus auszuführen.

Schritt 3: Überprüfen und definieren Sie alle erforderlichen Control-Eigenschaften

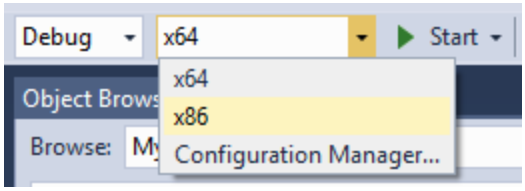
Die Beispielapplikation enthält eine Instanz von XMLSpyControlDocument[2125] und eine Instanz von XMLSpyControlPlaceHolder[2132] Controls. Überprüfen Sie noch einmal, ob die folgenden Eigenschaften dieser Controls, wie in der Tabelle unten angegeben, konfiguriert sind:

Control-Name	Eigenschaft	Eigenschaftswert
XMLSpyControl	IntegrationLevel	ICActiveXIntegrationOnDocumentLevel
XPathDialog	PlaceholderWindowID	16

So können Sie die Eigenschaften eines ActiveX Control anzeigen oder definieren:

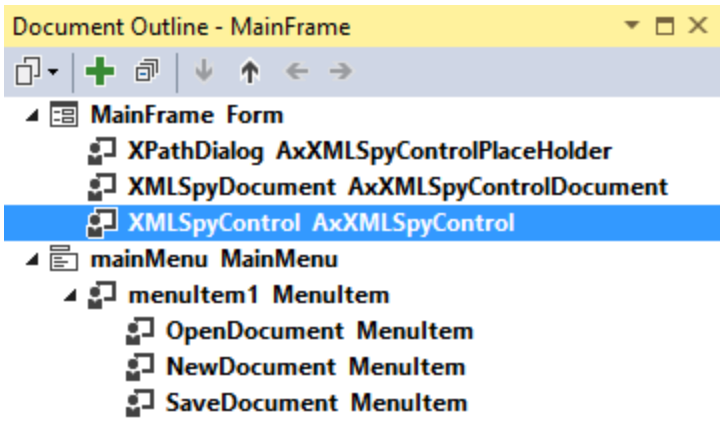
1. Öffnen Sie das Formular **MDIMain.cs** im Designer-Fenster.

Anmerkung: Unter 64-Bit Windows müssen Sie die Build-Konfiguration der Visual Studio-Projektmappen eventuell in "x86" ändern, **bevor** Sie das Designer-Fenster öffnen. Wenn Sie das Beispiel als 64-Bit-Applikation erstellen müssen, schlagen Sie nach unter Voraussetzungen[2073].

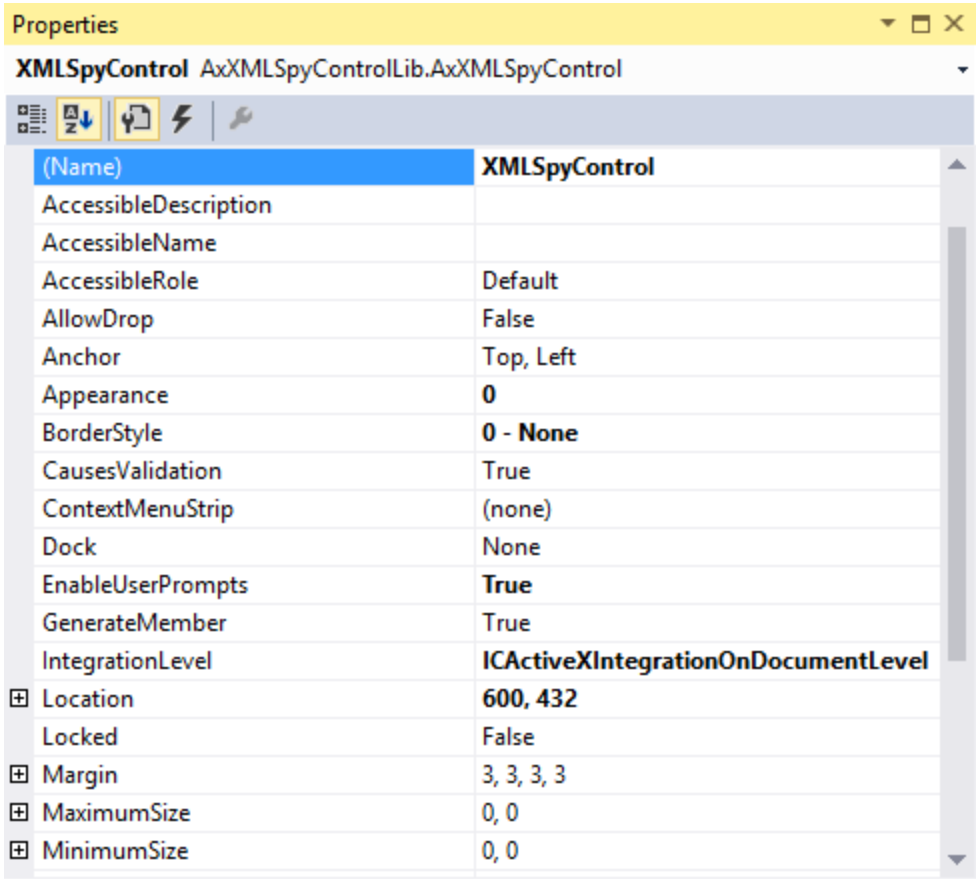


2. Öffnen Sie das **Document Outline**-Fenster von Visual Studio (Klicken Sie im Menü **View** auf **Other Windows | Document Outline**).

© 2016-2022 Altova GmbH



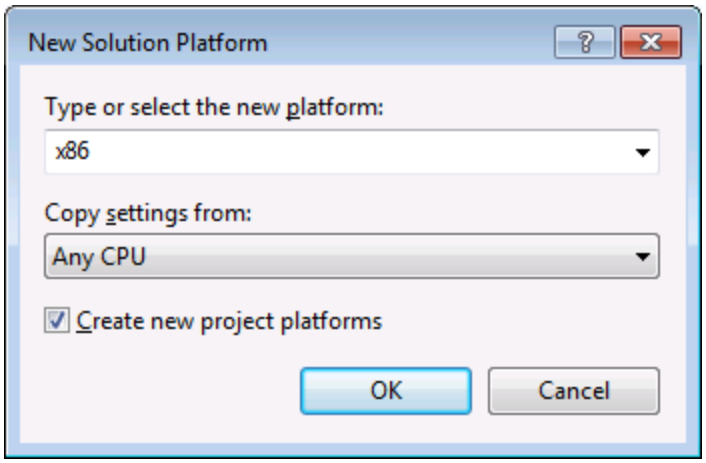
3. Klicken Sie im Fenster **Document Outline** auf ein ActiveX Control und bearbeiten Sie die benötigte Eigenschaft im Fenster **Properties,** z.B.:



Schritt 4: Definieren Sie die Build-Plattform

- Erstellen Sie eine Build-Plattform-Konfiguration für die Plattform, unter der Sie die Projektmappe erstellen möchten (x86, x64). So erstellen Sie die Build-Konfiguration:

a. Klicken Sie mit der rechten Maustaste auf die Projektmappe in Visual Studio und wählen Sie **Configuration Manager**.
b. Wählen Sie unter **Active solution platform** den Befehl **New...** und wählen Sie anschließend die x86- oder x64-Konfiguration (in diesem Beispiel **x86**).



Sie sind nun fertig und können die Projektmappe in Visual Studio erstellen und ausführen. Denken Sie daran, die Projektmappe mit der richtigen Konfiguration für Ihre Zielplattform (x86, x64) zu erstellen, da es sonst zu Laufzeitfehlern kommen kann.

Bei der Ausführung des Beispiels wird das MDI-Haupt-Frame-Fenster erstellt. Es enthält ein Bearbeitungsfenster mit einem leeren XML-Dokument und am unteren Rand ein XPath-Dialogfenster von XMLSpy. Öffnen Sie mit dem Befehl **Datei | Öffnen** eine XML-Datei aus dem XMLSpy-Beispielordner. Die Datei wird geladen und angezeigt. Nachdem Sie das Dokument geladen haben, können Sie im XPath-Dialogfenster arbeiten. Eventuell müssen Sie die rechte untere Ecke das Dialogfelds ein Stück mit der Maus ziehen, damit das Dialogfeld neu gezeichnet wird und der Inhalt darin angezeigt wird.

29.4.5.2 Java

XMLSpy ActiveX-Komponenten können von Java-Code aus aufgerufen werden. Die Java-Integration wird von den unten aufgelisteten Bibliotheken zur Verfügung gestellt. Diese Bibliotheken stehen im Ordner `<ApplicationFolder>\Examples\JavaAPI` Ihrer XMLSpy Installation zur Verfügung, nachdem Sie XMLSpy und das XMLSpy Integrationspaket installiert haben (siehe auch Voraussetzungen 2073).

- `AltovaAutomation.dll`: ein JNI Wrapper für Altova Automation Server (bei 32-Bit-Installationen von XMLSpy)
- `AltovaAutomation_x64.dll`: ein JNI-Wrapper für Altova Automation Server (bei 64-Bit-Installationen von XMLSpy)
- `AltovaAutomation.jar`: Java-Klassen zum Aufrufen von Altova Automation Servern
- `XMLSpyActiveX.jar`: Java-Klassen, die als Wrapper für die XMLSpy ActiveX-Schnittstelle dienen
- `XMLSpyActiveX_JavaDoc.zip`: eine Javadoc Datei, die die Hilfedokumentation zur Java API enthält

Anmerkung: Um die Java ActiveX Integration verwenden zu können, müssen sich die .dll- und .jar-Dateien im Java Class-Suchpfad befinden.

© 2016-2022 Altova GmbH

Java-Beispielprojekt

Im Lieferumfang Ihres Produkts ist ein Java-Beispielprojekt enthalten. Sie können das Java-Projekt nach Belieben testen und verwenden. Nähere Informationen finden Sie im Abschnitt Java-Beispielprojekt[2086].

Regeln für das Mappen der ActiveX Control-Namen auf Java

Eine Dokumentation zu den ActiveX Controls finden Sie in der Objektreferenz[2113]. Beachten Sie, dass sich die Objektbenennungskonventionen in Java etwas von anderen Sprachen unterscheiden. Für das Mapping zwischenden ActiveX Controls und dem Java Wrapper gelten die folgenden Regeln:

- ***Klassen und Klassennamen***
  Für jede Komponente des XMLSpy ActiveX Interface gibt es eine Java-Klasse mit dem Namen der Komponente.

- ***Methodennamen***
  Die Methodennamen im Java Interface sind dieselben wie die in den COM Interfaces, beginnen aber aufgrund der Java-Namenskonventionen mit einem Kleinbuchstaben. Zum Aufrufen von COM-Eigenschaften können dem Eigenschaftsnamen vorangestellte Java-Methoden mit `get` und `set` verwendet werden. Wenn eine Eigenschaft keinen Schreibzugriff ermöglicht, steht keine Setter-Methode zur Verfügung. Beispiel: Für die Eigenschaft `IntegrationLevel` des XMLSpyControl stehen die Java-Methoden `getIntegrationLevel` und `setIntegrationLevel` zur Verfügung.

- ***Enumerationen***
  Für jede im ActiveX Interface definierte Enumeration ist eine Java-Enumeration desselben Namens und mit denselben Werten definiert.

- ***Events und Event Handler***
  Für jedes Interface im Automation Interface, das Events unterstützt, steht ein Java-Interface desselben Namens plus '`Event`' zur Verfügung. Um das Überladen von Einzel-Events zu vereinfachen, gibt es eine Java-Klasse mit Standardimplementierungen für alle Events. Der Name dieser Java-Klasse ist der Name des Event Interface plus '`DefaultHandler`'. Beispiel:
  `XMLSpyControl`: Java-Klasse zum Aufrufen der Applikation
  `XMLSpyControlEvents`: Events Interface für das XMLSpyControl Control
  `XMLSpyControlEventsDefaultHandler`: Standard-Handler für `XMLSpyControlEvents`

Ausnahmen für Mapping-Regeln

Zu den oben aufgelisteten Regeln gibt es die folgenden Ausnahmen:

Interface	Java-Name
`XMLSpyControlDocument`, Methode `New`	`newDocument`
`Document`, Methode `SetEncoding`	`setFileEncoding`
`AuthenticView`, Methode `Goto`	`gotoElement`
`AuthenticRange`, Methode `Goto`	`gotoElement`
`AuthenticRange`, Methode `Clone`	`cloneRange`

Dieser Abschnitt

In diesem Abschnitt wird erklärt, wie einige grundlegende XMLSpy ActiveX-Funktionen über Java-Code aufgerufen werden können. Der Abschnitt ist in die folgenden Unterabschnitte gegliedert:

- Java-Beispielprojekt 2086
- Erstellen der ActiveX Controls 2088
- Laden der Daten in die Controls 2089
- Behandlung von Events 2089
- Menüs 2090
- Behandlung von UI Update Events 2091
- Erstellen einer XML-Baumstruktur 2092

29.4.5.2.1 Java-Beispielprojekt

Das XMLSpy-Installationspaket enthält ein Java-Beispielprojekt, das Sie Ordner "ActiveX Examples" des Applikationsordners `<ApplicationFolder>\Examples\ActiveX\Java\` finden.

Im Java-Beispiel wird gezeigt, wie Sie das XMLSpyControl in eine mit Java erstellte Desktop Applikation integrieren können. Sie können das Beispielprojekt mit Hilfe der Batch-Datei `BuildAndRun.bat`, direkt über die Befehlszeile testen oder Sie können es in Eclipse kompilieren und ausführen. Anleitungen dafür finden Sie weiter unten.

Dateiliste

Der Ordner für die Java-Beispiele enthält alle zum Ausführen des Beispielprojekts erforderlichen Dateien. Diese Dateien sind unten aufgelistet:

`.classpath`	Hilfedatei Eclipse-Projekt
`.project`	Eclipse-Projektdatei
`AltovaAutomation.dll`	Java-COM Bridge: DLL-Teil (für die 32-Bit-Installation)
`AltovaAutomation_x64.dll`	Java-COM Bridge: DLL-Teil (für die 64-Bit-Installation)
`AltovaAutomation.jar`	Java-COM Bridge: Java-Bibliotheksteil
`BuildAndRun.bat`	Batch-Datei zum Kompilieren und Ausführen des Beispielcodes über die Befehlszeile. Es wird ein Ordner benötigt, in dem sich die Java Virtual Machine als Parameter befindet.
`XMLSpyActiveX.jar`	Java-Klassen des XMLSpy ActiveX Control
`XMLSpyActiveX_JavaDoc.zip`	Javadoc Datei, die die Hilfedokumentation für die Java API enthält
`XMLSpyContainer.java`	Java-Beispielquellcode
`XMLSpyContainerEventHandler.java`	Java-Beispielquellcode
`XMLTreeDialog.java`	Java-Beispielquellcode

© 2016-2022 Altova GmbH

Funktionen in diesem Beispiel

Im Beispiel werden ein XMLSpy Dokument-Editor-Fenster, das XMLSpy-Projektfenster, das XMLSpy XPath-Fenster und eine XMLSpy-Eingabehilfe in ein AWT-Rahmenfenster platziert. Das für XMLSpy definierte Menü "Datei" wird ausgelesen und ein AWT-Menü mit derselben Struktur wird erstellt. Über dieses Menü oder über das Projektfenster können Sie Dateien im Dokument-Editor öffnen und damit arbeiten.

Sie können das Beispiel nach Ihren Wünschen modifizieren.

In den Codefragmenten werden die folgenden spezifischen Schritte beschrieben:

- Erstellen der ActiveX Controls[2088]: Startet XMLSpy, das als Automation Server registriert ist, bzw. aktiviert das Programm, wenn XMLSpy bereits ausgeführt wird.
- Laden der Daten in die Controls[2089]: Navigiert zu einem der mit XMLSpy installierten Beispieldokumente und öffnet es.
- Grundlegendes zur Event-Behandlung[2089]: Wechselt von der Ansicht aller offenen Dokumente in die Textansicht. Im Code wird auch gezeigt, wie man durch offene Dokumente iteriert.
- Menüs[2090]: Validiert das aktive Dokument und zeigt das Ergebnis in einem Meldungsfeld an. Im Code wird gezeigt, wie Ausgabeparameter verwendet werden.
- Behandlung von UI Update Events[2091]: Zeigt, wie XMLSpy Events behandelt werden.
-
- Erstellen einer XML-Baumstruktur[2092]: Zeigt wie, Sie eine XML-Baumstruktur erstellen und sie für die modale Aktivierung vorbereiten.

Aktualisieren des Pfads zum Ordner "Examples"

Bevor Sie das zur Verfügung gestellte Beispiel ausführen, müssen Sie die **XMLSpyContainer.java**-Datei möglicherweise bearbeiten und überprüfen, ob der folgende Pfad auf den tatsächlichen Ordner verweist, in dem die XMLSpy Beispieldateien auf Ihrem Betriebssystem gespeichert sind.

```
// Locate samples installed with the product.
final String strExamplesFolder = System.getenv( "USERPROFILE" ) + "\\Documents\\Altova\
\XMLSpy2023\\XMLSpyExamples\\";
```

Ausführen des Beispiels über die Befehlszeile

So führen Sie das Beispiel über die Befehlszeile aus:

1. Überprüfen Sie, ob alle Voraussetzungen erfüllt werden (siehe Voraussetzungen[2073]).
2. Öffnen Sie ein Eingabeaufforderungsfester, wechseln Sie in den Java-Beispielprojektordner und geben Sie folgende Zeile ein:

   ```
   buildAndRun.bat "<Pfad-zum-Java-bin-Ordner>"
   ```

1. Drücken Sie die **Eingabetaste**.

Der Java-Quellcode in `XMLSpyContainer.java` wird kompiliert und anschließend ausgeführt.

Kompilieren und Ausführen des Beispiels in Eclipse

So importieren Sie das Java-Beispielprojekt in Eclipse:

3. Überprüfen Sie, ob alle Voraussetzungen erfüllt werden (siehe Voraussetzungen 2073).
1. Klicken Sie im Menü **File** auf **Import**.
2. Wählen Sie **Existing Projects into Workspace** und navigieren Sie zur Eclipse-Projektdatei unter `<ApplicationFolder>\Examples\ActiveX\Java\`. Da Sie möglicherweise keinen Schreibzugriff auf diesen Ordner haben, wird empfohlen, im Importdialogfeld das Kontrollkästchen **Copy projects into workspace** zu aktivieren.

Um die Beispielapplikation zu starten, klicken Sie mit der rechten Maustaste im Package Explorer auf das Paket und wählen Sie den Befehl **Run as | Java Application**.

Hilfe zu Java API-Klassen steht in Form von Kommentaren im Code sowie als Javadoc-Ansicht von Eclipse zur Verfügung. Um die Javadoc-Ansicht in Eclipse zu aktivieren, wählen Sie den Menübefehl **Window | Show View | JavaDoc**.

29.4.5.2.2 Erstellen der ActiveX Controls

Im unten gezeigten Code sehen Sie wie ActiveX Controls erstellt werden. Die Konstruktoren erstellen die Java Wrapper-Objekte. Durch Hinzufügen dieser aus Canvas stammenden Objekte zu einem Bereich oder Rahmen wird die Erstellung des in Wrapper verpackten ActiveX-Objekts ausgelöst.

```
01  /**
02  * XMLSpy manager control - always needed
03  */
04    public static XMLSpyControl         xmlSpyControl = null;
05
06  /**
07  * XMLSpy document editing control
08  */
09    public static XMLSpyControlDocument     xmlSpyDocument = null;
10
11  /**
12  * Tool windows - XMLSpy place-holder controls
13  */
14    private static XMLSpyControlPlaceHolder   xmlSpyProjectToolWindow = null;
15    private static XMLSpyControlPlaceHolder   xmlSpyXpathToolWindow = null;
16    private static XMLSpyControlPlaceHolder   xmlSpyEHAttributeToolWindow = null;
17
18  // Create the XMLSpy ActiveX control; the parameter determines that we want
19  // to place document controls and place-holder controls individually.
20  // It gives us full control over the menu, as well.
21     xmlSpyControl = new XMLSpyControl(
       ICActiveXIntegrationLevel.ICActiveXIntegrationOnDocumentLevel.getValue() );
22     xmlSpyDocument = new XMLSpyControlDocument();
23     xmlSpyDocument.setPreferredSize( new Dimension ( 640, 480 ) );
24
25  // Create a project window and open the sample project in it
26     xmlSpyProjectToolWindow = new XMLSpyControlPlaceHolder(
27     XMLSpyControlPlaceholderWindow.XMLSpyControlProjectWindowToolWnd.getValue() );
```


© 2016-2022 Altova GmbH

```
28      xmlSpyProjectToolWindow.setPreferredSize( new Dimension( 200, 200 ) );
29      xmlSpyXpathToolWindow = new XMLSpyControlPlaceHolder(
        XMLSpyControlPlaceholderWindow.XMLSpyControlXPathDialogToolWnd.getValue() );
30      xmlSpyEHAttributeToolWindow = new XMLSpyControlPlaceHolder(
        XMLSpyControlPlaceholderWindow.XMLSpyControlEntryHelperTopToolWnd.getValue() );
31
32      frame.add( xmlSpyControl, BorderLayout.NORTH );
33      frame.add( xmlSpyDocument, BorderLayout.CENTER );
34      southPanel.add( xmlSpyProjectToolWindow );
35      southPanel.add( xmlSpyXpathToolWindow );
36      southPanel.add( xmlSpyEHAttributeToolWindow );
).
```

29.4.5.2.3 Laden der Daten in die Controls

Im unten gezeigten Code sehen Sie wie Daten in ActiveX Controls geladen werden können.

```
1     // Locate samples installed with the product.
2        final String strExamplesFolder = System.getenv( "USERPROFILE" ) +
         "\\Documents\\Altova\\XMLSpy2022\\Examples\\";
3        xmlSpyProjectToolWindow.openProject( strExamplesFolder  + "Examples.spp" );
```

29.4.5.2.4 Grundlegendes zur Event-Behandlung

Im unten gezeigten Code sehen Sie wie grundlegende Events behandelt werden. Bei Aufruf der XMLSpyControl-Methode `open` oder beim Öffnen einer Datei über das Menü oder die Projektstruktur wird das `onOpenedOrFocused` Event an den dazugehörigen Event Handler gesendet. Im Prinzip wird dieses Event behandelt, indem die Datei durch Aufruf der `open`-Methode des XMLSpyDocumentControl geöffnet wird.

```
01        // Open the PXF file when button is pressed
02        btnOpenPxf.addActionListener( new ActionListener() {
03          public void actionPerformed(ActionEvent e) {
04            try {
05              xmlSpyControl.open( strExamplesFolder + "OrgChart.pxf" );
06            } catch (AutomationException e1) {
07              e1.printStackTrace();
08            }
09          }
10        } );
11        public void onOpenedOrFocused( String i_strFileName, boolean
i_bOpenWithThisControl, boolean i_bFileAlreadyOpened ) throws AutomationException
12    {
13      // Handle the New/Open events coming from the Project tree or from the menus
14      if ( !i_bFileAlreadyOpened )
15      {
16        // This is basically an SDI interface, so open the file in the already existing
document control
17        try {
18          XMLSpyContainer.xmlSpyDocument.open( i_strFileName );
19          XMLSpyContainer.xmlSpyDocument.requestFocusInWindow();
20        } catch (Exception e) {
```

```
21          e.printStackTrace();
22        }
23      }
24    }
```

29.4.5.2.5 Menüs

Im unten gezeigten Code sehen Sie, wie Menüeinträge erstellt werden können. Jedes `XMLSpyCommand` Objekt holt ein entsprechendes `MenuItem` Objekt, wobei der `ActionCommand` auf die ID des Befehls gesetzt wird. Die von allen Menüeinträgen generierten Aktionen werden von derselben Funktion gehandelt. Diese kann bestimmte Handlings (wie z.B Neuinterpretieren der Schließfunktion) durchführen oder die Ausführung durch Aufruf seiner `exec`-Methode an das `XMLSpyControl` Objekt delegieren. Das `menuMap` Objekt, das bei der Menüerstellung befüllt wird, wird später verwendet (siehe Abschnitt Behandlung von UI Update Events 2091).

```
01  // Load the file menu when the button is pressed
02        btnMenu.addActionListener( new ActionListener() {
03          public void actionPerformed(ActionEvent e) {
04            try {
05              // Create the menubar that will be attached to the frame
06              MenuBar mb = new MenuBar();
07              // Load the main menu's first item - the File menu
08              XMLSpyCommand xmlSpyMenu =
xmlSpyControl.getMainMenu().getSubCommands().getItem( 0 );
09              // Create Java menu items from the Commands objects
10              Menu fileMenu = new Menu();
11              handlerObject.fillMenu( fileMenu, xmlSpyMenu.getSubCommands() );
12              fileMenu.setLabel( xmlSpyMenu.getLabel().replace( "&", "" ) );
13              mb.add( fileMenu );
14              frame.setMenuBar( mb );
15              frame.validate();
16            } catch (AutomationException e1) {
17              e1.printStackTrace();
18            }
19            // Disable the button when the action has been performed
20            ((AbstractButton) e.getSource()).setEnabled( false );
21          }
22        } ) ;
23 /** * Populates a menu with the commands and submenus contained in an XMLSpyCommands
object  */
24        public void fillMenu(Menu newMenu, XMLSpyCommands xmlSpyMenu) throws
AutomationException
25    {
26      // For each command/submenu in the xmlSpyMenu
27      for ( int i = 0 ; i < xmlSpyMenu.getCount() ; ++i  )
28      {
29        XMLSpyCommand xmlSpyCommand = xmlSpyMenu.getItem( i );
30        if ( xmlSpyCommand.getIsSeparator() )
31          newMenu.addSeparator();
32        else
33        {
34          XMLSpyCommands subCommands = xmlSpyCommand.getSubCommands();
35          // Is it a command (leaf), or a submenu?
36          if ( subCommands.isNull() || subCommands.getCount() == 0 )
37          {
```

 © 2016-2022 Altova GmbH

```
38            // Command -> add it to the menu, set its ActionCommand to its ID and store in
in the menuMap
39            MenuItem mi = new MenuItem( xmlSpyCommand.getLabel().replace( "&", "" ) );
40            mi.setActionCommand( "" + xmlSpyCommand.getID() );
41            mi.addActionListener( this );
42            newMenu.add( mi );
43            menuMap.put( xmlSpyCommand.getID(), mi );
44          }
45          else
46          {
47            // Submenu -> create submenu and repeat recursively
48            Menu newSubMenu = new Menu();
49            fillMenu( newSubMenu, subCommands );
50            newSubMenu.setLabel( xmlSpyCommand.getLabel().replace( "&", "" ) );
51            newMenu.add( newSubMenu );
52          }
53        }
54      }
55    }
56
57     /**
58     * Action handler for the menu items
59     * Called when the user selects a menu item; the item's action command corresponds to
the command table for XMLSpy
60     */
61    public void actionPerformed( ActionEvent e )
62    {
63      try
64      {
65        int iCmd = Integer.parseInt( e.getActionCommand() );
66        // Handle explicitly the Close commands
67        switch ( iCmd )
68        {
69          case 57602:       // Close
70          case 34050:       // Close All
71            XMLSpyContainer.initXmlSpyDocument();
72            break;
73          default:
74            XMLSpyContainer.xmlSpyControl.exec( iCmd );
75            break;
76        }
77      }
78      catch ( Exception ex )
79      {
80        ex.printStackTrace();
81      }
82
83    }
```

29.4.5.2.6 Behandlung von UI Update Events

Im unten gezeigten Code, sehen Sie wie ein UI-Update Event Handler erstellt werden kann.

```
01 /**
02     * Call-back from the XMLSpyControl.
03     * Called to enable/disable commands
```

```
04     */
05    @Override
06    public void onUpdateCmdUI() throws AutomationException
07    {
08      // A command should be enabled if the result of queryStatus contains the Supported
(1) and Enabled (2) flags
09      for ( java.util.Map.Entry<Integer, MenuItem> pair : menuMap.entrySet() )
10
       pair.getValue().setEnabled( XMLSpyContainer.xmlSpyControl.queryStatus( pair.getKey()
) > 2 );
11    }
12     /**
13     * Call-back from the XMLSpyControl.
14     * Usually called while enabling/disabling commands due to UI updates
15     */
16    @Override
17    public boolean onIsActiveEditor( String i_strFilePath ) throws AutomationException
18    {
19      try {
20        return
XMLSpyContainer.xmlSpyDocument.getDocument().getFullName().equalsIgnoreCase( i_strFilePath
);
21      } catch ( Exception e ) {
22        return false;
23      }
24    }
```

29.4.5.2.7 Erstellen einer XML-Baumstruktur

Im unten gezeigten Code wird ein XML-Datenobjekt in Form von Nodes in eine Baumstruktur geladen.

```
01 // access required XMLSpy Java-COM classes
02 import com.altova.automation.XMLSpy.XMLData;
03
04 // access AWT and Swing components
05 import java.awt.*;
06 import javax.swing.*;
07 import javax.swing.tree.*;
08
09 /**
10  * A simple example of a tree control loading the structure from an XMLData object.
11  * The class receives an XMLData object, loads its nodes in a JTree, and prepares
12  * for modal activation.
13  *
14  * Feel free to modify and extend this sample.
15  *
16  * @author Altova GmbH
17  */
18 class XMLTreeDialog extends JDialog
19 {
20   /**
21    * The tree control
22    */
23   private JTree myTree;
```

```
24
25    /**
26     * Root node of the tree control
27     */
28    private DefaultMutableTreeNode top ;
29
30    /**
31     * Constructor that prepares the modal dialog containing the filled tree control
32     * @param xml    The data to be displayed in the tree
33     * @param parent  Parent frame
34     */
35    public XMLTreeDialog( XMLData xml, Frame parent )
36    {
37      // Construct the modal dialog
38      super( parent, "XML tree", true );
39      // Arrange controls in the dialog
40      top = new DefaultMutableTreeNode("root");
41      myTree = new JTree(top);
42      setContentPane( new JScrollPane( myTree ) );
43      // Build up the tree
44      fillTree( top, xml );
45      myTree.expandRow( 0 );
46    }
47
48    /**
49     * Loads the nodes of an XML element under a given tree node
50     * @param node  Target tree node
51     * @param elem  Source XML element
52     */
53    private void fillTree( DefaultMutableTreeNode node, XMLData elem)
54    {
55      try
56      {
57        // There are several ways to iterate through child elements: either using the
getFirstChild/getNextChild,
58        // or by incrementing an index up to countChildren and calling getChild [as shown
below].
59        // If you only want to get childer of one kind, you should use
countChildrenKind/getChildKind,
60        // or provide a kind to the getFirstChild before iterating with the getNextChild.
61        int nSize = elem.countChildren() ;
62        for ( int i = 0 ; i < nSize ; ++i)
63        {
64          // Create a new tree node for each child element, and continue recursively
65          XMLData newElem = elem.getChild(i) ;
66          DefaultMutableTreeNode newNode = new DefaultMutableTreeNode( newElem.getName() )
;
67          node.add( newNode ) ;
68          fillTree( newNode, newElem ) ;
69        }
70      }
71      catch (Exception e)
72      {
73        e.printStackTrace();
74      }
75    }
76
77  }
```

29.4.6 Befehlsreferenz

In diesem Abschnitt werden die Namen und Identifier aller Menübefehle aufgelistet, die in XMLSpy zur Verfügung stehen. In den einzelnen Unterabschnitten werden jeweils die Befehle aus dem entsprechenden Menü von XMLSpy aufgelistet. Die Befehlstabellen sind folgendermaßen gegliedert:

- In der Spalte "Menübefehl" sehen Sie den Menütext des Befehls, wie er in XMLSpy, angezeigt wird, damit Sie die dem Befehl zugrunde liegende Funktionalität leichter erkennen.
- In der Spalte "Befehlsname" ist der String angegeben, anhand dessen ein Symbol desselben Namens aus dem Ordner **ActiveX\Images** des XMLSpy Installationsverzeichnis aufgerufen werden kann.
- In den "ID"-Spalten sehen Sie die numerischen Identifier der Spalte, die als Argument an die Methoden geliefert wird, die diesen Befehl ausführen oder abfragen.

Um einen Befehl auszuführen, verwenden Sie die Methoden `XMLSpyControl.Exec`[2121] oder `XMLSpyControlDocument.Exec`[2128]. Um den Status eines Befehls abzufragen, verwenden Sie die Methoden `XMLSpyControl.QueryStatus`[2122] oder `XMLSpyControlDocument.QueryStatus`[2128].

Einige dieser Befehle werden je nach installierter XMLSpy Edition eventuell nicht unterstützt.

29.4.6.1 Menü "Datei"

Das Menü "Datei" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Neu...	ID_FILE_NEW	57600
Öffnen...	ID_FILE_OPEN	57601
Neu laden	IDC_FILE_RELOAD	34065
Kodierung...	IDC_ENCODING	34061
Schließen	ID_FILE_CLOSE	57602
Alle schließen	IDC_CLOSE_ALL	34050
Alle inaktiven schließen	IDC_CLOSE_OTHERS	34271
Speichern	ID_FILE_SAVE	57603
Speichern unter...	ID_FILE_SAVE_AS	57604
Alles speichern	ID_FILE_SAVE_ALL	34208
Als Mail senden...	ID_FILE_SEND_MAIL	57612
Drucken...	ID_FILE_PRINT	57607

© 2016-2022 Altova GmbH

Menübefehl	Befehlsname	ID
Druckvorschau	IDC_PRINT_PREVIEW	34104
Druckereinrichtung...	ID_FILE_PRINT_SETUP	57606
Letzte Datei	ID_FILE_MRU_FILE1	57616
Beenden	ID_APP_EXIT	57665

29.4.6.2 Menü "Bearbeiten"

Das Menü "Bearbeiten" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Rückgängig	ID_EDIT_UNDO	57643
Wiederherstellen	ID_EDIT_REDO	57644
Ausschneiden	ID_EDIT_CUT	57635
Kopieren	ID_EDIT_COPY	57634
Einfügen	ID_EDIT_PASTE	57637
Löschen	ID_EDIT_CLEAR	57632
Als XML-Text kopieren	IDC_COPY_AS_XML_TEXT	33443
Als strukturierten Text kopieren	IDC_COPY_AS_STRUCTURED_TEXT	33442
XPath kopieren	IDC_COPY_XPATH	33444
XPointer kopieren	IDC_COPY_XPOINTER	33445
Dateipfad...	IDC_EDIT_INSERT_PATH_STRING	34013
XInclude...	IDC_EDIT_INSERT_XINCLUDE_STRING	34017
Externe kodierte Datei...	IDC_EDIT_INSERT_ENCODED_BINARY_STRING	34273
Pretty-Print	IDC_PRETTY_PRINT	34101
Whitespaces entfernen	IDC_STRIP_WHITESPACES	34296
Alles markieren	ID_EDIT_SELECT_ALL	57642
Suchen...	ID_EDIT_FIND	57636
Weitersuchen	ID_EDIT_REPEAT	57640
Ersetzen...	ID_EDIT_REPLACE	57641

Menübefehl	Befehlsname	ID
In Dateien suchen...	IDC_FIND_IN_FILES	34000
Lesezeichen einfügen/löschen	IDC_TOGGLE_BOOKMARK	34162
Alle Lesezeichen löschen	IDC_REMOVEALLBOOKMARKS	34132
Zum nächsten Lesezeichen	IDC_GOTONEXTBOOKMARK	34070
Zum vorhergehenden Lesezeichen	IDC_GOTOPREVBOOKMARK	34071
Ein-/Auskommentieren	IDC_TOGGLE_XML_COMMENT	34029

29.4.6.3 Menü "Projekt"

Das Menü "Projekt" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Neues Projekt	IDC_ICPROJECTGUI_NEW	37200
Projekt öffnen...	IDC_ICPROJECTGUI_OPEN	37201
Projekt neu laden	IDC_ICPROJECTGUI_RELOAD	37202
Projekt schließen	IDC_ICPROJECTGUI_CLOSE	37203
Projekt speichern	IDC_ICPROJECTGUI_SAVE	37204
Projekt speichern unter...	IDC_ICPROJECTGUI_SAVE_AS	37207
Versionskontrolle aktivieren	ID_SCC_ENABLE	38602
Dateien zu Projekt hinzufügen...	IDC_ICPROJECTGUI_ADD_FILES_TO_PROJECT	37205
Globale Ressource zu Projekt hinzufügen...	IDC_ICPROJECTGUI_ADD_GLOBAL_RESOURCE_TO_PROJECT	37239
URL zu Projekt hinzufügen...	IDC_ICPROJECTGUI_ADD_URL_TO_PROJECT	37206
Aktive Datei zu Projekt hinzufügen	IDC_ICPROJECTGUI_ADD_ACTIVE_FILE_TO_PROJECT	37208
Aktive und verwandte Dateien zu Projekt hinzufügen	IDC_ICPROJECTGUI_ADD_ACTIVE_AND_RELATED_FILES_TO_PROJECT	37209
Projektordner zu Projekt hinzufügen...	IDC_ICPROJECTGUI_ADD_FOLDER_TO_PROJECT	37210

© 2016-2022 Altova GmbH

Menübefehl	Befehlsname	ID
Externes Verzeichnis zum Projekt hinzufügen...	IDC_ICPROJECTGUI_ADD_EXT_FOLDER_TO_PROJECT	37211
Externes Webverzeichnis zum Projekt hinzufügen...	IDC_ICPROJECTGUI_ADD_EXT_URL_FOLDER_TO_PROJECT	37212
Skript-Einstellungen...	IDC_PROJECT_SCRIPT_SETTINGS	34136
Eigenschaften...	IDC_ICPROJECTGUI_PROJECT_PROPERTIES	37223
Letztes Projekt	IDC_ICPROJECTGUI_RECENT	37224

29.4.6.4 Menü "XML"

Das Menü "XML" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Attribute	IDC_INSERT_ATTRIBUTE	33449
Element	IDC_INSERT_STRUCT	33459
Text	IDC_INSERT_TEXT	33460
CDATA	IDC_INSERT_CDATA	33450
Comment	IDC_INSERT_COMMENT	33451
XML	IDC_INSERT_XML	33461
Processing Instruction	IDC_INSERT_PI	33458
XInclude...	IDC_INSERT_XINCLUDE	34019
DOCTYPE	IDC_INSERT_DEF_DOCTYPE	33453
ExternalID	IDC_INSERT_DEF_EXTERNAL_ID	33456
ELEMENT	IDC_INSERT_DEF_ELEMENT	33454
ATTLIST	IDC_INSERT_DEF_ATTLIST	33452
ENTITY	IDC_INSERT_DEF_ENTITY	33455
NOTATION	IDC_INSERT_DEF_NOTATION	33457
Externe kodierte Datei...	IDC_INSERT_ENCODED_BINARY	34274
Attribute	IDC_APPEND_ATTRIBUTE	33415
Element	IDC_APPEND_STRUCT	33425

Menübefehl	Befehlsname	ID
Text	IDC_APPEND_TEXT	33426
CDATA	IDC_APPEND_CDATA	33416
Comment	IDC_APPEND_COMMENT	33417
XML	IDC_APPEND_XML	33427
Processing Instruction	IDC_APPEND_PI	33424
XInclude...	IDC_APPEND_XINCLUDE	34026
DOCTYPE	IDC_APPEND_DEF_DOCTYPE	33419
ExternalID	IDC_APPEND_DEF_EXTERNAL_ID	33422
ELEMENT	IDC_APPEND_DEF_ELEMENT	33420
ATTLIST	IDC_APPEND_DEF_ATTLIST	33418
ENTITY	IDC_APPEND_DEF_ENTITY	33421
NOTATION	IDC_APPEND_DEF_NOTATION	33423
Externe kodierte Datei...	IDC_APPEND_ENCODED_BINARY	34276
Attribute	IDC_ADD_CHILD_ATTRIBUTE	33402
Element	IDC_ADD_CHILD_STRUCT	33412
Text	IDC_ADD_CHILD_TEXT	33413
CDATA	IDC_ADD_CHILD_CDATA	33403
Comment	IDC_ADD_CHILD_COMMENT	33404
XML	IDC_ADD_CHILD_XML	33414
Processing Instruction	IDC_ADD_CHILD_PI	33411
XInclude...	IDC_ADD_CHILD_XINCLUDE	34027
DOCTYPE	IDC_ADD_CHILD_DEF_DOCTYPE	33406
ExternalID	IDC_ADD_CHILD_DEF_EXTERNAL_ID	33409
ELEMENT	IDC_ADD_CHILD_DEF_ELEMENT	33407
ATTLIST	IDC_ADD_CHILD_DEF_ATTLIST	33405
ENTITY	IDC_ADD_CHILD_DEF_ENTITY	33408
NOTATION	IDC_ADD_CHILD_DEF_NOTATION	33410
Externe kodierte Datei...	IDC_ADD_CHILD_ENCODED_BINARY	34277

© 2016-2022 Altova GmbH

Menübefehl	Befehlsname	ID
Attribute	IDC_CONVERT_TO_ATTRIBUTE	33429
Element	IDC_CONVERT_TO_STRUCT	33439
Text	IDC_CONVERT_TO_TEXT	33440
CDATA	IDC_CONVERT_TO_CDATA	33430
Comment	IDC_CONVERT_TO_COMMENT	33431
XML	IDC_CONVERT_TO_XML	33441
Processing Instruction	IDC_CONVERT_TO_PI	33438
DOCTYPE	IDC_CONVERT_TO_DEF_DOCTYPE	33433
ExternalID	IDC_CONVERT_TO_DEF_EXTERNAL_ID	33436
ELEMENT	IDC_CONVERT_TO_DEF_ELEMENT	33434
ATTLIST	IDC_CONVERT_TO_DEF_ATTLIST	33432
ENTITY	IDC_CONVERT_TO_DEF_ENTITY	33435
NOTATION	IDC_CONVERT_TO_DEF_NOTATION	33437
Als Tabelle anzeigen	IDC_GRID_VIEW_AS_TABLE	34075
Zeile einfügen	IDC_TABLE_INSERT_ROW	34158
Zeile anhängen	IDC_TABLE_APPEND_ROW	34157
Aufsteigend sortieren	IDC_TABLE_SORT_ASC	33464
Absteigend sortieren	IDC_TABLE_SORT_DESC	33465
Höher stufen	IDC_MOVE_LEFT	34091
Tiefer stufen	IDC_MOVE_RIGHT	34092
In Element einbinden	IDC_ENCLOSE_IN_ELEMENT	33446
XPath auswerten...	IDC_EVALUATE_XPATH	34007
Wohlgeformtheit prüfen	IDC_CHECK_WELL_FORM	34049
XML validieren	IDC_VALIDATE	32954
XML am Server validieren (Hochleistungsvalidierung)	IDC_VALIDATE_RAPTOR	34309
Eingabehilfen aktualisieren	IDC_UPDATE_ELEMENT_CHOICE	34173
Namespace Präfix...	IDC_NAMESPACE	33462

Menübefehl	Befehlsname	ID
XML-Signatur erstellen...	IDC_XML_SIGNATURE_CREATE	34280
XML-Signatur überprüfen...	IDC_XML_SIGNATURE_VERIFY	34281

29.4.6.5 Menü "DTD/Schema"

Das Menü "DTD/Schema" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
DTD zuweisen...	IDC_ASSIGN_DTD	34032
Schema zuweisen...	IDC_ASSIGN_SCHEMA	34033
Weitere DTD einfügen...	IDC_INCLUDE_DTD	34084
Gehe zu DTD	IDC_GOTO_DTD	34072
Gehe zu Schema	IDC_GOTO_SCHEMA	34074
Gehe zu Definition	IDC_GOTO_DEFINITION	33447
DTD/Schema generieren...	IDC_GENERATE_DTD_SCHEMA	34068
DTD nivellieren...	IDC_FLATTEN_DTD	34301
DTD in Schema konvertieren...	IDC_CONVERT_DTD_TO_SCHEMA	34299
Schema nivellieren...	IDC_FLATTEN_SCHEMA	34302
Schema in DTD konvertieren...	IDC_CONVERT_SCHEMA_TO_DTD	34300
In UML konvertieren...	IDC_CONVERT_SCHEMA_TO_UML	34008
XML anhand von DB, Excel, EDI in MapForce generieren...	IDC_DTD_OPENIN_MAPFORCE	34056
HTML/PDF/Word-Ausgabe in StyleVision erstellen...	IDC_DTD_OPENIN_STYLEVISION	34057
XML/JSON-Beispieldatei generieren...	IDC_GENERATE_XML_FROM_SCHEMA	34069
Programmcode generieren...	IDC_GENERATE_CODE_FROM_SCHEMA	34067
Cache-Speicher leeren	IDC_FLUSH_CACHED_FILES	34066

29.4.6.6 Menü "Schema-Design"

Das Menü "Schema-Design" enthält die folgenden Befehle:

© 2016-2022 Altova GmbH

Menübefehl	Befehlsname	ID
Schema-Einstellungen...	IDC_SCHEMA_NAMESPACES	33571
Diagramm speichern...	IDC_SCHEMA_SAVE_DIAGRAM	33581
Dokumentation generieren...	IDC_SCHEMA_DOCUMENTATION	34146
Ansicht konfigurieren...	IDC_SCHEMA_VIEW_CONFIG	33593
Vergrößern/Verkleinern..	IDC_SCHEMA_ZOOM	34150
Globale Elemente anzeigen	IDC_SCHEMA_MODE_GLOBALS	34147
Diagramm anzeigen	IDC_SCHEMA_MODE_DIAGRAM	33570
Oracle Schema Extensions aktivieren	IDC_SCHEMA_ORACLE_EXTENSIONS	33577
Oracle Schema-Einstellungen...	IDC_SCHEMA_ORACLE_SCHEMA_SETTINGS	33578
Microsoft SQL Server Schema Extensions aktivieren	IDC_SCHEMA_SQLSERVER_EXTENSIONS	33588
Named Schema Relationships...	IDC_SCHEMA_SQLSERVER_GLOBAL_RELATIONSHIPS	33589
Unnamed Element Relationships...	IDC_SCHEMA_SQLSERVER_LOCAL_RELATIONSHIPS	33590
Verbindung zu SchemaAgent Server herstellen...	IDC_SCHEMA_SCHEMAAGENT_SERVER_CONNECT	33582
Verbindung mit SchemaAgent Server trennen	IDC_SCHEMA_SCHEMAAGENT_SERVER_DISCONNECT	33583
Nur Datei	IDC_SCHEMAAGENT_SHOW_FILE_ONLY	33504
Datei und alle direkt referenzierten Schema-Dateien	IDC_SCHEMAAGENT_SHOW_WITH_DIRECTLY_REFERENCED_SCHEMAS	33608
Datei und alle direkt referenzierenden Schema-Dateien	IDC_SCHEMAAGENT_SHOW_WITH_DIRECTLY_REFERENCING_SCHEMAS	33602
Datei und alle direkt verwandten Schema-Dateien	IDC_SCHEMAAGENT_SHOW_WITH_DIRECTLY_RELATED_SCHEMAS	33613
SchemaAgent-Validierung...	IDC_SCHEMA_EXTVALID_MENU	33539
Schemamodul erstellen...	IDC_SCHEMA_CREATE_SUBSET	33650
Schema nivellieren...	IDC_SCHEMA_FLATTEN	33651

29.4.6.7 Menü "XSL/XQuery"

Das Menü "XSL/XQuery" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
XSL-Transformation	IDC_TRANSFORM_XSL	33006
XSL-Geschwindigkeitsoptimierung	IDC_TRANSFORM_XSLPBO	34306
XSL:FO-Transformation	IDC_TRANSFORM_XSLFO	33007
XSL-Parameter / XQuery-Variablen...	IDC_TRANSFORM_XSL_PARAMS	33008
XQuery/Update-Ausführung	IDC_TRANSFORM_XQUERY	34170
Rückwärts-Mapping aktivieren	IDC_ENABLE_BACKMAPPING	34364
XSLT / XQuery Profilerstellung aktivieren....	IDC_PROFILING_OPTIONS	34105
XSL zuweisen...	IDC_ASSIGN_XSL	33001
XSL-FO zuweisen...	IDC_ASSIGN_XSLFO	33002
XML-Beispieldatei zuweisen...	IDC_ASSIGN_SAMPLE_XML	33000
Gehe zu XSL	IDC_GOTO_XSL	33004
Debugger starten	ID_PROCESS_XSL	34212
Debugger anhalten	ID_XSLT_DEBUGGER_STOP	33017
Debugger neu starten	ID_XSLT_DEBUGGER_RESTART	33013
Debuggersitzung beenden	ID_XSLT_DEBUGGER_END_SESSION	33011
Einsteigen	ID_XSLT_DEBUGGER_STEP	33014
Aussteigen	ID_XSLT_DEBUGGER_STEP_OUT	33015
Überspringen	ID_XSLT_DEBUGGER_STEP_OVER	33016
Aktuellen Execution Node einblenden	ID_XSLT_DEBUGGER_GO_TO_CURRENT_EXECUTION_NODES	33012
Breakpoint einfügen/löschen	IDC_TOGGLE_BREAKPOINT	34246
Tracepoint einfügen/löschen	IDC_TOGGLE_TRACEPOINT	34248
Breakpoint aktivieren/deaktivieren	IDC_ENABLE_BREAKPOINT	34245
Tracepoint aktivieren/deaktivieren	IDC_ENABLE_TRACEPOINT	34247
Breakpoints/Tracepoints...	ID_XSLTDEBUGGER_BREAKPOINTS	33009

© 2016-2022 Altova GmbH

Menübefehl	Befehlsname	ID
Call Stack	ID_XSL_DEBUGWINDOWS_CALLSTACK	34238
XPath-Watch	ID_XSL_DEBUGWINDOWS_WATCH	34244
Kontext	ID_XSL_DEBUGWINDOWS_CONTEXT	34239
Variablen	ID_XSL_DEBUGWINDOWS_VARIABLE	34243
Meldungen	ID_XSL_DEBUGWINDOWS_MESSAGES	34240
Templates	ID_XSL_DEBUGWINDOWS_TEMPLATES	34241
Info	ID_XSLXQUERY_DEBUGWINDOWS_INFO	34237
Ablaufverfolgung	ID_XSL_DEBUGWINDOWS_TRACES	34242
Debugeinstellungen...	ID_XSLTDEBUGGER_SETTINGS	33010

29.4.6.8 Menü "Authentic"

Das Menü "Authentic" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Neues Dokument...	IDC_AUTHENTIC_NEW_FILE	34036
Datenbank bearbeiten...	IDC_AUTHENTIC_EDIT_DB	34035
StyleVision Stylesheet zuweisen...	IDC_ASSIGN_SPS	34034
StyleVision Stylesheet bearbeiten...	IDC_EDIT_SPS	34060
Neue Zeile mit XML-Daten für Bearbeitung auswählen...	IDC_CHANGE_WORKING_DB_XML_CELL	32861
XML-Signatur...	IDC_AUTHENTICGUI_XMLSIGNATURE	32862
XML Entities definieren...	IDC_DEFINE_ENTITIES	32805
Markup-Symbole ausblenden	IDC_MARKUP_HIDE	32855
Kleine Markup-Symbole einblenden	IDC_MARKUP_SMALL	32858
Große Markup-Symbole einblenden	IDC_MARKUP_LARGE	32856
Gemischte Markup-Symbole einblenden	IDC_MARKUP_MIXED	32857
Fett ein/aus	IDC_AUTHENTICGUI_RICHEDIT_TOGGLEBOLD	32813
Kursiv ein/aus	IDC_AUTHENTICGUI_RICHEDIT_TOGGLEITALIC	32814

Menübefehl	Befehlsname	ID
Unterstrichen ein/aus	IDC_AUTHENTICGUI_RICHEDIT_TOGGLEUNDERLINE	32815
Durchgestrichen ein/aus	IDC_AUTHENTICGUI_RICHEDIT_TOGGLESTRIKETHROUGH	32816
Vordergrundfarbe	IDC_AUTHENTICGUI_RICHEDIT_COLOR_FOREGROUND	32824
Hintergrundfarbe	IDC_AUTHENTICGUI_RICHEDIT_COLOR_BACKGROUND	32830
Linksbündig	IDC_AUTHENTICGUI_RICHEDIT_ALIGN_LEFT	32818
Zentriert	IDC_AUTHENTICGUI_RICHEDIT_ALIGN_CENTER	32819
Rechtsbündig	IDC_AUTHENTICGUI_RICHEDIT_ALIGN_RIGHT	32820
Zeile anhängen	IDC_ROW_APPEND	32806
Zeile einfügen	IDC_ROW_INSERT	32809
Zeile kopieren	IDC_ROW_DUPLICATE	32808
Zeile nach oben	IDC_ROW_MOVE_UP	32811
Zeile nach unten	IDC_ROW_MOVE_DOWN	32810
Zeile löschen	IDC_ROW_DELETE	32807
HTML-Dokument generieren	IDC_PXF_GENERATE_HTML	34283
RTF-Dokument generieren	IDC_PXF_GENERATE_RTF	34284
PDF-Dokument generieren	IDC_PXF_GENERATE_PDF	34285
Word 2007+-Dokument generieren	IDC_PXF_GENERATE_DOCX	34286
Textdokument generieren	DC_PXF_GENERATE_TEXT	
Vertrauenswürdige Pfade...	IDC_TRUSTED_LOCATIONS	34288

29.4.6.9 Menü "DB"

Das Menü "DB" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Datenbank abfragen	IDC_QUERYDATABASE	34012

© 2016-2022 Altova GmbH

Menübefehl	Befehlsname	ID
XML-Schemas verwalten...	IDC_DB_MANAGESCHEMAS	34014
XML-Schema zuweisen	IDC_DB_CHOOSEVALIDATIONSCHEMA	34016
XML-Schemas verwalten...	IDC_DB_MANAGESCHEMAS	34014
XML-Schemas verwalten...	IDC_DB_MANAGESCHEMAS	34014
Oracle XML-Dokumente durchsuchen...	ID_CONVERT_ORACLEXMLDB_BROWSE	34205

29.4.6.10 Menü "Konvertieren"

Das Menü "Konvertieren" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Textdatei importieren...	IDC_IMPORT_TEXT	34082
Datenbankdaten importieren...	IDC_IMPORT_DATABASE	34080
MS Word-Dokument importieren...	IDC_IMPORT_WORD	34083
XML-Schema von DB-Struktur erzeugen	IDC_CREATE_DB_SCHEMA	34054
DB-Import auf Basis von XML-Schema	IDC_IMPORT_DB_SCHEMA	34081
DB-Struktur von XML-Schema erzeugen	IDC_CREATE_DB_BASED_ON_SCHEMA	34053
Export in Textdateien...	IDC_EXPORT_TEXTFILE	34064
Export in Datenbank...	IDC_EXPORT_DB	34003
XML-Instanz von/in JSON konvertieren...	IDC_JSON_CONVERT_TOFROM_XML	34135
XML-Schema von/in JSON-Schema konvertieren...	IDC_JSON_CONVERT_TOFROM_XSD	34350

29.4.6.11 Menü "Ansicht"

Das Menü "Ansicht" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Textansicht	IDC_VIEW_TEXT	34180
Grid-Ansicht	IDC_VIEW_GRID	34178
Schema-Design-Ansicht	IDC_VIEW_SCHEMA	34179

Menübefehl	Befehlsname	ID
WSDL-Design-Ansicht	IDC_VIEW_WSDL	34117
XBRL-Taxonomie-Ansicht	IDC_VIEW_XBRL	34118
Authentic-Ansicht	IDC_VIEW_CONTENT	34177
Browser-Ansicht	IDC_VIEW_BROWSER	34176
Erweitern +	IDC_SEL_EXPAND	34152
Reduzieren -	IDC_SEL_COLLAPSE	34151
Alles erweitern	IDC_SEL_EXPAND_ALL	33463
Unselektierte reduzieren	IDC_COLLAPSE_UNSELECTED	33428
Optimale Breite	IDC_OPTIMAL_WIDTHS	34099
Zeilenumbruch	IDC_WORD_WRAP	34181
Gehe zu Zeile/Zeichen	IDC_GOTO_LINE	34073
Gehe zu Datei	IDC_GOTO_FILE	33448
Einstellungen für Textansicht	IDC_TEXTVIEW_SETTINGS	34119

29.4.6.12 Menü "Browser"

Das Menü "Browser" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Zurück	IDC_STEP_BACK	32958
Vorwärts	IDC_STEP_FORWARD	32957
Abbrechen	IDC_BROWSER_STOP	34047
Aktualisieren	IDC_BROWSER_REFRESH	34046
Maximale Größe	IDC_BROWSER_FONT_LARGEST	34041
Größer	IDC_BROWSER_FONT_LARGE	34040
Mittel	IDC_BROWSER_FONT_MEDIUM	34042
Kleiner	IDC_BROWSER_FONT_SMALL	34043
Minimale Größe	IDC_BROWSER_FONT_SMALLEST	34044

© 2016-2022 Altova GmbH

29.4.6.13 Menü "WSDL"

Das Menü "WSDL" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Message einfügen	ID_WSDL_MESSAGES_ADDNEWMESSAGE	33715
Message löschen	ID_WSDL_MESSAGES_DELETESELECTEDMESSAGE	33717
Message Part (Parameter) hinzufügen	ID_WSDL_MESSAGES_ADDMESSAGEPART	33714
Message Part (Parameter) löschen	ID_WSDL_MESSAGES_DELETEMESSAGEPART	33716
request-response	IDC_WSDL_OPERATION_APPENDREQUESTRESPONSE	33734
solicit-response	IDC_WSDL_OPERATION_APPENDSOLICITRESPONSE	33737
one-way	IDC_WSDL_OPERATION_APPENDONEWAY	33735
notification	IDC_WSDL_OPERATION_APPENDNOTIFICATION	33736
Leere Operation	ID_WSDL_OPERATIONS_APPENDAOPERATIONTOTHISPORTTYPE	33722
Operation löschen	ID_WSDL_OPERATIONS_DELETEOPERATION	33724
Input-Element hinzufügen	ID_WSDL_OPERATIONS_ADDINPUTFUNCTION	33719
Output-Element hinzufügen	ID_WSDL_OPERATIONS_ADDOUTPUTFUNCTION	33721
Fault-Element hinzufügen	ID_WSDL_OPERATIONS_ADDFAULTFUNCTION	33718
Input/Output/Fault-Element löschen	ID_WSDL_OPERATIONS_DELETEINPUTOUTPUTFUNCTION	33723
Neue Message zu Input/Output/Fault-Element hinzufügen	ID_WSDL_OPERATIONS_ADDNEWMESSAGETOTHISELEMENT	33720
PortType einfügen	ID_WSDL_PORTTYPE_INSERTAPORTTYPE	33727
PortType löschen	ID_WSDL_PORTTYPE_DELETETHISPORTTYPE	33726
Binding einfügen	ID_WSDL_BINDING_NEWBINDING	33713

Menübefehl	Befehlsname	ID
Binding löschen	ID_WSDL_BINDING_DELETEBINDING	33711
soap:body	ID_WSDL_BINDING_APPENDEXTENSIBILITY_SOAPBODY	33706
soap:header	ID_WSDL_BINDING_APPENDEXTENSIBILITY_SOAPHEADER	33708
soap:headerfault	ID_WSDL_BINDING_APPENDEXTENSIBILITY_SOAPHEADERFAULT	33709
soap:fault	ID_WSDL_BINDING_APPENDEXTENSIBILITY_SOAPFAULT	33707
mime:content	ID_WSDL_BINDING_APPENDEXTENSIBILITY_MIMECONTENT	33702
mime:multipartrelated	ID_WSDL_BINDING_APPENDEXTENSIBILITY_MIMEMULTIPARTRELATED	33704
mime:part	ID_WSDL_BINDING_APPENDEXTENSIBILITY_MIMEPART	33705
mime:mimeXml	ID_WSDL_BINDING_APPENDEXTENSIBILITY_MIMEMIMEXML	33703
http:urlencoded	ID_WSDL_BINDING_APPENDEXTENSIBILITY_HTTPURLENCODED	33700
http:urlreplacement	ID_WSDL_BINDING_APPENDEXTENSIBILITY_HTTPURLREPLACEMENT	33701
Extensibility-Element löschen	ID_WSDL_BINDING_DELETEEXTENSIBILITY	33712
Service einfügen	ID_WSDL_SERVICE_INSERTSERVICE	33731
Service löschen	ID_WSDL_SERVICE_DELETETHISSERVICE	33729
Port einfügen	ID_WSDL_SERVICE_INSERTNEWPORT	33730
Port löschen	ID_WSDL_SERVICE_DELETETHISPORT	33728
Neues Interface hinzufügen	IDC_WSDL20_ADDINTERFACE	33794
Interface löschen	IDC_WSDL20_DELETEINTERFACE	33795
Neues Fault-Element hinzufügen	IDC_WSDL20_ADDINTERFACEFAULT	33796
Fault-Element löschen	IDC_WSDL20_DELETEINTERFACEFAULT	33808
In-only	IDC_WSDL20_ADDINTERFACEOPERATION_INONLY	33797

© 2016-2022 Altova GmbH

Menübefehl	Befehlsname	ID
Robust-in-only	IDC_WSDL20_ADDINTERFACEOPERATION_ROBUSTINONLY	33798
In-out	IDC_WSDL20_ADDINTERFACEOPERATION_INOUT	33801
In-opt-out	IDC_WSDL20_ADDINTERFACEOPERATION_INOPTOUT	33802
Out-in	IDC_WSDL20_ADDINTERFACEOPERATION_OUTIN	33803
Out-opt-in	IDC_WSDL20_ADDINTERFACEOPERATION_OUTOPTIN	33804
Out-only	IDC_WSDL20_ADDINTERFACEOPERATION_OUTONLY	33800
Robust-out-only	IDC_WSDL20_ADDINTERFACEOPERATION_ROBUSTOUTONLY	33799
Leere Operation	IDC_WSDL20_ADDINTERFACEOPERATION_EMPTY	33805
Operation löschen	IDC_WSDL20_DELETEINTERFACEOPERATION	33809
Neues Binding hinzufügen	IDC_WSDL20_ADDBINDING	33820
Binding löschen	IDC_WSDL20_DELETEBINDING	33821
Neues Fault-Element hinzufügen	IDC_WSDL20_ADDBINDINGFAULT	33822
Fault-Element löschen	IDC_WSDL20_DELETEBINDINGFAULT	33826
Neue Operation hinzufügen	IDC_WSDL20_ADDBINDINGOPERATION	33823
Operation löschen	IDC_WSDL20_DELETEBINDINGOPERATION	33827
Neuen Service hinzufügen	IDC_WSDL20_ADDSERVICE	33839
Service löschen	IDC_WSDL20_DELETESERVICE	33840
Neuen Endpoint hinzufügen	IDC_WSDL20_ADDENDPOINT	33841
Endpoint löschen	IDC_WSDL20_DELETEENDPOINT	33842
Neues Schema	ID_WSDL_TYPES_NEWSCHEMA	33733
Schema einbetten	ID_WSDL_TYPES_EMBEDSCHEMA	39456
Schema(s) extrahieren	ID_WSDL_TYPES_EXTRACTSCHEMAS	39459
Schema(s) in Schema-Ansicht bearbeiten	ID_WSDL_TYPES_EDITTHISSCHEMA	33732

Menübefehl	Befehlsname	ID
Diagramm speichern...	IDC_WSDL_SAVE_DIAGRAM	39451
Dokumentation generieren...	ID_WSDL_GENERATEDOCUMENTATION	39452
WSDL-Dokument neu parsen	IDC_WSDL_REPARSE	33774
In WSDL 2.0 konvertieren	IDC_WSDL_CONVERT_TO_WSDL20	39453
WSDL-Programmcode mit MapForce generieren...	IDC_WSDL_GENERATE_CODE_MAPFORCE	34122

29.4.6.14 Menü "SOAP"

Das Menü "SOAP" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Neuen SOAP Request erzeugen	ID_SOAP_GENERATESOAPMESSAGE	34224
Request an Server senden	ID_SOAP_SENDREQUESTTOSERVER	34225
SOAP Request-Einstellungen	ID_SOAP_SOAPREQUESTSETTINGS	34227
Soap Debugger-Sitzung	ID_SOAP_SOAPDEBUGGER	34226
Start	ID_SOAPDEBUGGER_BUTTONPLAY	34221
Einzelschritt	ID_SOAPDEBUGGER_SINGLESTEP	34222
Bei nächstem Request anhalten	ID_SOAPDEBUGGER_BREAKONNEXTREQUEST	34219
Bei nächstem Response anhalten	ID_SOAPDEBUGGER_BREAKONNEXTRESPONSE	34220
Proxyserver anhalten	ID_SOAPDEBUGGER_STOPSERVER	34223
SOAP Debugger-Optionen	ID_SOAPDEBUGGEROPTIONS	34218

29.4.6.15 Menü "XBRL"

Das Menü "XBRL" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Arcroles...	IDC_XMLSPYXBRLEDITOR_ARCROLES	34114
Linkroles...	IDC_XMLSPYXBRLEDITOR_LINKROLES	34115

© 2016-2022 Altova GmbH

Menübefehl	Befehlsname	ID
Namespace-Präfixe...	IDC_XMLSPYXBRLEDITOR_NAMESPACES	34116
Target Namespace definieren...	IDC_XMLSPYXBRLEDITOR_SET_TARGETNAMESPACE	34039
Parameterwerte...	IDC_ICXBRLEDITOR_PARAMETER_VALUES	38913
Importieren/Referenzieren...	IDC_XMLSPYXBRLEDITOR_IMPORT_REFERENCE	34137
Komponente nach ID suchen...	IDC_ICXBRLEDITOR_FIND_COMPONENT_BY_ID	38893
Dokumentation generieren...	IDC_XMLSPYXBRLEDITOR_GENERATEDOCUMENTATION	34125
Ansichtseinstellungen...	IDC_XMLSPYXBRLEDITOR_VIEWSETTINGS	34113
Mit MapForce XBRL anhand von DB, Excel, CSV generieren...	IDC_XBRL_GENERATE_WITH_MAPFORCE	34045
XBRL mit StyleVision als HTML/PDF/Word-Dokument darstellen...	IDC_XBRL_PRESENT_WITH_STYLEVISION	34121
Formula ausführen ...	IDC_XBRL_EXECUTE_FORMULA	34305
Formula am Server ausführen (Hochleistungsausführung)...	IDC_XBRL_EXECUTE_FORMULA_RAPTOR	34352
Tabelle generieren ...	IDC_XBRL_GENERATE_TABLE	34304
Tabelle am Server generieren (Hochleistungsausführung)...	IDC_XBRL_GENERATE_TABLE_RAPTOR	34353
Inline XBRL transformieren	IDC_IXBRL_TRANSFORM	34354

29.4.6.16 Menü "Extras"

Das Menü "Extras" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Rechtschreibung...	IDC_SPELL_CHECK	34154
Rechtschreiboptionen...	IDC_SPELL_OPTIONS	34155
Skript-Editor...	ID_SCRIPTFORMEDITOR_EDIT_PROJECT	39666
Keines	ID_SCRIPTFORMEDITOR_EXECUTE_MACRO_MENU_UPPDATE	39600

Menübefehl	Befehlsname	ID
Geöffnete Datei vergleichen mit...	ID_XMLDIFF_CHOOSE_FILES	34235
Verzeichnisse vergleichen...	ID_XMLDIFF_CHOOSE_DIRECTORIES	34234
Vergleichseinstellungen...	ID_XMLDIFF_SETTINGS	34236
	IDC_TOOLS_ENTRY	34292
Globale Ressourcen	IDC_GLOBALRESOURCES	37401
	IDC_GLOBALRESOURCES_SUBMENUENTRY1	37408
Raptor-Server verwalten ...	IDC_VALIDATE_RAPTOR_MANAGER	34311
Keines	IDC_VALIDATE_RAPTOR_NOCFG	34326
Anpassen...	IDC_APP_TOOLS_CUSTOMIZE	32959
Optionen...	IDC_SETTINGS	34133
	ID_SCRIPTING_MACROITEMS	34249

29.4.6.17 Menü "Fenster"

Das Menü "Fenster" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Überlappend	ID_WINDOW_CASCADE	57650
Horizontal anordnen	ID_WINDOW_TILE_HORZ	57651
Vertikal anordnen	ID_WINDOW_TILE_VERT	57652
Projekt-Fenster	IDC_PROJECT_WINDOW	34128
Info-Fenster	IDC_INFO_WINDOW	34085
Eingabehilfen	IDC_ENTRY_HELPERS	34062
Ausgabefenster	IDC_OUTPUT_DIALOGBARS	34004
Projekt und Eingabehilfen	IDC_PROJECT_ENTRYHELPERS	34006
Alle ein/aus	IDC_ALL_BARS	34031

© 2016-2022 Altova GmbH

29.4.6.18 Menü "Hilfe"

Das Menü "Hilfe" enthält die folgenden Befehle:

Menübefehl	Befehlsname	ID
Inhaltsverzeichnis...	IDC_HELP_CONTENTS	32966
Index...	IDC_HELP_INDEX	32967
Suchen...	IDC_HELP_SEARCH	32969
Tastaturzuordnungen...	IDC_HELP_KEYMAPDLG	32968
Software-Aktivierung...	IDC_ACTIVATION	32970
Bestellformular...	IDC_OPEN_ORDER_PAGE	32971
Registrieren...	IDC_REGISTRATION	32972
Auf Updates überprüfen...	IDC_CHECK_FOR_UPDATES	32973
XMLSpy-Produktvergleich...	IDC_PRODUCT_COMPARISON	32955
Support Center...	IDC_OPEN_SUPPORT_PAGE	32961
Fragen und Antworten im Web...	IDC_OPEN_FAQ_PAGE	32962
Komponenten und Gratistools downloaden...	IDC_OPEN_COMPONENTS_PAGE	32963
XMLSpy im Internet..	IDC_OPEN_HOME_PAGE	32964
XMLSpy Training...	IDC_OPEN_TRAINING_PAGE	32965
Über XMLSpy...	ID_APP_ABOUT	57664

29.4.7 Objektreferenz

Objekte:
XMLSpyCommand 2114
XMLSpyCommands 2116
XMLSpyControl 2117
XMLSpyControlDocument 2125
XMLSpyControlPlaceHolder 2132

Um Zugriff auf die XMLSpy Standardfunktionalitäten zu erhalten, können auch Objekte des **XMLSpy Automation Interface** aufgerufen werden. Nähere Informationen dazu finden Sie unter XMLSpyControl.Application 2118, XMLSpyControlDocument.Document 2126 und XMLSpyControlPlaceHolder.Project 2133.

29.4.7.1 XMLSpyCommand

Eigenschaften:
ID 2115
Label 2115
Name 2115
IsSeparator 2115
ToolTip 2116
StatusText 2115
Accelerator 2114
SubCommands 2116

Beschreibung:
Jedes Command Objekt kann einer von drei möglichen Typen sein: ein ausführbarer Befehl, ein Befehlscontainer (z.B. ein Menü, ein Untermenü oder eine Symbolleiste) oder ein Menütrennzeichen. Um herauszufinden, welche Art von Informationen im aktuellen `Command`-Objekt gespeichert sind, fragen Sie seine Eigenschaften `ID`, `IsSeparator` und `SubCommands` folgendermaßen ab.

Das Befehlsobjekt ist...	wenn...
ein ausführbarer Befehl	• `ID` größer als Null ist • `IsSeparator` "false" ist • `SubCommands` leer ist
ein Befehlscontainer	• `ID` Null ist • `IsSeparator` "false" ist • `SubCommands` eine Sammlung von `Command` Objekten enthält
ein Trennzeichen	• `ID` Null ist • `IsSeparator` "true" ist

29.4.7.1.1 Accelerator

Eigenschaft: `Accelerator` als string

Beschreibung:
Gibt die für den Befehl definierte Zugriffstaste zurück. Wenn dem Befehl keine Zugriffstaste zugewiesen wurde, gibt diese Eigenschaft den leeren String zurück. Die String-Darstellung der Zugriffstaste hat das folgenden Format:

```
[ALT+][CTRL+][SHIFT+]key
```

`key` wird mittels der Windows Plattform SDK-Funktion `GetKeyNameText` konvertiert.

© 2016-2022 Altova GmbH

29.4.7.1.2 ID

Eigenschaft: `ID` als long

Beschreibung:
Mit dieser Eigenschaft wird der eindeutige Identifier des Befehls abgerufen. Die ID eines Befehls wird benötigt, um den Befehl (mittels `Exec`[2121]) auszführen oder seinen Status (mittels `QueryStatus`[2122]) abzurufen. Wenn der Befehl ein Container für andere Befehle (z.B. für ein Menü der obersten Ebene) oder ein Trennzeichen ist, ist die ID 0.

29.4.7.1.3 IsSeparator

Eigenschaft: `IsSeparator` als boolean

Beschreibung:
Die Eigenschaft gibt `true` zurück, wenn das Befehlsobjekt ein Menütrennzeichen ist; andernfalls wird `false` zurückgegebven. Siehe auch Command[2114].

29.4.7.1.4 Label

Eigenschaft: `Label` als string

Beschreibung:
Diese Eigenschaft ruft den Text des Befehls, wie er auf der grafischen Benutzeroberfläche von XMLSpy angezeigt wird, ab. Wenn der Befehl ein Trennzeichen ist, ist "Label" ein leerer String. Diese Eigenschaft kann für einige Symbolleistenbefehle, zu denen es keinen GUI-Text gibt, auch einen leeren String zurückgeben.

29.4.7.1.5 Name

Eigenschaft: `Name` als string

Beschreibung:
Diese Eigenschaft ruft den eindeutigen Namen des Befehls ab. Anhand dieses Werts kann die Symboldatei das Befehls, falls verfügbar, abgerufen werden. Die verfügbaren Symboldateien befinden sich im Ordner **<ApplicationFolder>\Examples\ActiveX\Images** Ihrer XMLSpy-Installation.

29.4.7.1.6 StatusText

Eigenschaft: `Label` als string

Beschreibung:
Der Statustext ist der Text, der in der Statusleiste von XMLSpy angezeigt wird, wenn der Befehl ausgewählt wird. Dies gilt nur für Befehlsobjekte, die keine Trennzeichen oder Container von anderen Befehlen sind. Andernfalls ist die Eigenschaft ein leerer String.

29.4.7.1.7 SubCommands

Eigenschaft: `SubCommands` als `Commands`[2116]

Beschreibung:
Die Eigenschaft `SubCommands` ruft die Sammlung von `Command`[2114]-Objekten, die dem aktuellen Befehl untergeordnet sind, auf. Die Eigenschaft gilt nur für Befehle, die Container für andere Befehle sind (Menüs, Untermenüs oder Symbolleisten). Bei solchen Container-Befehlen ist die `ID` auf `0` gesetzt und die Eigenschaft `IsSeparator` auf `false`.

29.4.7.1.8 ToolTip

Eigenschaft: `ToolTip` als string

Beschreibung:
Diese Eigenschaft ruft den Text, der als Tooltipp zu den einzelnen Befehlen angezeigt wird, ab. Wenn der Befehl keinen Tooltipp-Text hat, so gibt die Eigenschaft einen leeren String zurück.

29.4.7.2 XMLSpyCommands

Eigenschaften:
`Count`[2116]
`Item`[2116]

Beschreibung:
Sammlung von `Command`[2114]-Objekten für den Zugriff auf Befehlsbezeichnungen und IDs des XMLSpyControl. Diese Befehle können mit der `Exec`[2121] Methode ausgeführt werden und ihr Status kann mittels `QueryStatus`[2122] abgefragt werden.

29.4.7.2.1 Count

Eigenschaft: `Count` als long

Beschreibung:
Anzahl der `Command`[2114] Objekte auf dieser Ebene der Collection

29.4.7.2.2 Item

Eigenschaft: `Item` (`n` als long) als `Command`[2114]

Beschreibung:
Ruft den Befehl mit den Index `n` in dieser Sammlung auf. Der Index basiert auf 1.

© 2016-2022 Altova GmbH

29.4.7.3 XMLSpyControl

Eigenschaften:
IntegrationLevel 2119
Appearance 2118
Application 2118
BorderStyle 2118
CommandsList 2118
EnableUserPrompts 2119
MainMenu 2119
Toolbars 2120

Methoden:
Open 2121
Exec 2121
QueryStatus 2122

Events:
OnUpdateCmdUI 2124
OnOpenedOrFocused 2124
OnCloseEditingWindow 2122
OnFileChangedAlert 2123
OnContextChanged 2123
OnDocumentOpened 2123
OnValidationWindowUpdated 2125

Dieses Objekt ist ein vollständiges ActiveX Control und sollte nur sichtbar sein, wenn die XMLSpy Bibliothek im Applikationsebenenmodus verwendet wird.

```
CLSID: a258bba2-3835-4c16-8590-72b44f52c471
ProgID: Altova.XMLSpyControl
```

29.4.7.3.1 Eigenschaften

Es sind die folgenden Eigenschaften definiert:

IntegrationLevel 2119
EnableUserPrompts 2119
Appearance 2118
BorderStyle 2118

Befehlsbezogene Eigenschaften:
CommandsList 2118
MainMenu 2119
Toolbars 2120

Zugriff auf die XMLSpyAPI:
Application 2118

29.4.7.3.1.1 Appearance

Eigenschaft: `Appearance` als short

Dispatch Id: *-520*

Beschreibung:
Bei einem Wert, der nicht gleich `0` ist, wird ein Client-Rand rund um das Control angezeigt. Der Standardwert ist `0`.

29.4.7.3.1.2 Application

Eigenschaft: `Application` als Application

Dispatch Id: *1*

Beschreibung:
Mit der Eigenschaft `Application` erhalten Sie Zugriff auf das `Application` Objekt der vollständigen XMLSpy Automation Server API. Die Eigenschaft ist schreibgeschützt.

29.4.7.3.1.3 BorderStyle

Eigenschaft: `BorderStyle` als short

Dispatch Id: *-504*

Beschreibung:
Bei einem Wert `1` wird das Control mit einer dünnen Umrandung angezeigt. Der Standardwert ist `0`.

29.4.7.3.1.4 CommandsList

Eigenschaft: `CommandList` als `Commands`[2116] (schreibgeschützt)

Dispatch Id: *1004*

Beschreibung:
Diese Eigenschaft gibt eine flache Liste aller Befehle zurück, die mit XMLSpyControl verfügbar sind. Um Befehle, geordnet nach Menüstruktur, abzurufen, verwenden Sie `MainMenu`[2119]. Um Symbolleistenbefehle abzurufen, verwenden Sie `Toolbars`[2120].

© 2016-2022 Altova GmbH

```
public void GetAllXmlSpyCommands()
{
    // Get all commands from the XMLSpy ActiveX control assigned to the current form
    XMLSpyControlLib.XMLSpyCommands commands = this.axXMLSpyControl1.CommandList;
    // Iterate through all commands
    for (int i = 0; i < commands.Count; i++)
    {
      // Get each command by index and output it to the console
      XMLSpyControlLib.XMLSpyCommand cmd = axXMLSpyControl1.CommandList[i];
      Console.WriteLine("{0} {1} {2}", cmd.ID, cmd.Name, cmd.Label.Replace("&", ""));
    }
}
```

C#-Beispiel

29.4.7.3.1.5 EnableUserPrompts

Eigenschaft: `EnableUserPrompts` als boolean

Dispatch Id: *1006*

Beschreibung:
Wenn Sie diese Eigenschaft auf *false* setzen, wird die Eingabeaufforderung im Control deaktiviert. Der Standardwert ist *true*.

29.4.7.3.1.6 IntegrationLevel

Eigenschaft: `IntegrationLevel` als `ICActiveXIntegrationLevel` 2135

Dispatch Id: *1000*

Beschreibung:
Die Eigenschaft `IntegrationLevel` bestimmt den Operationsmodus des Control. Nähere Informationen dazu siehe auch Integration auf Applikationsebene 2076 und Integration auf Dokumentebene 2078.

Anmerkung: Diese Eigenschaft muss unbedingt sofort nach Erstellung des `XMLSpyControl` Objekts definiert werden.

29.4.7.3.1.7 MainMenu

Eigenschaft: `MainMenu` als Command 2114 (schreibgeschützt)

Dispatch Id: *1003*

Beschreibung:

Diese Eigenschaft enthält Informationen über die Struktur und die Befehle im XMLSpyControl-Hauptmenü als `Command`-Objekt. Das `Command`-Objekt enthält alle verfügbaren Untermenüs von XMLSpy (z.B. Datei, Bearbeiten, Ansicht, usw.). Verwenden Sie die Eigenschaft `SubCommands` der Eigenschaft `MainMenu`, um die Untermenüobjekte abzurufen. Jedes Untermenü ist ebenfalls ein `Command`-Objekt. Sie können bei jedem Untermenü weiter durch dessen `SubCommands`-Eigenschaft iterieren, um die jeweiligen Child-Befehle und Trennzeichen dieser Untermenüs abzurufen (Auf diese Art können Sie z.B. das Applikationsmenü programmatisch erstellen). Beachten Sie, dass einige Menübefehle als Container ("Parents") für andere Menübefehle dienen. In diesen Fällen haben diese ebenfalls eine Eigenschaft `SubCommands`. Um die Struktur aller Menübefehle programmatisch abzurufen, müssen Sie wahrscheinlich eine rekursive Funktion erstellen.

```
public void GetXmlSpyMenus()
{
    // Get the main menu from the XMLSpy ActiveX control assigned to the current form
    XMLSpyControlLib.XMLSpyCommand mainMenu = this.axXMLSpyControl1.MainMenu;

    // Loop through entries of the main menu (e.g. File, Edit, etc.)
    for (int i = 0; i < mainMenu.SubCommands.Count; i++)
    {
      XMLSpyControlLib.XMLSpyCommand menu = mainMenu.SubCommands[i];
      Console.WriteLine("{0} menu has {1} children items (including separators)",
menu.Label.Replace("&", ""), menu.SubCommands.Count);
    }
}
```

C# example

29.4.7.3.1.8 Toolbars

Eigenschaft: `Toolbars` als Commands 2116 (schreibgeschützt)

Dispatch Id: *1005*

Beschreibung:
Diese Eigenschaft enthält Informationen über die Struktur von XMLSpyControl-Symbolleisten, als `Command`-Objekt. Das `Command`-Objekt enthält alle verfügbaren Symbolleisten von XMLSpy. Verwenden Sie die Eigenschaft `SubCommands` der Eigenschaft `Toolbars`, um die Symbolleisten abzurufen. Jede Symbolleiste ist ebenfalls ein `Command`-Objekt. Sie können bei jeder Symbolleiste weiter durch deren `SubCommands`-Eigenschaft iterieren, um deren Befehle abzurufen (Auf diese Art können Sie z.B. die Symbolleisten der Applikation programmatisch erstellen).

```
public void GetXmlSpyToolbars()
{
    // Get the application toolbars from the StyleVision ActiveX control assigned to the
current form
    XMLSpyControlLib.XMLSpyCommands toolbars = this.axXMLSpyControl1.Toolbars;

    // Iterate through all toolbars
    for (int i = 0; i < toolbars.Count; i++)
    {
      XMLSpyControlLib.XMLSpyCommand toolbar = toolbars[i];
```


© 2016-2022 Altova GmbH

```
      Console.WriteLine();
      Console.WriteLine("The toolbar \"{0}\" has the following commands:",
toolbar.Label);

      // Iterate through all commands of this toolbar
      for (int j = 0; j < toolbar.SubCommands.Count; j++)
      {
         XMLSpyControlLib.XMLSpyCommand cmd = toolbar.SubCommands[j];
         // Output only command objects that are not separators
         if (!cmd.IsSeparator)
         {
            Console.WriteLine("{0}, {1}, {2}", cmd.ID, cmd.Name, cmd.Label.Replace("&",
""));
         }
      }
   }
}
```

C# example

29.4.7.3.2 Methoden

Es sind die folgenden Methoden definiert:

Open 2121
Exec 2121
QueryStatus 2122

29.4.7.3.2.1 Exec

Methode: `Exec` (`nCmdID` als long) als boolean

Dispatch Id: *6*

Beschreibung:
`Exec` ruft den XMLSpy Befehl mit der ID `nCmdID` auf. Wenn der Befehl ausgeführt werden kann, gibt die Methode `true` zurück. Eine Liste aller verfügbaren Befehle finden Sie unter `CommandsList` 2118. Um den Status eines Befehls abzurufen, verwenden Sie `QueryStatus` 2122.

29.4.7.3.2.2 Open

Methode: `Open` (`strFilePath` als string) als boolean

Dispatch Id: *5*

Beschreibung:
Das Ergebnis der Methode ist von der Erweiterung abhängig, die im Argument `strFilePath` übergeben wird. Ist die Dateierweiterung `.sps`, wird ein neues Dokument geöffnet. Ist die Dateierweiterung `.svp`, wird das

entsprechende Projekt geöffnet. Wird eine andere Dateierweiterung in die Methode übergeben, versucht das Control, die Datei als neue Komponente in das aktive Dokument zu laden.

Verwenden Sie diese Methode nicht, um Dokumente oder Projekte bei Verwendung des Control auf Dokumentebene zu laden. Verwenden Sie statt dessen `XMLSpyControlDocument.Open`[2128] und `XMLSpyControlPlaceHolder.OpenProject`[2133].

29.4.7.3.2.3 QueryStatus

Methode: `QueryStatus` (`nCmdID` als long) als long

Dispatch Id: *7*

Beschreibung:
`QueryStatus` gibt den Status "enabled/disabled" und "checked/unchecked" des von `nCmdID`. definierten Befehls zurück. Der Status wird als Bitmaske zurückgegeben.

Bit	Wert	Name	Bedeutung
0	1	Supported	Setzen, wenn der Befehl unterstützt wird.
1	2	Enabled	Setzen, wenn der Befehl aktiviert ist (ausgeführt werden kann).
2	4	Checked	Setzen, wenn der Befehl angehakt ist.

Das bedeutet, dass die Befehls-ID bei Rückgabe von 0 durch `QueryStatus` nicht als gültiger XMLSpy Befehl erkannt wird. Wenn `QueryStatus` einen Wert `1` oder `5` zurückgibt, wird der Befehl deaktiviert.

29.4.7.3.3 Events

Das XMLSpyControl ActiveX Control stellt die folgenden Verbindungspunkt-Events bereit:

`OnUpdateCmdUI`[2124]
`OnOpenedOrFocused`[2124]
`OnCloseEditingWindow`[2122]
`OnFileChangedAlert`[2123]
`OnContextChanged`[2123]

`OnDocumentOpened`[2123]
`OnValidationWindowUpdated`[2125]

29.4.7.3.3.1 OnCloseEditingWindow

Event: `OnCloseEditingWindow` (i_strFilePath als String) als boolean

Dispatch Id: *1002*

Beschreibung:
Dieses Event wird ausgelöst, wenn XMLSpy ein bereits geöffnetes Dokument schließen muss. Als Antwort auf dieses Event müssen Clients das mit *i_strFilePath* verknüpfte Bearbeitungsfenster schließen. Bei Rückgabe

© 2016-2022 Altova GmbH

von *true* von diesem Event, wird angezeigt, dass der Client das Dokument geschlossen hat. Clients können *false* zurückgeben, wenn keine bestimmte Behandlung erforderlich ist und XMLSpyControl versuchen soll, das Bearbeitungsfenster zu schließen und das damit verknüpfte Dokument-Control zu zerstören.

29.4.7.3.3.2 OnContextChanged

Event: `OnContextChanged` (`i_strContextName` als String, `i_bActive` als bool) als bool

Dispatch Id: *1004*

Beschreibung:
Dieses Event wird ausgelöst, wenn in XMLSpy einer der folgenden Operationskontexte aktiviert oder deaktiviert wird:

- XSLT Profiling - `"XSLTProfiling"` wird als Kontextname übergeben
- XSLT / XQuery Debugging - `"DebuggingXSLT"` wird als Kontextname übergeben
- SOAP Debugging - `"DebuggingSOAP"` wird als Kontextname übergeben (Nur Enterprise Edition)

29.4.7.3.3.3 OnDocumentOpened

Event: `OnDocumentOpened` (`objDocument` als Document)

Dispatch Id: *1*

Beschreibung:
Dieses Event wird immer, wenn ein Dokument geöffnet wird, ausgelöst. Das Argument `objDocument` ist ein `Document` Objekt aus dem XMLSpy Automation Interface und kann dazu verwendet werden, weitere Details zum Dokument abzurufen oder weitere Operationen durchzuführen. Bei Integration auf Dokumentebene ist es oft besser, stattdessen das Event `XMLSpyControlDocument.OnDocumentOpened` 2131 zu verwenden.

29.4.7.3.3.4 OnFileChangedAlert

Event: `OnFileChangedAlert` (`i_strFilePath` als String) als bool

Dispatch Id: *1001*

Beschreibung:
Dieses Event wird ausgelöst, wenn eine mit XMLSpyControl geladene Datei auf der Festplatte von einer anderen Applikation geändert wurde. Clients sollten true zurückgeben, wenn Sie das Event behandelt haben oder false, wenn XMLSpy es auf die übliche Art behandeln soll, also fragen soll, ob das Dokument neu geladen werden soll.

29.4.7.3.3.5 OnLicenseProblem

Event: `OnLicenseProblem` (`i_strLicenseProblemText` als String)

Dispatch Id: *1005*

Beschreibung:
Dieses Event wird ausgelöst, wenn XMLSpyControl feststellt, dass für dieses Control keine gültige Lizenz vorhanden ist. Wenn die Lizenz auf eine bestimmte Anzahl an Benutzern beschränkt ist, kann dies auch einige Zeit nach Initialisierung des Control geschehen. Dieses Event sollte dazu verwendet werden, um den Zugriff auf die Funktionalität dieses Control zu deaktivieren. Nach diesem Event blockiert dieses Control den Zugriff auf seine Funktionalitäten (und zeigt z.B. leere Fenster in seinen Controls an und gibt bei Anfragen Fehlermeldungen zurück).

29.4.7.3.3.6 OnOpenedOrFocused

Event: `OnOpenedOrFocused` (`i_strFilePath` als String, `i_bOpenWithThisControl` als bool)

Dispatch Id: *1000*

Beschreibung:
Bei Integration auf Applikationsebene informiert dieses Event Clients, dass ein Dokument von XMLSpy geöffnet oder aktiv gemacht wurde.

Bei Integration auf Dokumentebene gibt dieses Event dem Client die Anweisung, die Datei `i_strFilePath` in einem Dokumentfenster zu öffnen. Wenn die Datei bereits offen ist, sollte das entsprechende Dokumentfenster zum aktiven Fenster gemacht werden.

Wenn `i_bOpenWithThisControl true` ist, muss das Dokument mit XMLSpyControl geöffnet werden, da ein interner Zugriff erforderlich ist. Andernfalls kann die Datei mit anderen Editoren geöffnet werden.

29.4.7.3.3.7 OnToolWindowUpdated

Event: `OnToolWindowUpdated(pToolWnd` als long `)`

Dispatch Id: *1006*

Beschreibung:
Dieses Event wird ausgelöst, wenn das Fenster "Extras" aktualisiert wird.

29.4.7.3.3.8 OnUpdateCmdUI

Event: `OnUpdateCmdUI ()`

Dispatch Id: *1003*

Beschreibung:
Wird häufig aufgerufen, um dem Programmierer die Möglichkeit zu geben, den Status von XMLSpy Befehls mittels `XMLSpyControl.QueryStatus` 2122 zu überprüfen. Führen Sie an diesem Callback keine langen Operationen durch.

© 2016-2022 Altova GmbH

29.4.7.3.3.9 OnValidationWindowUpdated

Event: `OnValidationWindowUpdated` ()

Dispatch Id: *3*

Beschreibung:
Dieses Event wird ausgelöst, wenn das Fenster "Validierungsausgabe" mit neuen Informationen aktualisiert wird.

29.4.7.4 XMLSpyControlDocument

Eigenschaften:
`Appearance` 2126
`BorderStyle` 2126
`Document` 2126
`IsModified` 2127
`Path` 2127
`ReadOnly` 2127

Methoden:
`Exec` 2128
`New` 2128
`Open` 2128
`QueryStatus` 2128
`Reload` 2129
`Save` 2129
`SaveAs` 2129

Events:
`OnDocumentOpened` 2131
`OnDocumentClosed` 2130
`OnModifiedFlagChanged` 2131
`OnContextChanged` 2130
`OnFileChangedAlert` 2131
`OnActivate` 2130

Wenn das XMLSpyControl im Modus auf Dokumentebene integriert ist, wird jedes Dokument in einem eigenen Objekt vom Typ `XMLSpyControlDocument` angezeigt. Das `XMLSpyControlDocument` enthält immer nur ein Dokument, kann aber wiederverwendet werden, um hintereinander mehrere Dateien anzuzeigen.

Dieses Objekt ist ein komplettes ActiveX Control.

```
CLSID: 52A552E6-2AB8-4e3e-B545-BE998233DDA0
ProgID: Altova.XMLSpyControlDocument
```

29.4.7.4.1 Eigenschaften

Es sind die folgenden Eigenschaften definiert:

ReadOnly 2127
IsModified 2127
Path 2127
Appearance 2126
BorderStyle 2126

Zugriff auf die XMLSpyAPI:
Document 2126

29.4.7.4.1.1 Appearance

Eigenschaft: `Appearance` als short

Dispatch Id: *-520*

Beschreibung:
Bei einem Wert, der nicht gleich `0` ist, wird ein Client-Rand rund um das Control angezeigt. Der Standardwert ist `0`.

29.4.7.4.1.2 BorderStyle

Eigenschaft: `BorderStyle` als short

Dispatch Id: *-504*

Beschreibung:
Bei einem Wert `1` wird das Control mit einer dünnen Umrandung angezeigt. Der Standardwert ist `0`.

29.4.7.4.1.3 Document

Eigenschaft: `Document` als Document

Dispatch Id: *3*

Beschreibung:
Mit der Eigenschaft `Document` erhalten Sie Zugriff auf das `Document` Objekt der XMLSpy Automation Server API. Diese Schnittstelle bietet zusätzliche Funktionalitäten, die mit dem im Control geladenen Dokument verwendet werden können. Die Eigenschaft ist schreibgeschützt.

© 2016-2022 Altova GmbH

29.4.7.4.1.4 IsModified

Eigenschaft: `IsModified` als boolean (schreibgeschützt)

Dispatch Id: *1006*

Beschreibung:
`IsModified` ist *true,* wenn der Dokumentinhalt seit dem letzten Öffnen, Neuladen oder Speichern geändert wurde. Andernfalls ist es *false*.

29.4.7.4.1.5 Path

Eigenschaft: `Path` als String

Dispatch Id: *1005*

Beschreibung:
Definiert den vollständigen Pfadnamen des im Control geladenen Dokuments bzw. ruft diesen ab.

29.4.7.4.1.6 ReadOnly

Eigenschaft: `ReadOnly` als boolean

Dispatch Id: *1007*

Beschreibung:
Mit Hilfe dieser Eigenschaft können Sie den schreibgeschützten Status des Dokuments aktivieren und deaktivieren. Wenn `ReadOnly true` ist, können keine Änderungen vorgenommen werden.

29.4.7.4.2 Methoden

Es sind die folgenden Methoden definiert:

Behandlung von Dokumenten:
`New` 2128
`Open` 2128
`Reload` 2129
`Save` 2129
`SaveAs` 2129

Behandlung von Befehlen:
`Exec` 2128
`QueryStatus` 2128

29.4.7.4.2.1 Exec

Methode: `Exec` (`nCmdID` as long) als boolean

Dispatch Id: *8*

Beschreibung:
`Exec` ruft den XMLSpy Befehl mit der ID `nCmdID` auf. Wenn der Befehl ausgeführt werden kann, gibt die Methode `true` zurück. Diese Methode sollte nur dann aufgerufen werden, wenn in der Applikation gerade ein aktives Dokument verfügbar ist.

Um Befehle, geordnet nach Menüstruktur, abzurufen, verwenden Sie die Eigenschaft `MainMenu` 2119 von XMLSpyControl. Um Symbolleistenbefehle abzurufen, verwenden Sie die Eigenschaft `Toolbars` 2120 von XMLSpyControl.

29.4.7.4.2.2 New

Methode: `New` () als boolean

Dispatch Id: *1000*

Beschreibung:
Diese Methode initialisiert ein neues Mapping innerhalb des Control.

29.4.7.4.2.3 Open

Methode: `Open` (`strFilePath` als string) als boolean

Dispatch Id: *1001*

Beschreibung:
`Open` lädt die Datei `strFileName` als das neue Dokument in das Control.

29.4.7.4.2.4 QueryStatus

Methode: `QueryStatus` (`nCmdID` als long) als long

Dispatch Id: *9*

Beschreibung:
`QueryStatus` gibt den Status "enabled/disabled" und "checked/unchecked" des von `nCmdID`. definierten Befehls zurück. Der Status wird als Bitmaske zurückgegeben.

Bit	Wert	Name	Bedeutung

© 2016-2022 Altova GmbH

0	1	Supported	Setzen, wenn der Befehl unterstützt wird.
1	2	Enabled	Setzen, wenn der Befehl aktiviert ist (ausgeführt werden kann).
2	4	Checked	Setzen, wenn der Befehl angehakt ist.

Das bedeutet, dass die Befehls-ID bei Rückgabe von 0 durch `QueryStatus` nicht als gültiger XMLSpy Befehl erkannt wird. Wenn `QueryStatus` einen Wert `1` oder `5` zurückgibt, ist der Befehl deaktiviert. Der Client sollte die `QueryStatus` Methode des Dokument-Control aufrufen, wenn in der Applikation gerade ein aktives Dokument vorhanden ist.

29.4.7.4.2.5 Reload

Methode: `Reload` () als boolean

Dispatch Id: *1002*

Beschreibung:
`Reload` aktualisiert den Dokumentinhalt aus dem Dateisystem.

29.4.7.4.2.6 Save

Methode: `Save` () als boolean

Dispatch Id: *1003*

Beschreibung:
`Save` speichert das aktuelle Dokument unter dem Pfad `Path`[2127].

29.4.7.4.2.7 SaveAs

Methode: `SaveAs` (`strFileName` als string) als boolean

Dispatch Id: *1004*

Beschreibung:
`SaveAs` setzt `Path`[2127] auf *strFileName* und speichert das Dokument anschließend unter diesem Pfad.

29.4.7.4.3 Events

Das XMLSpyControlDocument ActiveX Control stellt die folgenden Verbindungspunkt-Events zur Verfügung:

`OnDocumentOpened`[2131]
`OnDocumentClosed`[2130]
`OnModifiedFlagChanged`[2131]

OnContextChanged 2130
OnFileChangedAlert 2131
OnActivate 2130
OnSetEditorTitle 2132

29.4.7.4.3.1 OnActivate

Event: `OnActivate` ()

Dispatch Id: *1005*

Beschreibung:
Dieses Event wird ausgelöst, wenn das Dokument-Control aktiviert ist, den Fokus hat und bereit für die Benutzereingabe ist.

29.4.7.4.3.2 OnContextChanged

Event: `OnContextChanged` (`i_strContextName` als String, `i_bActive` als bool) als bool

Dispatch Id: *1004*

Beschreibung:
Dieses Event wird ausgelöst, wenn dieses Dokument in einer anderen XMLSpy Ansicht angezeigt wird. Es werden die folgenden Werte übergeben:

- Grid-Ansicht - "`View_0`" wird als Kontextname übergeben
- Textansicht - "`View_1`" wird als der Kontextname übergeben
- Browser-Ansicht - "`View_2`" wird als der Kontextname übergeben.
- Schema-Ansicht - "`View_2`" wird als der Kontextname übergeben.
- Authentic-Ansicht - "`View_2`" wird als der Kontextname übergeben.
- WSDL-Ansicht - "`View_2`" wird als der Kontextname übergeben.

29.4.7.4.3.3 OnDocumentClosed

Event: `OnClosed` ()

Dispatch Id: *1001*

Beschreibung:
Dieses Event wird immer dann ausgelöst, wenn das in dieses Control geladene Event geschlossen wird. Das Argument `objDocument` ist ein `Document` Objekt aus dem XMLSpy Automation Interface und sollte mit Vorsicht verwendet werden.

© 2016-2022 Altova GmbH

29.4.7.4.3.4 OnDocumentOpened

Event: `OnDocumentOpened` (`objDocument` als Document)

Dispatch Id: *1000*

Beschreibung:
Dieses Event wird immer dann ausgelöst, wenn ein Dokument in diesem Control geöffnet wird. Das Argument `objDocument` ist ein `Document` Objekt aus dem XMLSpy Automation Interface und dient dazu, nähere Informationen über das Dokument abzurufen oder weitere Operationen auszuführen.

29.4.7.4.3.5 OnDocumentSaveAs

Event: `OnContextDocumentSaveAs` (`i_strFileName` als String)

Dispatch Id: *1007*

Beschreibung:
Dieses Event wird ausgelöst, wenn dieses Dokument intern unter einem neuen Namen gespeichert wird.

29.4.7.4.3.6 OnFileChangedAlert

Event: `OnFileChangedAlert` () als bool

Dispatch Id: *1003*

Beschreibung:
Dieses Event wird ausgelöst, wenn eine in dieses Control geladene Datei auf der Festplatte von einer anderen Applikation geändert wurde. Clients sollten true zurückgeben, wenn Sie das Event behandelt haben oder false, wenn XMLSpy es auf die übliche Art behandeln soll, also fragen soll, ob das Dokument neu geladen werden soll.

29.4.7.4.3.7 OnModifiedFlagChanged

Event: `OnModifiedFlagChanged` (`i_bIsModified` als boolean)

Dispatch Id: *1002*

Beschreibung:
Dieses Event wird immer dann ausgelöst, wenn das Dokument zwischen dem Status "geändert" und "nicht geändert" wechselt. Der Parameter *i_bIsModifed* ist *true,* wenn sich der Inhalt des Dokuments vom ursprünglichen Inhalt unterscheidet und ist andernfalls *false*.

29.4.7.4.3.8 OnSetEditorTitle

Event: OnSetEditorTitle ()

Dispatch Id: *1006*

Beschreibung:
Dieses Event wird ausgelöst, wenn das enthaltene Dokument intern umbenannt wird.

29.4.7.5 XMLSpyControlPlaceHolder

Eigenschaften für alle Arten von Platzhalterfenstern:
`PlaceholderWindowID` 2133

Eigenschaften für das Projekt-Platzhalterfenster:
`Project` 2133

Methoden für das Projekt-Platzhalterfenster:
`OpenProject` 2133
`CloseProject` 2134

Das `XMLSpyControlPlaceHolder` Control dient dazu, die zusätzlichen XMLSpy Fenster, wie Übersicht, Bibliothek oder das Projektfenster anzuzeigen. Es wird wie jedes andere ActiveX Control verwendet und kann überall in die Client-Applikation platziert werden.

```
CLSID: 135DEEF4-6DF0-47c2-8F8C-F145F5F3F672
ProgID: Altova.XMLSpyControlPlaceHolder
```

29.4.7.5.1 Eigenschaften

Es sind die folgenden Eigenschaften definiert:

`PlaceholderWindowID` 2133

Zugriff auf die XMLSpyAPI:
`Project` 2133

29.4.7.5.1.1 Label

Eigenschaft: `Label` als String (schreibgeschützt)

Dispatch Id: *1001*

Beschreibung:

© 2016-2022 Altova GmbH

Mit dieser Eigenschaft erhalten Sie Zugriff auf den Titel des Platzhalters. Diese Eigenschaft ist schreibgeschützt.

29.4.7.5.1.2 PlaceholderWindowID

Eigenschaft: `PlaceholderWindowID als XMLSpyControlPlaceholderWindow`[2135]

Dispatch Id: *1*

Beschreibung:
Mit Hilfe dieser Eigenschaft weiß das Objekt, welches XMLSpy Fenster im Client-Bereich des Control angezeigt werden soll. Die `PlaceholderWindowID` kann jederzeit auf jeden gültigen Wert der `XMLSpyControlPlaceholderWindow`[2135] Enumeration gesetzt werden. Das Control ändert seinen Status sofort und zeigt das neue XMLSpy Fenster an.

29.4.7.5.1.3 Project

Eigenschaft: `Project` als Project (schreibgeschützt)

Dispatch Id: *2*

Beschreibung:
Mit der Eigenschaft `Project` erhalten Sie Zugriff auf das `Project` Objekt der XMLSpy Automation Server API. Diese Schnittstelle bietet zusätzliche Funktionalitäten, die mit dem in das Control geladenen Projekt verwendet werden können. Die Eigenschaft gibt nur dann eine gültige Projektschnittstelle zurück, wenn das Platzhalterfenster `PlaceholderWindowID`[2133] mit einem Wert von `XMLSpyXProjectWindow (=3)` hat. Die Eigenschaft ist schreibgeschützt.

29.4.7.5.2 Methoden

Es sind die folgenden Methoden definiert:

`OpenProject`[2133]
`CloseProject`[2134]

29.4.7.5.2.1 OpenProject

Methode: `OpenProject` (`strFileName` als string) als boolean

Dispatch Id: *3*

Beschreibung:

`OpenProject` lädt die Datei `strFileName` als das neue Projekt in das Control. Wenn das Platzhalterfenster eine `PlaceholderWindowID`[2133] hat, die nicht mit `XMLSpyXProjectWindow (=3)` übereinstimmt, schlägt die Methode fehl.

29.4.7.5.2.2 CloseProject

Methode: `CloseProject` ()

Dispatch Id: *4*

Beschreibung:
`CloseProject` schließt das in das Control geladene Projekt. Wenn das Platzhalterfenster eine `PlaceholderWindowID`[2133] hat, die nicht mit `XMLSpyXProjectWindow (=3)` übereinstimmt, schlägt die Methode fehl.

29.4.7.5.3 Events

Das XMLSpyControlPlaceholder ActiveX Control stellt die folgenden Verbindungspunkt-Events zur Verfügung:
`OnModifiedFlagChanged`[2134]

29.4.7.5.3.1 OnModifiedFlagChanged

Event: `OnModifiedFlagChanged` (`i_bIsModified` als boolean)

Dispatch Id: *1*

Beschreibung:
Dieses Event wird nur bei Platzhalter-Controls mit einer `PlaceholderWindowID`[2133] von `XMLSpyXProjectWindow (=3)` ausgelöst. Das Event wird immer dann ausgelöst, wenn der Projektinhalt sich zwischen dem Status "geändert" und "nicht geändert" ändert. Der Parameter *i_bIsModifed* ist *true,* wenn sich der Projektinhalt vom ursprünglichen Inhalt unterscheidet und *false*, wenn dies nicht der Fall ist.

29.4.7.5.3.2 OnSetLabel

Event: `OnSetLabel` (i_strLabel als String)

Dispatch Id: *1000*

Beschreibung:
Dieses Event wird immer dann ausgelöst, wenn die Bezeichnung eines Platzhalter-Control-Fensters geändert wird.

© 2016-2022 Altova GmbH

29.4.7.6 Enumerationen

Es sind die folgenden Enumerationen definiert:

ICActiveXIntegrationLevel[2135]
XMLSpyControlPlaceholderWindow[2135]

29.4.7.6.1 ICActiveXIntegrationLevel

Mögliche Werte für die Eigenschaft IntegrationLevel[2119] für das XMLSpyControl.

```
ICActiveXIntegrationOnApplicationLevel  = 0
ICActiveXIntegrationOnDocumentLevel     = 1
```

29.4.7.6.2 XMLSpyControlPlaceholderWindow

Diese Enumeration enthält die Liste der unterstützten zusätzlichen XMLSpy Fenster.

```
XMLSpyControlNoToolWnd                          = -1
XMLSpyControlEntryHelperTopToolWnd       = 0
XMLSpyControlEntryHelperMiddleToolWnd           = 1
XMLSpyControlEntryHelperBottomToolWnd           = 2
XMLSpyControlValidatorOutputToolWnd             = 3
XMLSpyControlProjectWindowToolWnd               = 4
XMLSpyControlXSLTDebuggerContextToolWnd         = 5
XMLSpyControlXSLTDebuggerCallstackToolWnd= 6
XMLSpyControlXSLTDebuggerVariableToolWnd = 7
XMLSpyControlXSLTDebuggerWatchToolWnd           = 8
XMLSpyControlXSLTDebuggerTemplateToolWnd = 9
XMLSpyControlXSLTDebuggerInfoToolWnd            = 10
XMLSpyControlXSLTDebuggerMessageToolWnd         = 11
XMLSpyControlXSLTDebuggerTraceToolWnd           = 12
XMLSpyControlSOAPDebuggerToolWnd                = 13
XMLSpyControlXPathProfilerListToolWnd           = 14
XMLSpyControlXPathProfilerTreeToolWnd           = 15
XMLSpyControlXPathDialogToolWnd                 = 16
XMLSpyControlDBQueryManagerToolWnd       = 17
XMLSpyControlInfoToolWnd                        = 18
XMLSpyControlXSLOutlineToolWnd                  = 19
XMLSpyControlSchemaFindToolWnd                  = 20
XMLSpyControlXBRLFindToolWnd                    = 21
XMLSpyControlChartsToolWnd                      = 22
```

30 Anhänge

Diese Anhänge enthalten technische Informationen über XMLSpy und wichtige Lizenzinformationen. Die einzelnen Anhänge sind in die folgenden Unterabschnitte gegliedert:

Prozessorinformationen 2137

- Informationen zum XSLT- und XQuery-Prozessor 2137
- XSLT- und XPath/XQuery-Erweiterungsfunktionen 2143

Datentypkonvertierungen zwischen Datenbanken und XML-Schemas 2256

- Datenbanken in XML-Schemas 2256
- XML-Schemas in Datenbanken 2262

Technische Daten 2270

- Betriebssystem und Arbeitsspeicheranforderungen 2270
- Altova XSLT- und XQuery-Prozessor 2270
- Unicode-Unterstützung 2271
- Internet-Verwendung 2271

Lizenzinformationen 2273

- Electronic Software Distribution (EDI) 2273
- Software-Aktivierung und Lizenzüberwachung 2274
- Endbenutzer-Lizenzvereinbarung 2275

© 2016-2022 Altova GmbH

30.1 Informationen zum XSLT- und XQuery-Prozessor

Der XSLT- und der XQuery-Prozessor von XMLSpy hält sich genau an die W3C-Spezifikationen und ist daher strenger als die früheren Altova-Prozessoren, wie z.B. die in frühere Versionen von XMLSpy integrierten . Infolgedessen werden auch leichte Fehler, die von früheren Prozessoren ignoriert wurden, von XMLSpy als Fehler gekennzeichnet.

Zum Beispiel:

- Wenn das Ergebnis eines Pfad-Operators sowohl Nodes als auch Nicht-Nodes enthält, wird ein Typfehler (`err:XPTY0018`) ausgegeben.
- Wenn `E1` in einem Pfadausdruck `E1/E2` nicht zu einer Node-Sequenz ausgewertet wird, wird ein Typfehler (`err:XPTY0019`) ausgegeben.

Ändern Sie bei Auftreten eines solchen Fehlers je nach Bedarf, entweder das XSLT/XQuery-Dokument oder das Instanzdokument.

In diesem Abschnitt sind implementierungsspezifische Funktionalitäten der Prozessoren geordnet nach Spezifikation beschrieben:

- XSLT 1.0 2137
- XSLT 2.0 2137
- XSLT 3.0 2139
- XQuery 1.0 2140
- XQuery 3.1 2142

30.1.1 XSLT 1.0

Der XSLT 1.0-Prozessor von XMLSpy entspricht der XSLT 1.0 Recommendation vom 16. November 1999 und der XPath 1.0 Recommendation vom 16. November 1999 des World Wide Web Consortium (W3C). Beachten Sie die folgenden Informationen zur Implementierung.

Anmerkungen zur Implementierung

Wenn das `method`-Attribut von `xsl:output` auf HTML gesetzt ist oder wenn standardmäßig die HTML-Ausgabe ausgewählt ist, werden Sonderzeichen in der XML- oder XSLT-Datei als HTML-Zeichenreferenzen in das HTML-Ausgabedokument eingefügt. So wird z.B. das Zeichen U+00A0 (die hexadezimale Zeichenreferenz für ein geschütztes Leerzeichen) entweder als Zeichenreferenz (` ` or ` `) oder als Entity-Referenz ` ` in den HTML-Code eingefügt.

30.1.2 XSLT 2.0

In diesem Abschnitt:

- Prozessorkonformität 2138
- Rückwärtskompatibilität 2138

- Namespaces 2138
- Schema-Fähigkeit 2139
- Implementierungsspezifisches Verhalten 2139

Standardkonformität

Der XSLT 2.0-Prozessor von XMLSpy entspricht der XSLT 2.0 Recommendation vom 23. Jänner 2007 und der XPath 2.0 Recommendation vom 14. Dezember 2010 des World Wide Web Consortium (W3C).

Rückwärtskompatibilität

Der XSLT 2.0-Prozessor ist rückwärtskompatibel. Die Rückwärtskompatibilität des XSLT 2.0.-Prozessors kommt nur dann zum Einsatz, wenn Sie den XSLT 2.0-Prozessor zur Verarbeitung eines XSLT 1.0 Stylesheets verwenden. Beachten Sie, dass sich das Ergebnis des XSLT 1.0-Prozessors und des rückwärtskompatiblen XSLT 2.0.Prozessors unter Umständen unterscheiden kann.

Namespaces

In Ihrem XSLT 2.0 Stylesheet sollten die folgenden Namespaces deklariert sein, damit Sie die in XSLT 2.0 verfügbaren Typ-Konstruktoren und Funktionen verwenden können. Normalerweise werden die unten aufgelisteten Präfixe verwendet; bei Bedarf können Sie auch andere Präfixe verwenden.

Namespace Name	Präfix	Namespace URI
XML Schema-Typen	`xs:`	`http://www.w3.org/2001/XMLSchema`
XPath 2.0-Funktionen	`fn:`	`http://www.w3.org/2005/xpath-functions`

Normalerweise werden diese Namespaces im Element `xsl:stylesheet` oder `xsl:transform` deklariert, wie unten gezeigt:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions"
   ...
</xsl:stylesheet>
```

Beachten Sie die folgenden Punkte:

- Der XSLT 2.0-Prozessor verwendet als **Standard-Funktions-Namespace** den Namespace für XPath 2.0- und XQuery 1.0-Funktionen (siehe Tabelle oben). Sie können daher XPath 2.0- und XSLT 2.0-Funktionen in Ihrem Stylesheet ohne Präfix verwenden. Wenn Sie den Namespace für XPath 2.0-Funktionen in Ihrem Stylesheet mit einem Präfix deklarieren, können Sie zusätzlich dazu das in der Deklaration zugewiesene Präfix verwenden.
- Bei Verwendung von Typ-Konstruktoren und Typen aus dem XML Schema-Namespace, muss bei Aufruf des Typ-Konstruktors (z.B. `xs:date`) das in der jeweiligen Namespace-Deklaration verwendeten Präfix verwendet werden.
- Einige XPath 2.0-Funktionen haben denselben Namen wie XML Schema-Datentypen. So gibt es z.B. für die XPath-Funktionen `fn:string` und `fn:boolean` XML-Schema-Datentypen mit demselben lokalen Namen: `xs:string` und `xs:boolean`. Wenn Sie daher den XPath-Ausdruck `string('Hello')` verwenden, wird der Ausdruck als `fn:string('Hello')` ausgewertet und nicht als `xs:string('Hello')`.

© 2016-2022 Altova GmbH

Schemafähigkeit

Der XSLT 2.0-Prozessor ist schemafähig. Sie können daher benutzerdefinierte Schematypen und die `xsl:validate`-Anweisung verwenden.

Implementierungsspezifisches Verhalten

Im Folgenden finden Sie eine Beschreibung, wie der XSLT 2.0-Prozessor implementierungsspezifische Aspekte des Verhaltens bestimmter XSLT 2.0-Funktionen behandelt.

`xsl:result-document`
Zusätzlich werden die folgenden Kodierungen unterstützt: `x-base16tobinary` und `x-base64tobinary`.

`function-available`
Die Funktion überprüft, ob in-scope-Funktionen (XSLT, XPath und Erweiterungsfunktionen) verfügbar sind.

`unparsed-text`
Das Attribut `href` akzeptiert (i) relative Pfade für Dateien im Basis-URI-Ordner und (ii) absolute Pfade mit oder ohne das `file://`-Protokoll. Zusätzlich werden die folgenden (Altova-spezifischen) Kodierungen unterstützt: `binarytobase16` und `binarytobase64`.

`unparsed-text-available`
Das Attribut `href` akzeptiert (i) relative Pfade für Dateien im Basis-URI-Ordner und (ii) absolute Pfade mit oder ohne das `file://`-Protokoll. Zusätzlich werden die folgenden (Altova-spezifischen) Kodierungen unterstützt: `binarytobase16` und `binarytobase64`.

Anmerkung: Die folgenden Kodierungswerte, die in früheren Versionen von AltovaXML, dem Vorgängerprodukt von RaptorXML, verwendet wurden, werden nun nicht mehr verwendet:`base16tobinary`, `base64tobinary`, `binarytobase16` und `binarytobase64`.

30.1.3 XSLT 3.0

Der XSLT 3.0-Prozessor von XMLSpy entspricht der XSLT 3.0 Recommendation vom 8. Juni 2017 und der XPath 3.1 Recommendation vom 21. März 2017 des World Wide Web Consortium (W3C).

Der XSLT 3.0-Prozessor hat dieselben implementierungsspezifischen Eigenschaften wie der XSLT 2.0-Prozessor[2137]. Zusätzlich dazu unterstützt er eine Reihe neuer XSLT 3.0-Funktionen: XPath- und XQuery 3.1-Funktionen und Operatoren und die XPath 3.1-Spezifikation.

Anmerkung: Die optionale Streaming-Funktion wird derzeit nicht unterstützt., d.h. das gesamte Dokument wird unabhängig vom Wert des Attributs `streamable` in den Arbeitsspeicher geladen. Falls genügend Arbeitsspeicher zur Verfügung steht, (i) wird das gesamte Dokument - ohne Streaming - verarbeitet (ii) werden "guaranteed-streamable" Konstrukte korrekt verarbeitet, so als würde bei der Ausführung Streaming verwendet und (iii) Streaming-Fehler würden nicht erkannt werden. In 64-Bit-Apps sollte die Nicht-Streaming-Verarbeitung kein Problem darstellen. Falls zu wenig Arbeitsspeicher zur Verfügung steht, können Sie mehr Speicherplatz zum System hinzufügen.

30.1.4 XQuery 1.0

In diesem Abschnitt:

- Standardkonformität 2140
- Schema-Fähigkeit 2140
- Kodierung 2140
- Namespaces 2138
- XML-Quelle und Validierung 2141
- Statische und dynamische Typüberprüfung 2141
- Bibliotheksmodule 2141
- Externe Funktionen 2142
- Collations 2142
- Präzision von numerischen Daten 2142
- Unterstützung für XQuery-Anweisungen 2142

Standardkonformität

Der XQuery 1.0-Prozessor von XMLSpy entspricht der XQuery 1.0 Recommendation vom 14. Dezember 2010 des W3C. Der Query-Standard stellt bei vielen Funktionen frei, wie viele diese zu implementieren sind. Im Folgenden finden Sie eine Liste, wie der Altova XQuery 1.0-Prozessor diese Funktionen implementiert.

Schemafähigkeit

Der Altova XQuery 1.0-Prozessor ist **schemafähig.**

Kodierung

Die UTF-8 und die UTF-16 Zeichen-Kodierungen werden unterstützt.

Namespaces

Die folgenden Namespace-URIs und die damit verknüpften Bindings sind vordefiniert.

Namespace-Name	Präfix	Namespace URI
XML Schema-Typen	`xs:`	`http://www.w3.org/2001/XMLSchema`
Schema-Instanz	`xsi:`	`http://www.w3.org/2001/XMLSchema-instance`
Vordefinierte Funktionen	`fn:`	`http://www.w3.org/2005/xpath-functions`
Lokale Funktionen	`local:`	`http://www.w3.org/2005/xquery-local-functions`

Beachten Sie bitte die folgenden Punkte:

- Der Altova XQuery 1.0-Prozessor ist so konfiguriert, dass die oben aufgelisteten Präfixe an die entsprechenden Namespaces gebunden sind.
- Da der oben angeführte Namespace für vordefinierte Funktionen der Standard-Funktions-Namespace in XQuery ist, muss beim Aufruf von vordefinierten Funktionen das Präfix `fn:` nicht verwendet werden (`string("Hello")` ruft z.B. die Funktion `fn:string` auf). Das Präfix `fn:` kann jedoch verwendet

© 2016-2022 Altova GmbH

werden, um eine vordefinierte Funktion aufzurufen, ohne die Namespace im Abfrage-Prolog deklarieren zu müssen (z.B.: `fn:string("Hello")`).
- Sie können den Standard-Funktions-Namespace durch Deklarierung des `default function namespace`-Ausdrucks im Abfrageprolog ändern.
- Bei Verwendung von Typen aus dem XML Schema-Namespace kann das Präfix `xs:` verwendet werden, ohne dass Sie den Namespace explizit deklarieren müssen und dieses Präfix im Abfrageprolog daran binden müssen. (Beispiele: `xs:date` und `xs:yearMonthDuration`.) Wenn Sie ein anderes Präfix verwenden möchten, muss dieses im Abfrageprolog für die Namespaces explizit deklariert werden. (Beispiel `declare namespace alt = "http://www.w3.org/2001/XMLSchema"; alt:date("2004-10-04")`.)
- Beachten Sie, dass die Datentypen `untypedAtomic`, `dayTimeDuration` und `yearMonthDuration` mit den Candidate Recommendations vom 23 January 2007 aus dem XPath Datentypen-Namespace in den XML-Schema Namespace verschoben wurden, d.h. `xs:yearMonthDuration`.

Wenn Namespaces für Funktionen, Typ-Konstruktoren, Node Tests usw. falsch zugewiesen wurden, wird ein Fehler ausgegeben. Beachten Sie jedoch, dass einige Funktionen denselben Namen wie Schema-Datentypen haben, z.B. `fn:string` und `fn:boolean`. (Sowohl `xs:string` als auch `xs:boolean` ist definiert.) Das Namespace-Präfix legt fest, ob die Funktion oder der Typ-Konstruktor verwendet wird.

XML-Quelldokument und Validierung

XML-Dokumente, die bei der Ausführung eines XQuery-Dokuments mit dem Altova XQuery 1.0-Prozessor verwendet werden, müssen wohlgeformt sein. Sie müssen jedoch nicht gemäß einem XML-Schema gültig sein. Wenn die Datei nicht gültig ist, wird die ungültige Datei ohne Schemainformationen geladen. Wenn die XML-Datei mit einem externen Schema verknüpft ist und gemäß diesem Schema gültig ist, werden für die XML-Daten nachträglich Validierungsinformationen generiert und für die Auswertung der Abfrage verwendet.

Statische und dynamische Typ-Überprüfung

In der statischen Analysephase werden Aspekte der Abfrage überprüft wie z.B. die Syntax, ob externe Referenzen (z.B. für Module) vorhanden sind, ob aufgerufene Funktionen und Variablen definiert sind, usw. Wenn in dieser Phase ein Fehler gefunden wird, wird eine Meldung ausgegeben und die Ausführung wird gestoppt.

Die dynamische Typ-Überprüfung wird in Laufzeit durchgeführt, während die Abfrage ausgeführt wird. Wenn ein Typ mit den Anforderungen einer Operation nicht kompatibel ist, wird ein Fehler ausgegeben. So gibt z.B. der Ausdruck `xs:string("1") + 1` einen Fehler zurück, weil die Operation "Addition" nicht an einem Operanden vom Typ `xs:string` ausgeführt werden kann.

Bibliotheksmodule

Bibliotheksmodule dienen zum Speichern von Funktionen und Variablen, damit diese wiederverwendet werden können. Der Altova XQuery 1.0-Prozessor unterstützt Module, die in einer **einzigen externen XQuery-Datei** gespeichert sind. Eine solche Moduldatei muss im Prolog eine `module`-Deklaration enthalten, in der ein Target Namespace zugewiesen wird. Hier ein Beispielmodul:

```
module namespace libns="urn:module-library";
declare variable $libns:company := "Altova";
declare function libns:webaddress() { "http://www.altova.com" };
```

Alle im Modul deklarierten Funktionen und Variablen gehören zu dem mit dem Modul verknüpften Namespace. Das Modul wird durch Import in eine XQuery-Datei mittels der `import module`-Anweisung im Abfrageprolog

verwendet. Die `import module`-Anweisung importiert nur Funktionen und Variablen, die direkt in der Bibliotheksmodul-Datei deklariert sind:

```
import module namespace modlib = "urn:module-library" at "modulefilename.xq";
if    ($modlib:company = "Altova")
then   modlib:webaddress()
else   error("No match found.")
```

Externe Funktionen

Externe Funktionen, d.h. diejenigen Funktionen, die das Schlüsselwort `external` verwenden, werden nicht unterstützt:

```
declare function hoo($param as xs:integer) as xs:string external;
```

Collations

Die Standard-Collation ist die Unicode Codepoint Collation, die Strings auf Basis ihrer Unicode-Codepunkte vergleicht. Andere unterstützte Collations sind die hier[2143] aufgelisteten ICU-Collations. Um eine bestimmte Collation zu verwenden, geben Sie ihre in der Liste der unterstützten Collations[2143] angeführte URI an. String-Vergleiche, wie die Funktionen `fn:max` und `fn:min` werden anhand der angegebenen Collation durchgeführt. Wenn die Collation-Option nicht definiert ist, wird die Standard-Unicode Codepoint Collation verwendet.

Präzision von numerischen Typen

- Der Datentyp `xs:integer` hat eine beliebige Präzision, d.h. er kann beliebig viele Stellen haben.
- Der Datentyp `xs:decimal` kann nach dem Dezimalpunkt maximal 20 Stellen haben.
- Die Datentypen `xs:float` und `xs:double` sind auf 15 Stellen beschränkt.

Unterstützung für XQuery-Anweisungen

Die `Pragma` Anweisung wird nicht unterstützt. Gegebenenfalls wird sie ignoriert und der Fallback-Ausdruck wird evaluiert.

30.1.5 XQuery 3.1

Der XQuery 3.1-Prozessor von XMLSpy entspricht der XQuery 3.1 Recommendation vom 21. März 2017 des World Wide Web Consortium (W3C) und unterstützt XPath- und XQuery-Funktionen 3.1. Die XQuery 3.1-Spezifikation ist eine Übermenge der 3.0-Spezifikation. Daher unterstützt der XQuery 3.1-Prozessor XQuery 3.0-Funktionen..

Die implementierungsspezifischen Eigenschaften sind dieselben wie für XQuery 1.0[2140].

© 2016-2022 Altova GmbH

30.2 XSLT- und XPath/XQuery-Funktionen

Dieser Abschnitt enthält eine Liste von Altova-Erweiterungsfunktionen und anderen Erweiterungsfunktionen, die in XPath und/oder XQuery-Ausdrücken verwendet werden können. Altova-Erweiterungsfunktionen können mit dem XSLT- und XQuery-Prozessor von Altova verwendet werden und bieten zusätzliche Funktionalitäten zu den in den W3C-Standards definierten Funktionsbibliotheken.

Allgemeine Punkte

Beachten Sie bitte die folgenden allgemeinen Punkte:

- Funktionen aus den in den W3C-Spezifikationen definierten core-Funktionsbibliotheken können ohne Präfix aufgerufen werden, da der XSLT- und XQuery-Prozessor Funktionen, die kein Präfix haben, als Funktionen des Standard-Funktions-Namespace liest. Der Standard-Funktions-Namespace ist derjenige, der in den XPath/XQuery-Funktionsspezifikationen definiert ist `http://www.w3.org/2005/xpath-functions`. Wenn dieser Namespace in einem XSLT- oder XQuery-Dokument explizit deklariert ist, kann das in der Namespace-Deklaration definierte Präfix optional auch in Funktionsnamen verwendet werden.
- Grundsätzlich gilt: Wenn bei einer Funktion eine Sequenz von einem Datenelement als Argument erwartet wird und eine Sequenz von mehr als einem Datenelement gesendet wird, wird ein Fehler zurückgegeben.
- Alle String-Vergleiche werden unter Verwendung der Unicode Codepoint Collation ausgeführt.
- Ergebnisse, bei denen es sich um QNames handelt, werden in der Form `[prefix:]localname` serialisiert.

Präzision von xs:decimal

Die Präzision bezieht sich auf die Anzahl der Stellen in einer Zahl. Laut Spezifikation sind mindestens 18 Stellen erforderlich. Bei Divisionen, bei denen ein Ergebnis vom Typ `xs:decimal` erzeugt wird, beträgt die Präzision 19 Kommastellen ohne Runden.

Implizite Zeitzone

Beim Vergleich zweier `date`, `time`, oder `dateTime` Werte müssen die Zeitzonen der verglichenen Werte bekannt sein. Wenn die Zeitzone in einem solchen Wert nicht explizit angegeben ist, wird die implizite Zeitzone verwendet. Als implizite Zeitzone wird die der Systemuhr verwendet. Der Wert kann mit Hilfe der Funktion `implicit-timezone()` überprüft werden.

Collations

Die Standard-Collation ist die Unicode Codepoint Collation, die Strings auf Basis ihrer Unicode-Codepunkte vergleicht. Es wird der Unicode Collation Algorithmus verwendet. Andere unterstützte Collations sind die unten aufgelisteten ICU-Collations. Um eine bestimmte dieser Collations zu verwenden, geben Sie ihre in der nachstehenden Tabelle angegebene URI an. String-Vergleiche, wie die Funktionen `max` und `min` werden anhand der angegebenen Collation durchgeführt. Wenn die Collation-Option nicht definiert ist, wird die Standard-Unicode Codepoint Collation verwendet.

Sprache	URIs
`da`: Dänisch	`da_DK`

de: Deutsch	de_AT, de_BE, de_CH, de_DE, de_LI, de_LU
en: Englisch	en_AS, en_AU, en_BB, en_BE, en_BM, en_BW, en_BZ, en_CA, en_GB, en_GU, en_HK, en_IE, en_IN, en_JM, en_MH, en_MP, en_MT, en_MU, en_NA, en_NZ, en_PH, en_PK, en_SG, en_TT, en_UM, en_US, en_VI, en_ZA, en_ZW
es: Spanisch	es_419, es_AR, es_BO, es_CL, es_CO, es_CR, es_DO, es_EC, es_ES, es_GQ, es_GT, es_HN, es_MX, es_NI, es_PA, es_PE, es_PR, es_PY, es_SV, es_US, es_UY, es_VE
fr: Französisch	fr_BE, fr_BF, fr_BI, fr_BJ, fr_BL, fr_CA, fr_CD, fr_CF, fr_CG, fr_CH, fr_CI, fr_CM, fr_DJ, fr_FR, fr_GA, fr_GN, fr_GP, fr_GQ, fr_KM, fr_LU, fr_MC, fr_MF, fr_MG, fr_ML, fr_MQ, fr_NE, fr_RE, fr_RW, fr_SN, fr_TD, fr_TG
it: Italienisch	it_CH, it_IT
ja: Japanisch	ja_JP
nb: Norwegisch (Bokmal)	nb_NO
nl: Holländisch	nl_AW, nl_BE, nl_NL
nn: Norwegisch (Nynorsk)	nn_NO
pt: Portugiesisch	pt_AO, pt_BR, pt_GW, pt_MZ, pt_PT, pt_ST
ru: Russisch	ru_MD, ru_RU, ru_UA
sv: Schwedisch	sv_FI, sv_SE

Namespace-Achse
Die Namespace-Achse wird in XPath 2.0 nicht mehr verwendet, wird aber weiterhin unterstützt. Um Namespace-Informationen mit XPath 2.0-Mechanismen aufzurufen, verwenden Sie die Funktionen `in-scope-prefixes()`, `namespace-uri()` und `namespace-uri-for-prefix()`.

30.2.1 Altova-Erweiterungsfunktionen

Altova-Erweiterungsfunktionen können in XPath/XQuery-Ausdrücken verwendet werden. Dadurch stehen neben den Funktionen in der Standardbibliothek der XPath-, XQuery- und XSLT-Funktionen zusätzliche Funktionen zur Verfügung. Die Altova-Erweiterungsfunktionen befinden sich im **Altova-Erweiterungsfunktions-Namespace, `http://www.altova.com/xslt-extensions`** und sind in diesem Abschnitt mit dem Präfix **`altova:`**, das als an diesen Namespace gebunden angenommen wird, gekennzeichnet. Beachten Sie, dass manche Funktionen in zukünftigen Versionen Ihres Produkts eventuell nicht mehr unterstützt werden oder dass sich das Verhalten einzelner Funktionen ändern kann. Um zu sehen, welche Altova-Erweiterungsfunktionen unterstützt werden, lesen Sie bitte die Dokumentation zur jeweiligen Release.

Die in der "XPath/XQuery Functions"-Spezifikation des W3C definierten Funktionen können (i) in einem XSLT-

© 2016-2022 Altova GmbH

Kontext in XPath-Ausdrücken und (ii) in einem XQuery-Dokument in XQuery-Ausdrücken verwendet werden. In dieser Dokumentation sind die Funktionen, die im Zusammenhang mit XPath in XSLT verwendet werden können, mit einem **XP**-Symbol und Funktionen, die im Zusammenhang mit XQuery verwendet werden können, mit einem **XQ**-Symbol markiert; sie fungieren als XQuery-Funktionen. In den XSLT-Spezifikationen des W3C (nicht in den "XPath/XQuery Functions"-Spezifikationen) sind außerdem Funktionen definiert, die in XSLT-Dokumenten in XPath-Ausdrücken verwendet werden können. Diese Funktionen sind mit dem Symbol **XSLT** gekennzeichnet und werden als XSLT-Funktionen bezeichnet. In welcher XPath/XQuery- und XSLT-Version eine Funktion verwendet werden kann, wird in der Beschreibung der Funktion (*siehe Symbole unten)* angegeben. Funktionen aus der XPath/XQuery- und XSLT-Funktionsbibliothek werden ohne Präfix aufgelistet. Erweiterungsfunktionen aus anderen Bibliotheken wie z.B. Altova-Erweiterungsfunktionen werden mit einem Präfix angegeben.

XPath-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	**XP1** **XP2** **XP3.1**
XSLT-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	**XSLT1** **XSLT2** **XSLT3**
XQuery-Funktionen (in XQuery-Ausdrücken in XQuery verwendet):	**XQ1** **XQ3.1**

XSLT-Funktionen 2145

XSLT-Funktionen können in XPath-Ausdrücken nur im XSLT-Kontext verwendet werden (ähnlich wie die XSLT 2.0-Funktionen `current-group()` oder `key()`). Diese Funktionen sind nicht für Nicht-XSLT-Kontext gedacht und funktionieren in einem solchen Kontext (z.B. in einem XQuery-Kontext) nicht. Beachten Sie, dass XSLT-Funktionen für XBRL nur mit Altova Produkteditionen verwendet werden können, die XBRL unterstützen.

XPath/XQuery-Funktionen

XPath/XQuery-Funktionen können sowohl in XPath-Ausdrücken im XSLT-Kontext als auch in XQuery-Ausdrücken verwendet werden.

- Datum/Uhrzeit 2148
- Standort 2167
- Bildbezogene 2179
- Numerisch 2183
- Sequenz 2205
- String 2214
- Verschiedenes 2221

Diagrammfunktionen (nur Enterprise- und Server-Editionen)

Altova-Erweiterungsfunktionen für Diagramme 2223 werden nur in der Enterprise und Server Edition von Altova-Produkten unterstützt und ermöglichen die Generierung von Diagrammen anhand von XML-Daten.

30.2.1.1 XSLT-Funktionen

XSLT-Erweiterungsfunktionen können in XPath-Ausdrücken in einem XSLT-Kontext verwendet werden. In einem Nicht-XSLT-Kontext (z.B. in einem XQuery-Kontext) funktionieren sie nicht.

Anmerkung zur Benennung von Funktionen und zur Anwendbarkeit der Sprache

Altova-Erweiterungsfunktionen können in XPath/XQuery-Ausdrücken verwendet werden. Dadurch stehen neben den Funktionen in der Standardbibliothek der XPath-, XQuery- und XSLT-Funktionen zusätzliche Funktionen zur Verfügung. Die Altova-Erweiterungsfunktionen befinden sich im **Altova-Erweiterungsfunktions-Namespace, `http://www.altova.com/xslt-extensions`** und sind in diesem Abschnitt mit dem Präfix **`altova:`**, das als an diesen Namespace gebunden angenommen wird, gekennzeichnet. Beachten Sie, dass manche Funktionen in zukünftigen Versionen Ihres Produkts eventuell nicht mehr unterstützt werden oder dass sich das Verhalten einzelner Funktionen ändern kann. Um zu sehen, welche Altova-Erweiterungsfunktionen unterstützt werden, lesen Sie bitte die Dokumentation zur jeweiligen Release.

XPath-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XP1 XP2 XP3.1
XSLT-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XSLT1 XSLT2 XSLT3
XQuery-Funktionen (in XQuery-Ausdrücken in XQuery verwendet):	XQ1 XQ3.1

Allgemeine Funktionen

distinct-nodes [altova:]

`altova:distinct-nodes(`*`node()*`***`) as node()*`** XSLT1 XSLT2 XSLT3
Erhält eine Gruppe von einem oder mehreren Nodes als Input und gibt dieselbe Gruppe ohne Nodes mit doppelt vorhandenen Werten zurück. Der Vergleich wird mittels der XPath/XQuery-Funktion `fn:deep-equal` durchgeführt.

Beispiele

- **`altova:distinct-nodes`**`(country)` gibt alle Child `country` Nodes ohne diejenigen mit doppelt vorhandenen Werten zurück.

evaluate [altova:]

`altova:evaluate(XPathExpression` *`as xs:string`*`**[, ValueOf$p1, ... ValueOf$pN])`** XSLT1 XSLT2 XSLT3
Erhält einen XPath-Ausdruck als obligatorisches Argument, der als String übergeben wird, und gibt das Resultat des ausgewerteten Ausdrucks zurück. Beispiel: **`altova:evaluate('//Name[1]')`** gibt den Inhalt des ersten `Name` Elements im Dokument zurück. Beachten Sie, dass der Ausdruck `//Name[1]` durch Einschließen in einfache Anführungszeichen als String übergeben wird.

Die Funktion `altova:evaluate` kann zusätzliche (optionale) Argumente erhalten. Diese Argumente sind die Werte der einzelnen im Geltungsbereich befindlichen Variablen und haben die Namen `p1`, `p2`, `p3`... `pN`. Beachten Sie zur Verwendung die folgenden Punkte: (i) Die Variablennamen müssen die Form `pX` haben, wobei `X` eine Ganzzahl ist; (ii) die Argumente der Funktion `altova:evaluate` (*siehe Signatur oben*) liefern vom zweiten Argument an die Werte der Variablen, wobei die Reihenfolge der Argumente der numerisch geordneten Variablensequenz entspricht: `p1` bis `pN`. Das zweite Argument wird der Wert der Variablen `p1`, das dritte Argument der der Variablen `p2`, usw.; (iii) Die Werte der Variablen müssen vom Typ `item*` sein

Beispiel

```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="altova:evaluate($xpath, 10, 20, 'hi')" />
```

gibt aus `"hi 20 10"`

© 2016-2022 Altova GmbH

Beachten Sie im obigen Beispiel folgende Punkte:

- Das zweite Argument des Ausdrucks `altova:evaluate` ist der der Variablen `$p1` zugewiesene Wert, das dritte Argument ist das der Variablen `$p2` zugewiesene usw.
- Beachten Sie, dass das vierte Argument der Funktion ein String-Wert ist. Als String-Wert wird dieser innerhalb von Anführungszeichen gesetzt.
- Das `select` Attribut des Elements `xs:variable` liefert den XPath-Ausdruck. Da dieser Ausdruck den Typ `xs:string`, haben muss, wird er in einfache Anführungszeichen gesetzt.

Weitere Beispiele für die Verwendung der Variablen

- `<xsl:variable name="xpath" select="'$p1'" />`
  `<xsl:value-of select="altova:evaluate($xpath, //Name[1])" />`
  Gibt den Wert des ersten `Name` *Elements zurück.*

- `<xsl:variable name="xpath" select="'$p1'" />`
  `<xsl:value-of select="altova:evaluate($xpath, '//Name[1]')" />`
  Gibt "`//Name[1]`*" aus*

Die `altova:evaluate()` Erweiterungsfunktion ist in Situationen nützlich, in denen ein XPath-Ausdruck im XSLT-Stylesheet einen oder mehrere Teile enthält, die dynamisch ausgewertet werden müssen. Angenommen ein Benutzer gibt seinen Request für das Sortierkriterium ein und das Sortierkriterium ist im Attribut `UserReq/@sortkey` gespeichert. Im Stylesheet könnten Sie den folgenden Ausdruck haben: `<xsl:sort select="altova:evaluate(../UserReq/@sortkey)" order="ascending"/>` Die `altova:evaluate()` Funktion liest das `sortkey` Attribut des `UserReq` Child-Elements des Parent des Kontext-Node. Angenommen der Wert des `sortkey` Attributs ist `Price`, dann wird von der `altova:evaluate()` Funktion `Price` zurückgegeben und wird zum Wert des `select` Attributs: `<xsl:sort select="Price" order="ascending"/>`. Wenn diese `sort` Anweisung im Kontext eines Elements namens `Order` vorkommt, dann werden die `Order` Elemente nach den Werten Ihrer `Price` Children sortiert. Alternativ dazu, wenn der Wert von `@sortkey` z.B. `Date` ist, werden die `Order` Elemente nach den Werten ihrer `Date` Children sortiert. Das Sortierkriterium für `Order` wird also zur Laufzeit aus dem `sortkey` Attribut ausgewählt. Diese hätte man mit einem Ausdruck wie dem folgenden nicht bewerkstelligen können: `<xsl:sort select="../UserReq/@sortkey" order="ascending"/>`. Im oben gezeigten Beispiel wäre das Sortierkriterium das `sortkey` Attribut selbst, nicht `Price` oder `Date` (oder jeder beliebige andere Inhalt von `sortkey`)

Hinweis:
Der statische Kontext enthält Namespaces, Typen und Funktionen - aber keine Variablen - aus der aufrufenden Umgebung. Die Basis-URI und der Standard-Namespace werden vererbt.

Weitere Beispiele

- Statische Variablen: `<xsl:value-of select="$i3, $i2, $i1" />`
  Gibt die Werte von drei Variablen aus.

- Dynamischer XPath-Ausdruck mit dynamischen Variablen:
  `<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />`
  `<xsl:value-of select="altova:evaluate($xpath, 10, 20, 30)" />`
  Gibt "`30 20 10`*" aus*

- Dynamischer XPath-Ausdruck ohne dynamische Variable:
  ```
  <xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
  <xsl:value-of select="altova:evaluate($xpath)" />
  ```
  Gibt einen Fehler aus.: Es wurde keine Variable für `$p3` *definiert.*

▼ encode-for-rtf [altova:]

altova:encode-for-rtf(input *als xs:string*, **preserveallwhitespace** *als xs:boolean*, **preservenewlines** *als xs:boolean*) **als xs:string** XSLT2 XSLT3
Konvertiert den Input-String in Code für RTF. Whitespaces und neue Zeilen werden gemäß dem für die entsprechenden Parameter definierten Booleschen Wert beibehalten.

[Nach oben 2145]

XBRL-Funktionen

Altova XBRL-Funktionen können nur mit Editionen von Altova-Produkten verwendet werden, die XBRL unterstützen.

▼ xbrl-footnotes [altova:]

altova:xbrl-footnotes(*node()***)** als **node()*** XSLT2 XSLT3
Erhält einen Node als Input-Argument und gibt die durch den Input-Node referenzierte Gruppe der XBRL-Fußnoten-Nodes zurück.

▼ xbrl-labels [altova:]

altova:xbrl-labels(*xs:QName*, *xs:string***)** als **node()*** XSLT2 XSLT3
Erhält zwei Input-Argumente: einen Node-Namen und den Pfad der Taxonomiedatei, die den Node enthält. Die Funktion gibt die XBRL Labels zurück, die mit dem Input-Node verknüpft sind.

[Nach oben 2145]

30.2.1.2 XPath/XQuery-Funktionen: Datum und Uhrzeit

Die Datums- und Uhrzeit-Erweiterungsfunktionen von Altova können im Zusammenhang mit XPath- und XQuery-Ausdrücken verwendet werden und bieten zusätzliche Funktionalitäten für die Verarbeitung von Daten, die in Form von XML-Schema-Datums- und Uhrzeit-Datentypen zur Verfügung stehen. Die Funktionen in diesem Abschnitt können mit dem **XPath 3.0** - und dem **XQuery 3.0** -Prozessor von Altova verwendet werden. Sie stehen in verschiedenen XPath/XQuery-Kontexten zur Verfügung.

Anmerkung zur Benennung von Funktionen und zur Anwendbarkeit der Sprache

Altova-Erweiterungsfunktionen können in XPath/XQuery-Ausdrücken verwendet werden. Dadurch stehen neben den Funktionen in der Standardbibliothek der XPath-, XQuery- und XSLT-Funktionen zusätzliche Funktionen zur Verfügung. Die Altova-Erweiterungsfunktionen befinden sich im **Altova-Erweiterungsfunktions-Namespace,** `http://www.altova.com/xslt-extensions` und sind in diesem Abschnitt mit dem Präfix `altova:`, das als an diesen Namespace gebunden angenommen wird, gekennzeichnet. Beachten Sie, dass manche Funktionen in zukünftigen Versionen Ihres Produkts eventuell nicht mehr unterstützt werden oder dass sich das Verhalten einzelner Funktionen ändern kann. Um zu sehen, welche Altova-Erweiterungsfunktionen unterstützt werden, lesen Sie bitte die Dokumentation zur jeweiligen Release.

XPath-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XP1 XP2 XP3.1
XSLT-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XSLT1 XSLT2 XSLT3
XQuery-Funktionen (in XQuery-Ausdrücken in XQuery verwendet):	XQ1 XQ3.1

▼ Nach Funktionalität gruppiert

- Hinzufügen einer Zeitdauer zu xs:dateTime und Rückgabe von xs:dateTime 2150
- Hinzufügen einer Zeitdauer zu xs:date und Rückgabe von xs:date 2152
- Hinzufügen einer Zeitdauer zu xs:time und Rückgabe von xs:time 2153
- Formatieren und Abrufen einer Zeitdauer 2153
- Entfernen der Zeitzone aus Funktionen, die das aktuelle Datum/die aktuelle Uhrzeit generieren 2154
- Rückgabe von Tagen, Stunden, Minuten und Sekunden anhand einer Zeitdauer 2156
- Rückgabe des Wochentags anhand des Datums als Ganzzahl 2158
- Rückgabe eines Wochentags als Ganzzahl anhand eines Datums 2158
- Erstellen des Datums, der Uhrzeit oder des Zeitdauertyps anhand der lexikalischen Komponenten der einzelnen Typen 2160
- Konstruieren des Typs "Datum", "Datum und Uhrzeit" oder "Uhrzeit" anhand eines String Input 2162
- Funktionen zur Berechnung des Alters 2163
- Epochen-Zeit (Unix-Zeit)-Funktionen 2165

▼ in alphabetischer Reihenfolge

altova:add-days-to-date 2152
altova:add-days-to-dateTime 2150
altova:add-hours-to-dateTime 2150
altova:add-hours-to-time 2153
altova:add-minutes-to-dateTime 2150
altova:add-minutes-to-time 2153
altova:add-months-to-date 2152
altova:add-months-to-dateTime 2150
altova:add-seconds-to-dateTime 2150
altova:add-seconds-to-time 2153
altova:add-years-to-date 2152
altova:add-years-to-dateTime 2150
altova:age 2163
altova:age-details 2163
altova:build-date 2160
altova:build-duration 2160
altova:build-time 2160
altova:current-dateTime-no-TZ 2154
altova:current-date-no-TZ 2154
altova:current-time-no-TZ 2154

altova:date-no-TZ 2154
altova:dateTime-from-epoch 2165
altova:dateTime-from-epoch-no-TZ 2165
altova:dateTime-no-TZ 2154
altova:days-in-month 2156
altova:epoch-from-dateTime 2165
altova:hours-from-dateTimeDuration-accumulated 2156
altova:minutes-from-dateTimeDuration-accumulated 2156
altova:seconds-from-dateTimeDuration-accumulated 2156
altova:format-duration 2153
altova:parse-date 2162
altova:parse-dateTime 2162
altova:parse-duration 2153
altova:parse-time 2162
altova:time-no-TZ 2154
altova:weekday-from-date 2158
altova:weekday-from-dateTime 2158
altova:weeknumber-from-date 2159
altova:weeknumber-from-dateTime 2159

[Nach oben 2148]

Hinzufügen einer Zeitdauer zu xs:dateTime XP3.1 XQ3.1

Mit diesen Funktionen werden Zeitdauerwerte zu **xs:dateTime** hinzugefügt, bevor **xs:dateTime** zurückgegeben wird. Der Typ `xs:dateTime` hat das Format `JJJJ-MM-TTZhh:mm:ss.sss`. Es handelt sich hierbei um eine Verkettung des `xs:date` und `xs:time` Formats, getrennt durch den Buchstaben `Z`. Ein Zeitzonensuffix (wie z.B. `+01:00`) ist optional.

add-years-to-dateTime [altova:]

altova:add-years-to-dateTime(DateTime *als xs:dateTime*, **Years** *als xs:integer***) als xs:dateTime** XP3.1 XQ3.1

Fügt eine Zeitdauer in Jahren zu einem `xs:dateTime` Wert (*siehe Beispiele unten*) hinzu. Beim zweiten Argument handelt es sich um die Anzahl der Jahre, die zu dem im ersten Parameter angegebenen `xs:dateTime` Wert hinzugefügt werden sollen. Das Ergebnis ist vom Typ `xs:dateTime`.

Beispiele

- **altova:add-years-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) gibt 2024-01-15T14:00:00 zurück
- **altova:add-years-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -4) gibt 2010-01-15T14:00:00 zurück

add-months-to-dateTime [altova:]

altova:add-months-to-dateTime(DateTime *als xs:dateTime*, **Months** *als xs:integer***) als xs:dateTime** XP3.1 XQ3.1

Fügt eine Zeitdauer in Monaten zu einem `xs:dateTime` Wert (*siehe Beispiele unten*) hinzu. Beim zweiten Argument handelt es sich um die Anzahl der Monate, die zu dem im ersten Argument angegebenen `xs:dateTime` Wert hinzugefügt werden sollen. Das Ergebnis ist vom Typ `xs:dateTime`.

Beispiele

© 2016-2022 Altova GmbH

- **altova:add-months-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) gibt 2014-11-15T14:00:00 zurück
- **altova:add-months-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -2) gibt 2013-11-15T14:00:00 zurück

add-days-to-dateTime [altova:]

altova:add-days-to-dateTime(DateTime *als xs:dateTime*, **Days** *als xs:integer***)** als **xs:dateTime** **XP3.1** **XQ3.1**

Fügt eine Zeitdauer in Tagen zu einem xs:dateTime Wert (*siehe Beispiel unten*) hinzu. Beim zweiten Argument handelt es sich um die Anzahl der Tage, die zu dem im ersten Argument angegebenen xs:dateTime Wert hinzugefügt werden sollen. Das Ergebnis ist vom Typ xs:dateTime.

Beispiele

- **altova:add-days-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) gibt 2014-01-25T14:00:00 zurück
- **altova:add-days-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -8) gibt 2014-01-07T14:00:00 zurück

add-hours-to-dateTime [altova:]

altova:add-hours-to-dateTime(DateTime *als xs:dateTime*, **Hours** *als xs:integer***)** als **xs:dateTime** **XP3.1** **XQ3.1**

Fügt eine Zeitdauer in Stunden zu einem xs:dateTime Wert (*siehe Beispiel unten*) hinzu. Beim zweiten Argument handelt es sich um die Anzahl der Stunden, die zu dem im ersten Argument angegebenen xs:dateTime Wert hinzugefügt werden sollen. Das Ergebnis ist vom Typ xs:dateTime.

Beispiele

- **altova:add-hours-to-dateTime**(xs:dateTime("2014-01-15T13:00:00"), 10) gibt 2014-01-15T23:00:00 zurück
- **altova:add-hours-to-dateTime**(xs:dateTime("2014-01-15T13:00:00"), -8) gibt 2014-01-15T05:00:00 zurück

add-minutes-to-dateTime [altova:]

altova:add-minutes-to-dateTime(DateTime *als xs:dateTime*, **Minutes** *als xs:integer***)** als **xs:dateTime** **XP3.1** **XQ3.1**

Fügt eine Zeitdauer in Minuten zu einem xs:dateTime Wert (*siehe Beispiele unten)* hinzu. Beim zweiten Argument handelt es sich um die Anzahl der Minuten, die zu dem im ersten Argument angegebenen xs:dateTime Wert hinzugefügt werden sollen. Das Ergebnis ist vom Typ xs:dateTime.

Beispiele

- **altova:add-minutes-to-dateTime**(xs:dateTime("2014-01-15T14:10:00"), 45) gibt 2014-01-15T14:55:00 zurück
- **altova:add-minutes-to-dateTime**(xs:dateTime("2014-01-15T14:10:00"), -5) gibt 2014-01-15T14:05:00 zurück

add-seconds-to-dateTime [altova:]

altova:add-seconds-to-dateTime(DateTime *als xs:dateTime*, **Seconds** *als xs:integer*) als **xs:dateTime** **XP3.1** **XQ3.1**
Fügt eine Zeitdauer in Sekunden zu einem `xs:dateTime` Wert (*siehe Beispiele unten*) hinzu. Beim zweiten Argument handelt es sich um die Anzahl der Sekunden, die zu dem im ersten Argument angegebenen `xs:dateTime` Wert hinzugefügt werden sollen. Das Ergebnis ist vom Typ `xs:dateTime`.

Beispiele

- **altova:add-seconds-to-dateTime**(xs:dateTime("2014-01-15T14:00:10"), 20) gibt 2014-01-15T14:00:30 zurück
- **altova:add-seconds-to-dateTime**(xs:dateTime("2014-01-15T14:00:10"), -5) gibt 2014-01-15T14:00:05 zurück

[Nach oben 2148]

Hinzufügen einer Zeitdauer zu xs:date **XP3.1** **XQ3.1**

Mit diesen Funktionen werden Zeitdauerwerte zu **xs:date** hinzugefügt, bevor **xs:date** zurückgegeben wird. Der Typ `xs:date` hat das Format `JJJJ-MM-TT`.

add-years-to-date [altova:]

altova:add-years-to-date(Date *als xs:date*, **Years** *als xs:integer*) als **xs:date** **XP3.1** **XQ3.1**
Fügt eine Zeitdauer in Jahren zu einem Datumswert hinzu. Beim zweiten Argument handelt es sich um die Anzahl der Jahre, die zu dem im ersten Argument angegebenen `xs:date` Wert hinzugefügt werden sollen. Das Ergebnis ist vom Typ `xs:date`.

Beispiele

- **altova:add-years-to-date**(xs:date("2014-01-15"), 10) gibt 2024-01-15 zurück
- **altova:add-years-to-date**(xs:date("2014-01-15"), -4) gibt 2010-01-15 zurück

add-months-to-date [altova:]

altova:add-months-to-date(Date *als xs:date*, **Months** *als xs:integer*) als **xs:date** **XP3.1** **XQ3.1**
Fügt eine Zeitdauer in Monaten zu einem Datumswert hinzu. Beim zweiten Argument handelt es sich um die Anzahl der Monate, die zu dem im ersten Argument angegebenen `xs:date` Wert hinzugefügt werden sollen. Das Ergebnis ist vom Typ `xs:date`.

Beispiele

- **altova:add-months-to-date**(xs:date("2014-01-15"), 10) gibt 2014-11-15 zurück
- **altova:add-months-to-date**(xs:date("2014-01-15"), -2) gibt 2013-11-15 zurück

add-days-to-date [altova:]

altova:add-days-to-date(Date *als xs:date*, **Days** *als xs:integer*) als **xs:date** **XP3.1** **XQ3.1**
Fügt eine Zeitdauer in Tagen zu einem Datumswert hinzu. Beim zweiten Argument handelt es sich um die Anzahl der Tage, die zu dem im ersten Argument angegebenen `xs:date` Wert hinzugefügt werden sollen. Das Ergebnis ist vom Typ `xs:date`.

Beispiele

© 2016-2022 Altova GmbH

- **altova:add-days-to-date**(xs:date("2014-01-15"), 10) gibt 2014-01-25 zurück
- **altova:add-days-to-date**(xs:date("2014-01-15"), -8) gibt 2014-01-07 zurück

[Nach oben 2148]

Formatieren und Abrufen einer Zeitdauer XP3.1 XQ3.1

Mit diesen Funktionen wird ein Input **xs:duration**- oder **xs:string**-Wert geparst und ein **xs:string**- bzw. **xs:duration**-Wert zurückgegeben.

format-duration [altova:]

altova:format-duration(Duration *als xs:duration*, **Picture** *als xs:string*) als **xs:string**
XP3.1 XQ3.1
Formatiert eine Zeitdauer, die als erstes Argument bereitgestellt wird, gemäß einem Muster-String, der als zweites Argument bereitgestellt wird. Die Ausgabe ist ein Textstring, der dem Muster-String entsprechend formatiert ist.

Beispiele

- **altova:format-duration**(xs:duration("P2DT2H53M11.7S"), "Days:[D01] Hours:[H01] Minutes:[m01] Seconds:[s01] Fractions:[f0]") gibt "Days:02 Hours:02 Minutes:53 Seconds:11 Fractions:7" gibt
- **altova:format-duration**(xs:duration("P3M2DT2H53M11.7S"), "Months:[M01] Days:[D01] Hours:[H01] Minutes:[m01]") gibt "Months:03 Days:02 Hours:02 Minutes:53" gibt

parse-duration [altova:]

altova:parse-duration(InputString *als xs:string*, **Picture** *als xs:string*) als **xs:duration**
XP3.1 XQ3.1
Erhält einen Pattern-String als erstes Argument und eine Muster-String als zweites Argument. Der Input-Strings wird auf Basis des Muster-Strings geparst und ein xs:duration wird zurückgegeben.

Beispiele

- **altova:parse-duration**("Days:02 Hours:02 Minutes:53 Seconds:11 Fractions:7"), "Days:[D01] Hours:[H01] Minutes:[m01] Seconds:[s01] Fractions:[f0]") gibt "P2DT2H53M11.7S" zurück
- **altova:parse-duration**("Months:03 Days:02 Hours:02 Minutes:53 Seconds:11 Fractions:7", "Months:[M01] Days:[D01] Hours:[H01] Minutes:[m01]") gibt "P3M2DT2H53M" zurück

[Nach oben 2148]

Hinzufügen einer Zeitdauer zu xs:time XP3.1 XQ3.1

Diese Funktionen fügen einen Zeitdauerwert zu **xs:time** hinzu und geben **xs:time** zurück. Der Typ xs:time entspricht in seiner lexikalischen Form hh:mm:ss.sss. Eine optionale Zeitzone kann angehängt werden. Der Buchstabe Z steht für Coordinated Universal Time (UTC). Alle anderen Zeitzonen werden in Form des

Zeitunterschieds zur UTC im Format `+hh:mm`, oder `-hh:mm` dargestellt. Wenn kein Wert für die Zeitzone vorhanden ist, wird sie als unbekannt und nicht als UTC angenommen.

▼ add-hours-to-time [altova:]

`altova:add-hours-to-time(Time` *`als xs:time`*`, **Hours** `*`als xs:integer`*`)` als **`xs:time`** **XP3.1** **XQ3.1**
Fügt eine Zeitdauer in Stunden zu einem Uhrzeitwert hinzu. Beim zweiten Argument handelt es sich um die Anzahl der Stunden, die zu dem im ersten Argument angegebenen `xs:time` Wert hinzugefügt werden sollen. Das Ergebnis ist vom Typ `xs:time`.

Beispiele

- **`altova:add-hours-to-time`**`(xs:time("11:00:00"), 10)` gibt `21:00:00` zurück
- **`altova:add-hours-to-time`**`(xs:time("11:00:00"), -7)` gibt `04:00:00` zurück

▼ add-minutes-to-time [altova:]

`altova:add-minutes-to-time(Time` *`als xs:time`*`, **Minutes** `*`als xs:integer`*`)` als **`xs:time`** **XP3.1** **XQ3.1**
Fügt eine Zeitdauer in Minuten zu einem `xs:time` Wert hinzu. Beim zweiten Argument handelt es sich um die Anzahl der Minuten, die zu dem im ersten Argument angegebenen `xs:time` Wert hinzugefügt werden sollen. Das Ergebnis ist vom Typ `xs:time`.

Beispiele

- **`altova:add-minutes-to-time`**`(xs:time("14:10:00"), 45)` gibt `14:55:00` zurück
- **`altova:add-minutes-to-time`**`(xs:time("14:10:00"), -5)` gibt `14:05:00` zurück

▼ add-seconds-to-time [altova:]

`altova:add-seconds-to-time(Time` *`as xs:time`*`, **Minutes** `*`als xs:integer`*`)` als **`xs:time`** **XP3.1** **XQ3.1**
Fügt eine Zeitdauer in Sekunden zu einem Uhrzeitwert hinzu. Beim zweiten Argument handelt es sich um die Anzahl der Sekunden, die zu dem im ersten Argument angegebenen `xs:time` Wert hinzugefügt werden sollen. Das Ergebnis ist vom Typ `xs:time`. Die Seconds Komponenten kann sich im Bereich von `0` bis `59.999` befinden.

Beispiele

- **`altova:add-seconds-to-time`**`(xs:time("14:00:00"), 20)` gibt `14:00:20` zurück
- **`altova:add-seconds-to-time`**`(xs:time("14:00:00"), 20.895)` gibt `14:00:20.895` zurück

[Nach oben 2148]

Entfernen der Zeitzone aus date/time-Datentypen **XP3.1** **XQ3.1**

Diese Funktionen entfernen die Zeitzone aus den aktuellen **`xs:dateTime`**, **`xs:date`** bzw. **`xs:time`** Werten. Beachten Sie, dass im Unterschied zu `xs:dateTime` bei `xs:dateTimeStamp` die Zeitzone erforderlich ist (während sie im ersteren Fall optional ist). Das Format eines `xs:dateTimeStamp` Werts lautet daher: `JJJJ-MM-TTZhh:mm:ss.sss±hh:mm`. oder `JJJJ-MM-TTZhh:mm:ss.sssZ`. Wenn das Datum und die Uhrzeit von der Systemuhr als `xs:dateTimeStamp` ausgelesen wird, können Sie die Zeitzone, falls erforderlich, mit der Funktion `current-dateTime-no-TZ()` entfernen.

© 2016-2022 Altova GmbH

▼ current-date-no-TZ [altova:]

altova:current-date-no-TZ() als **xs:date** **XP3.1** **XQ3.1**
Die Funktion hat kein Argument. Sie entfernt die Zeitzone aus dem `current-date()` Wert (welcher das aktuelle Datum laut Systemuhr ist) und gibt einen `xs:date` Wert zurück.

▣ *Beispiele*

Wenn das aktuelle Datum **2014-01-15+01:00** lautet:

- **altova:current-date-no-TZ**() gibt 2014-01-15 zurück

▼ current-dateTime-no-TZ [altova:]

altova:current-dateTime-no-TZ() als **xs:dateTime** **XP3.1** **XQ3.1**
Die Funktion hat kein Argument. Sie entfernt die Zeitzone aus dem `current-dateTime()` Wert (welcher das aktuelle Datum und die aktuelle Uhrzeit laut Systemuhr ist) und gibt einen `xs:dateTime` Wert zurück.

▣ *Beispiele*

Wenn der aktuelle Datums- und Uhrzeitwert **2014-01-15T14:00:00+01:00** lautet:

- **altova:current-dateTime-no-TZ**() gibt 2014-01-15T14:00:00 zurück

▼ current-time-no-TZ [altova:]

altova:current-time-no-TZ() als **xs:time** **XP3.1** **XQ3.1**
Die Funktion hat kein Argument. Sie entfernt die Zeitzone aus dem `current-time()` Wert (welcher die aktuelle Uhrzeit laut Systemuhr ist) und gibt einen `xs:time` Wert zurück.

▣ *Beispiele*

Wenn der aktuelle Uhrzeitwert **14:00:00+01:00** lautet:

- **altova:current-time-no-TZ**() gibt 14:00:00 zurück

▼ date-no-TZ [altova:]

altova:date-no-TZ(InputDate *as xs:date*) als **xs:date** **XP3.1** **XQ3.1**
Diese Funktion verwendet ein `xs:date` Argument, entfernt den Zeitzonenteil daraus und gibt einen `xs:date` Wert zurück. Beachten Sie, dass das Datum nicht geändert wird..

▣ *Beispiele*

- **altova:date-no-TZ**(xs:date("2014-01-15+01:00")) gibt 2014-01-15 zurück

▼ dateTime-no-TZ [altova:]

altova:dateTime-no-TZ(InputDateTime *als xs:dateTime*) als **xs:dateTime** **XP3.1** **XQ3.1**
Diese Funktion verwendet ein `xs:dateTime` Argument, entfernt den Zeitzonenteil daraus und gibt einen `xs:dateTime` Wert zurück. Beachten Sie, dass weder Datum noch Uhrzeit geändert werden.

▣ *Beispiele*

- **altova:dateTime-no-TZ**(xs:date("2014-01-15T14:00:00+01:00")) gibt 2014-01-15T14:00:00 zurück

▼ time-no-TZ [altova:]

altova:time-no-TZ(InputTime *als xs:time***)** als **xs:time** **XP3.1** **XQ3.1**
Diese Funktion verwendet ein `xs:time` Argument, entfernt den Zeitzonenteil daraus und gibt einen `xs:time` Wert zurück. Beachten Sie, dass die Uhrzeit nicht geändert wird.

▣ *Beispiele*

- **altova:time-no-TZ**`(xs:time("14:00:00+01:00"))` gibt `14:00:00` zurück

[Nach oben 2148]

Rückgabe der Anzahl von Tagen, Stunden, Minuten, Sekunden anhand einer Zeitdauer **XP3.1** **XQ3.1**

Diese Funktionen geben die Anzahl der Tage in einem Monat bzw. die Anzahl der Stunden, Minuten und Sekunden anhand einer Zeitdauer zurück.

▼ days-in-month [altova:]

altova:days-in-month(Year *als xs:integer*, **Month** *als xs:integer***)** als **xs:integer** **XP3.1** **XQ3.1**
Gibt die Anzahl der Tage im angegebenen Monat zurück. Der Monat wird mit Hilfe der Argumente `Year` und `Month` angegeben.

▣ *Beispiele*

- **altova:days-in-month**`(2018, 10)` gibt `31` zurück
- **altova:days-in-month**`(2018, 2)` gibt `28` zurück
- **altova:days-in-month**`(2020, 2)` gibt `29` zurück

▼ hours-from-dayTimeDuration-accumulated

altova:hours-from-dayTimeDuration-accumulated(DayAndTime *als xs:duration***)** als **xs:integer** **XP3.1** **XQ3.1**
Gibt die Gesamtzahl der Stunden in der durch das Argument `DayAndTime` bereitgestellten Zeitdauer (vom Typ `xs:duration`) zurück. Die Stunden in den Komponenten `Day` und `Time` werden addiert, um ein Ergebnis in Form einer Ganzzahl zu erhalten. Nur für volle 60 Minuten wird eine neue Stunde berechnet. Negative Zeitdauerergebnisse ergeben einen negativen Stundenwert.

▣ *Beispiele*

- **altova:hours-from-dayTimeDuration-accumulated**`(xs:duration("P5D"))` gibt `120` zurück, d.h. die Gesamtanzahl der Stunden in 5 Tagen.
- **altova:hours-from-dayTimeDuration-accumulated**`(xs:duration("P5DT2H"))` gibt `122` zurück, d.h. die Gesamtzahl der Stunden in 5 Tagen plus 2 Stunden.
- **altova:hours-from-dayTimeDuration-accumulated**`(xs:duration("P5DT2H60M"))` gibt `123` zurück, d.h. die Gesamtzahl der Stunden in 5 Tagen plus 2 Stunden und 60 Minuten.
- **altova:hours-from-dayTimeDuration-accumulated**`(xs:duration("P5DT2H119M"))` gibt `123` zurück, d.h. die Gesamtzahl der Stunden in 5 Tagen plus 2 Stunden und 119 Minuten.
- **altova:hours-from-dayTimeDuration-accumulated**`(xs:duration("P5DT2H120M"))` gibt `124` zurück, d.h. die Gesamtzahl der Stunden in 5 Tagen plus 2 Stunden und 120 Minuten.
- **altova:hours-from-dayTimeDuration-accumulated**`(xs:duration("-P5DT2H"))` gibt `-122`

© 2016-2022 Altova GmbH

zurück.

minutes-from-dayTimeDuration-accumulated

altova:minutes-from-dayTimeDuration-accumulated(DayAndTime *als xs:duration*) als **xs:integer** **XP3.1** **XQ3.1**

Gibt die Gesamtzahl der Minuten in der durch das Argument `DayAndTime` bereitgestellten Zeitdauer (vom Typ `xs:duration`) zurück. Die Minuten in den Komponenten `Day` und `Time` werden addiert, um ein Ergebnis in Form einer Ganzzahl zu erhalten. Negative Zeitdauerergebnisse ergeben einen negativen Minutenwert.

Beispiele

- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("PT60M")) gibt 60 zurück.
- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("PT1H")) gibt 60 zurück, d.h. die Gesamtzahl der Minuten in 1 Stunde.
- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("PT1H40M") gibt 100 zurück.
- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("P1D")) gibt 1440 zurück, d.h. die Gesamtzahl der Minuten an einem Tag.
- **altova:minutes-from-dayTimeDuration-accumulated**(xs:duration("-P1DT60M")) gibt -1500 zurück.

seconds-from-dayTimeDuration-accumulated

altova:seconds-from-dayTimeDuration-accumulated(DayAndTime *als xs:duration*) als **xs:integer** **XP3.1** **XQ3.1**

Gibt die Gesamtzahl der Sekunden in der durch das Argument `DayAndTime` bereitgestellten Zeitdauer (vom Typ `xs:duration`) zurück. Die Sekunden in den Komponenten `Day` und `Time` werden addiert, um ein Ergebnis in Form einer Ganzzahl zu erhalten. Negative Zeitdauerergebnisse ergeben einen negativen Sekundenwert.

Examples

- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("PT1M")) gibt 60 zurück, d.h. die Gesamtzahl der Sekunden in 1 Minute.
- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("PT1H")) gibt 3600 zurück, d.h. die Gesamtzahl der Sekunden in 1 Stunde.
- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("PT1H2M")) gibt 3720 zurück.
- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("P1D"))gibt 86400 zurück, d.h. die Gesamtzahl der Sekunden an 1 Tag.
- **altova:seconds-from-dayTimeDuration-accumulated**(xs:duration("-P1DT1M")) gibt -86460 zurück.

Rückgabe des Wochentages anhand von xs:dateTime oder xs:date XP3.1 XQ3.1

Diese Funktionen geben anhand des `xs:dateTime` oder `xs:date` Werts den Wochentag in Form einer Ganzzahl zurück. Die Tage der Woche sind (im amerikanischen Format) von `1` bis `7` nummeriert, wobei `Sonntag=1`. Im europäischen Format beginnt die Woche am Montag (`=1`). Das amerikanische Format, in dem `Sonntag=1`, kann mittels der Ganzzahl `0` definiert werden, wenn das Format mittels einer Ganzzahl angegeben werden kann.

▼ weekday-from-dateTime [altova:]

altova:weekday-from-dateTime(DateTime *als xs:dateTime*) als **xs:integer** XP3.1 XQ3.1
Erhält ein Datum mit einer Uhrzeit als einziges Argument und gibt den Tag der Woche dieses Datums in Form einer Ganzzahl zurück. Die Wochentage sind beginnend mit `Sonntag=1` nummeriert. Wenn das europäische Format benötigt wird (wo `Montag=1`), verwenden Sie die andere Signatur dieser Funktion (*siehe nächste Signatur unten*).

Beispiele

- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00")) gibt `2` zurück, wobei `2` für Montag steht.

altova:weekday-from-dateTime(DateTime *als xs:dateTime*, **Format** *als xs:integer*) als **xs:integer** XP3.1 XQ3.1
Erhält ein Datum mit einer Uhrzeit als erstes Argument und gibt den Tag der Woche dieses Datums in Form einer Ganzzahl zurück. Die Wochentage sind beginnend mit `Montag=1` nummeriert. Wenn das zweite (Integer)-Argument `0` ist, werden die Wochentage beginnend mit `Sonntag=1` von `1` bis `7` nummeriert. Wenn das zweite Argument eine andere Ganzzahl als `0` ist, so ist `Montag=1`. Wenn es kein zweites Argument gibt, wird die Funktion gelesen, als ob sie die andere Signatur dieser Funktion hätte (*siehe vorherige Signatur).*

Beispiele

- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 1) gibt `1`, zurück, wobei `1` für Montag steht
- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 4) gibt `1`, zurück, wobei `1` für Montag steht
- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 0) gibt `2`, zurück, wobei `2` für Montag steht

▼ weekday-from-date [altova:]

altova:weekday-from-date(Date *als xs:date*) als **xs:integer** XP3.1 XQ3.1
Erhält ein Datum als einziges Argument und gibt den Tag der Woche dieses Datums in Form einer Ganzzahl zurück. Die Wochentage sind beginnend mit `Sonntag=1` nummeriert. Wenn das europäische Format benötigt wird (wo `Montag=1`), verwenden Sie die andere Signatur dieser Funktion (*siehe nächste Signatur unten*).

Beispiele

- **altova:weekday-from-date**(xs:date("2014-02-03+01:00")) gibt `2` zurück, wobei `2` für Montag steht.

altova:weekday-from-date(Date *als xs:date*, **Format** *als xs:integer*) als **xs:integer** XP3.1 XQ3.1
Erhält ein Datum als erstes Argument und gibt den Tag der Woche dieses Datums in Form einer

© 2016-2022 Altova GmbH

Ganzzahl zurück. Die Wochentage sind beginnend mit `Montag=1` nummeriert. Wenn das zweite Argument (`Format`) `0` ist, werden die Wochentage beginnend mit `Sonntag=1` von `1` bis `7` nummeriert. Wenn das zweite Argument eine andere Ganzzahl als `0` ist, so ist `Montag=1`. Wenn es kein zweites Argument gibt, wird die Funktion gelesen, als ob sie die andere Signatur dieser Funktion hätte (*siehe vorherige Signatur).*

Beispiele

- **altova:weekday-from-date**(xs:date("2014-02-03"), 1) gibt 1, zurück, wobei 1 für Montag steht
- **altova:weekday-from-date**(xs:date("2014-02-03"), 4) gibt 1, zurück, wobei 1 für Montag steht
- **altova:weekday-from-date**(xs:date("2014-02-03"), 0) gibt 2, zurück, wobei 2 für Montag steht

[Nach oben 2148]

Rückgabe der Wochennummer anhand von xs:dateTime oder xs:date XP2 XQ1 XP3.1 XQ3.1

Diese Funktionen geben anhand von `xs:dateTime` oder `xs:date` die Wochennummer als Ganzzahl zurück. Die Wochennummer steht in den Kalenderformaten US, ISO/European und Islamic zur Verfügung. Die Wochennummerierung unterscheidet sich in diesen Kalenderformaten, da die Woche in diesen Formaten an unterschiedlichen Tagen beginnt (Im Format US am Sonntag, im Format ISO/European am Montag und im Format Islamic am Samstag).

weeknumber-from-date [altova:]

altova:weeknumber-from-date(Date *als xs:date*, **Calendar** *als xs:integer*) als **xs:integer** XP2 XQ1 XP3.1 XQ3.1

Gibt die Wochennummer des bereitgestellten **Date** Arguments als Ganzzahl zurück. Das zweite Argument (**Calendar**) definiert das zu verwendende Kalendersystem.
Unterstützte **Calendar** Werte sind:

- **0 = US-Kalender** (Woche beginnt am Sonntag)
- **1 = ISO-Standard, Europäischer Kalender** (Woche beginnt am Montag)
- **2 = Islamischer Kalender** (Woche beginnt am Samstag)

Der Standardwert ist `0`.

Beispiele

- **altova:weeknumber-from-date**(xs:date("2014-03-23"), 0) gibt 13 zurück
- **altova:weeknumber-from-date**(xs:date("2014-03-23"), 1) gibt 12 zurück
- **altova:weeknumber-from-date**(xs:date("2014-03-23"), 2) gibt 13 zurück
- **altova:weeknumber-from-date**(xs:date("2014-03-23")) gibt 13 zurück

Der Tag des Datums in den obigen Beispielen (`2014-03-23`) ist ein Sonntag. Daher ist der US- und der islamische Kalender dem europäischen Kalender an diesem Tag eine Woche voraus.

weeknumber-from-dateTime [altova:]

altova:weeknumber-from-dateTime(DateTime *als xs:dateTime*, **Calendar** *als xs:integer*) **als xs:integer** XP2 XQ1 XP3.1 XQ3.1

Gibt die Wochennummer des bereitgestellten **DateTime** Arguments als Ganzzahl zurück. Das zweite Argument (**Calendar**) definiert das zu verwendende Kalendersystem.
Unterstützte **Calendar** Werte sind:

- **0 = US-Kalender** (*Woche beginnt am Sonntag*)
- **1 = ISO-Standard, Europäischer Kalender** (*Woche beginnt am Montag*)
- **2 = Islamischer Kalender** (*Woche beginnt am Samstag*)

Der Standardwert ist 0.

Beispiele

- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 0) gibt 13 zurück
- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 1) gibt 12 zurück
- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 2) gibt 13 zurück
- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00")) gibt 13 zurück

Der Tag des Datums- und Uhrzeitwerts in den obigen Beispielen (2014-03-23T00:00:00) ist ein Sonntag. Daher ist der US- und der islamische Kalender dem europäischen Kalender an diesem Tag eine Woche voraus.

[Nach oben 2148]

Erstellen des Datums-, Uhrzeit- oder Zeitdauer-Datentyps anhand der lexikalischen Komponenten der einzelnen Typen XP3.1 XQ3.1

Die Funktionen erhalten die lexikalischen Komponenten des xs:date, xs:time oder xs:duration-Datentyps als Input-Argumente und kombinieren diese zum entsprechenden Datentyp.

build-date [altova:]

altova:build-date(Year *als xs:integer*, **Month** *als xs:integer*, **Date** *als xs:integer*) als **xs:date** XP3.1 XQ3.1

Das erste, zweite und dritte Argument steht für das Jahr, bzw. den Monat bzw. das Datum. Sie werden zu einem Wert vom Typ xs:date kombiniert. Die Werte der Ganzzahlen müssen sich innerhalb des korrekten Bereichs dieses jeweiligen Datumsteils befinden. So sollte z.B. das zweite Argument nicht größer als 12 sein.

Beispiele

- **altova:build-date**(2014, 2, 03) gibt 2014-02-03 zurück

build-time [altova:]

altova:build-time(Hours *als xs:integer*, **Minutes** *als xs:integer*, **Seconds** *als xs:integer*)

© 2016-2022 Altova GmbH

als **xs:time** **XP3.1** **XQ3.1**
Das erste, zweite und dritte Argument steht für die Stunde (0 bis 23), bzw. die Minuten (0 bis 59) bzw. die Sekunden (0 bis 59). Sie werden zu einem Wert von Typ xs:time kombiniert. Die Werte der Ganzzahlen müssen sich innerhalb des korrekten Bereichs dieses jeweiligen Uhrzeitteils befinden. So sollte z.B. der zweite Parameter nicht größer als 59 sein. Um eine Zeitzone zum Wert hinzuzufügen, verwenden Sie die andere Signatur der Funktion (*siehe nächste Signatur*).

- *Beispiele*
  - **altova:build-time**(23, 4, 57) gibt 23:04:57 zurück

altova:build-time(Hours *als xs:integer*, **Minutes** *als xs:integer*, **Seconds** *als xs:integer*, **TimeZone** *als xs:string*) als **xs:time** **XP3.1** **XQ3.1**
Das erste, zweite und dritte Argument steht für die Stunde (0 bis 23), bzw. die Minuten (0 bis 59) bzw. die Sekunden (0 bis 59). Das vierte Argument ist ein String, der den Zeitzonenteil des Werts liefert. Die vier Argumente werden zu einem Wert vom Typ xs:time kombiniert. Die Werte der Ganzzahlen müssen sich innerhalb des korrekten Bereichs dieses jeweiligen Uhrzeitteils befinden. So sollte z.B. das zweite Argument (Minuten) nicht größer als 59 sein.
- *Beispiele*
  - **altova:build-time**(23, 4, 57, '+1') gibt 23:04:57+01:00 zurück

build-duration [altova:]

altova:build-duration(Years *als xs:integer*, **Months** *als xs:integer*) als **xs:yearMonthDuration** **XP3.1** **XQ3.1**
Setzt aus zwei Argumenten einen Wert vom Typ xs:yearMonthDuration zusammen. Das erste Argument liefert den Jahr-Teil des Zeitdauerwerts, während das zweite Argument den Monat-Teil liefert. Wenn der zweite Parameter (Monate) größer oder gleich 12 ist, so wird die Ganzzahl durch 12 dividiert. Der Quotient wird zum ersten Argument hinzugefügt, um den Jahr-Teil des Zeitdauerwerts zu liefern, während der Rest (der Division) den Monat-Teil liefert. Eine Beschreibung zur Erstellung einer Zeitdauer vom Typ xs:dayTimeDuration finden Sie in der nächsten Signatur.
- *Beispiele*
  - **altova:build-duration**(2, 10) gibt P2Y10M zurück
  - **altova:build-duration**(14, 27) gibt P16Y3M zurück
  - **altova:build-duration**(2, 24) gibt P4Y zurück

altova:build-duration(Days *als xs:integer*, **Hours** *als xs:integer*, **Minutes** *als xs:integer*, **Seconds** *als xs:integer*) als **xs:dayTimeDuration** **XP3.1** **XQ3.1**
Kombiniert vier Argumente zu einem Wert vom Typ xs:dayTimeDuration. Das erste Argument liefert den Tage-Teil, das zweite die Stunden, das dritte die Minuten und das vierte die Sekunden des Zeitdauerwerts. Die einzelnen Uhrzeitparameter werden in den entsprechenden Wert für die nächsthöhere Einheit konvertiert und das Ergebnis wird zur Berechnung der Gesamtdauer weitergegeben. So werden z.B. 72 Sekunden in 1M(inute)12S(ekunden) konvertiert. Dieser Wert wird zur Berechnung der Gesamtdauer weitergegeben. Um eine Zeitdauer vom Typ xs:yearMonthDuration zu berechnen, verwenden Sie die vorherige Signatur.
- *Beispiele*
  - **altova:build-duration**(2, 10, 3, 56) gibt P2DT10H3M56S zurück
  - **altova:build-duration**(1, 0, 100, 0) gibt P1DT1H40M zurück
  - **altova:build-duration**(1, 0, 0, 3600) gibt P1DT1H zurück

[Nach oben 2148]

Konstruieren von Datum, Datum und Uhrzeit und Zeit-Datentypen anhand des String-Input XP2 XQ1 XP3.1 XQ3.1

Diese Funktionen erhalten Strings als Argumente und konstruieren anhand dieser die Datentypen `xs:date`, `xs:dateTime` oder `xs:time`. Der String wird anhand eines bereitgestellten Pattern-Arguments nach Komponenten des Datentyps analysiert.

parse-date [altova:]

altova:parse-date(Date *als xs:string*, **DatePattern** *als xs:string*) als **xs:date** XP2 XQ1 XP3.1 XQ3.1

Gibt den Input-String **Date** als **xs:date** Wert zurück. Das zweite Argument **DatePattern** definiert das Pattern (die Komponentensequenz) des Input-String. **DatePattern** wird durch die unten aufgelisteten Komponenten-Specifier beschrieben. Als Komponententrennzeichen kann jedes beliebige Zeichen verwendet werden. Siehe Beispiele unten.

D	Datum
M	Monat
Y	Jahr

Das Pattern in **DatePattern** muss mit dem Pattern in **Date** übereinstimmen. Da die Ausgabe vom Typ **xs:date** ist, hat sie immer das lexikalische Format **YYYY-MM-DD**.

Beispiele

- **altova:parse-date**(xs:string("09-12-2014"), "[D]-[M]-[Y]") gibt 2014-12-09 zurück
- **altova:parse-date**(xs:string("09-12-2014"), "[M]-[D]-[Y]") gibt 2014-09-12 zurück
- **altova:parse-date**("06/03/2014", "[M]/[D]/[Y]") gibt 2014-06-03 zurück
- **altova:parse-date**("06 03 2014", "[M] [D] [Y]") gibt 2014-06-03 zurück
- **altova:parse-date**("6 3 2014", "[M] [D] [Y]") gibt 2014-06-03 zurück

parse-dateTime [altova:]

altova:parse-dateTime(DateTime *als xs:string*, **DateTimePattern** *als xs:string*) als **xs:dateTime** XP2 XQ1 XP3.1 XQ3.1

Gibt den Input-String **DateTime** als **xs:dateTime** Wert zurück. Das zweite Argument **DateTimePattern** definiert das Pattern (die Komponentensequenz) des Input-String. **DateTimePattern** wird durch die unten aufgelisteten Komponenten-Specifier beschrieben. Als Komponententrennzeichen kann jedes beliebige Zeichen verwendet werden. Siehe Beispiele unten.

D	Datum
M	Monat
Y	Jahr
H	Stunde
m	Minuten
s	Sekunden

© 2016-2022 Altova GmbH

Das Pattern in **DateTimePattern** muss mit dem Pattern in **DateTime** übereinstimmen. Da die Ausgabe vom Typ **xs:dateTime** ist, hat sie immer das lexikalische Format **YYYY-MM-DDTHH:mm:ss**.

Beispiele

- **altova:parse-dateTime**(xs:string("09-12-2014 13:56:24"), "[M]-[D]-[Y] [H]:[m]:[s]") gibt 2014-09-12T13:56:24 zurück
- **altova:parse-dateTime**("time=13:56:24; date=09-12-2014", "time=[H]:[m]:[s]; date=[D]-[M]-[Y]") gibt 2014-12-09T13:56:24 zurück

parse-time [altova:]

altova:parse-time(Time *als xs:string*, **TimePattern** *als xs:string***)** als **xs:time** XP2 XQ1 XP3.1 XQ3.1

Gibt den Input-String **Time** als **xs:time** Wert zurück. Das zweite Argument **TimePattern** definiert das Pattern (die Komponentensequenz) des Input-String. **TimePattern** wird durch die unten aufgelisteten Komponenten-Specifier beschrieben. Als Komponententrennzeichen kann jedes beliebige Zeichen verwendet werden. Siehe Beispiele unten.

H	Stunde
m	Minuten
s	Sekunden

Das Pattern in **TimePattern** muss mit dem Pattern in **Time** übereinstimmen. Da die Ausgabe vom Typ **xs:Time** ist, hat sie immer das lexikalische Format **HH:mm:ss**.

Beispiele

- **altova:parse-time**(xs:string("13:56:24"), "[H]:[m]:[s]") gibt 13:56:24 zurück
- **altova:parse-time**("13-56-24", "[H]-[m]") gibt 13:56:00 zurück
- **altova:parse-time**("time=13h56m24s", "time=[H]h[m]m[s]s") gibt 13:56:24 zurück
- **altova:parse-time**("time=24s56m13h", "time=[s]s[m]m[H]h") gibt 13:56:24 zurück

[Nach oben 2148]

Funktionen zur Berechnung des Alters XP3.1 XQ3.1

Diese Funktionen geben das Alter berechnet (i) anhand von einem Input-Argument und dem aktuellen Datum oder (ii) anhand zweier Input-Argumentdaten zurück. Die Funktion **altova:age** gibt das Alter in Jahren zurück, die Funktion **altova:age-details** gibt das Alter als Sequenz von drei Ganzzahlen zurück, die die Jahre, Monate und Tage des Alters angeben.

age [altova:]

altova:age(StartDate *als xs:date***)** als **xs:integer** XP3.1 XQ3.1

Gibt eine Ganzzahl zurück, die das Alter eines Objekts in *Jahren* angibt. Berechnet wird das Alter anhand des durch das Argument gelieferten Startdatums endend mit dem aktuellen Datum (laut Systemuhr). Wenn das Input-Argument eines Datums größer oder gleich einem Jahr in der Zukunft ist, ist der Rückgabewert negativ.

Beispiele

Wenn das aktuelle Datum **2014-01-15** lautet:

- **altova:age**(xs:date("2013-01-15")) gibt 1 zurück
- **altova:age**(xs:date("2013-01-16")) gibt 0 zurück
- **altova:age**(xs:date("2015-01-15")) gibt -1 zurück
- **altova:age**(xs:date("2015-01-14")) gibt 0 zurück

altova:age(StartDate *als xs:date*, **EndDate** *als xs:date*) als **xs:integer** **XP3.1** **XQ3.1**
Gibt eine Ganzzahl zurück, die das Alter eines Objekts in *Jahren* angibt. Berechnet wird das Alter anhand des durch das erste Argument gelieferten Startdatums endend mit dem als zweites Datum gelieferten Enddatum. Wenn das erste Argument ein Jahr oder mehr nach dem zweiten Argument liegt, ist der Rückgabewert negativ.

Beispiele

Wenn das aktuelle Datum **2014-01-15** lautet:

- **altova:age**(xs:date("2000-01-15"), xs:date("2010-01-15")) gibt 10 zurück
- **altova:age**(xs:date("2000-01-15"), current-date()) gibt 14 zurück, wenn das aktuelle Datum 2014-01-15 ist
- **altova:age**(xs:date("2014-01-15"), xs:date("2010-01-15")) gibt -4 zurück

age-details [altova:]

altova:age-details(InputDate *als xs:date*) als (**xs:integer**)* **XP3.1** **XQ3.1**
Gibt drei Ganzzahlen zurück. Dabei handelt es sich um die Jahre, Monate bzw. Tage zwischen dem als Argument angegebenen Datum und dem aktuellen Datum (laut Systemuhr). Die Summe der zurückgegebenen years+months+days gibt zusammen die Gesamtzeitdifferenz zwischen den beiden Datumswerten (dem Input-Datum und dem aktuellen Datum) an. Das Input-Datum hat eventuell einen Wert, der vor oder nach dem aktuellen Datum liegt, doch wird dies nicht aus dem Vorzeichen der Rückgabewerte ersichtlich; die Rückgabewerte sind immer positiv.

Beispiele

Wenn das aktuelle Datum **2014-01-15** lautet:

- **altova:age-details**(xs:date("2014-01-16")) gibt (0 0 1) zurück
- **altova:age-details**(xs:date("2014-01-14")) gibt (0 0 1) zurück
- **altova:age-details**(xs:date("2013-01-16")) gibt (1 0 1) zurück
- **altova:age-details**(current-date()) gibt (0 0 0) zurück

altova:age-details(Date-1 *als xs:date*, **Date-2** *als xs:date*) als (**xs:integer**)* **XP3.1** **XQ3.1**
Gibt drei Ganzzahlen zurück. Dabei handelt es sich um die Jahre, Monate bzw. Tage zwischen den beiden Argumentdaten. Die Summe der zurückgegebenen years+months+days gibt zusammen die Gesamtzeitdifferenz zwischen den beiden Input-Datumswerten an. Es ist unerheblich, ob das frühere oder spätere Datum als erstes Argument angegeben wird. Die Rückgabewerte geben nicht an, ob das Input-Datum vor oder nach dem aktuellen Datum liegt. Die Rückgabewerte sind immer positiv.

Beispiele

© 2016-2022 Altova GmbH

- **altova:age-details**(xs:date("2014-01-16"), xs:date("2014-01-15")) gibt (0 0 1) zurück
- **altova:age-details**(xs:date("2014-01-15"), xs:date("2014-01-16")) gibt (0 0 1) zurück

[Nach oben 2148]

Epochen-Zeit (Unix-Zeit)-Funktionen XP3.1 XQ3.1

Die Epochenzeit ist ein auf Unix-Systemen verwendetes Zeitsystem. Darin wird jeder Zeitpunkt als Anzahl der Sekunden seit 00:00:00 UTC des 1. Januar 1970 definiert. Diese Epochenzeitfunktionen konvertieren **xs:dateTime**-Werte in Epochenzeitwerte und umgekehrt.

dateTime-from-epoch [altova:]

altova:dateTime-from-epoch(Epoch *als xs:decimal* **als xs:dateTime** XP3.1 XQ3.1
Die Epochenzeit ist ein auf Unix-Systemen verwendetes Zeitsystem. Darin wird jeder Zeitpunkt als Anzahl der Sekunden seit 00:00:00 UTC des 1. Januar 1970 definiert. Die Funktion **dateTime-from-epoch** gibt das **xs:dateTime**-Äquivalent einer Epochenzeit zurück, passt die lokale Zeitzone an und inkludiert die Zeitzoneninformation im Ergebnis.

Die Funktion erhält ein xs:decimal-Argument und gibt einen xs:dateTime-Wert, der einen **zz**-Teil (Zeitzone) enthält, zurück. Das Ergebnis wird durch Berechnung des UTC **dateTime**-Äquivalents der Epochenzeit und Hinzufügen der (anhand der Systemuhr ermittelten) lokalen Zeitzone ermittelt. Wenn die Funktion z.B. auf einem Rechner, der in der Zeitzone +01:00 (relativ zur UTC) konfiguriert wurde, ausgeführt wird, so wird nach Berechnung des UTC-**dateTime**-Äquivalents eine Stunde zum Ergebnis addiert. Auch die Zeitzoneninformation, die einen optionalen lexikalischen Bestandteil des **xs:dateTime**-Ergebnisses bildet, wird im **dateTime**-Ergebnis ausgegeben. Vergleichen Sie dieses Ergebnis mit dem von **dateTime-from-epoch-no-TZ** und auch der Funktion **epoch-from-dateTime**.

Beispiele

In den Beispielen unten wird eine lokale Zeitzone von UTC +01:00 angenommen. Das UTC **dateTime**-Äquivalent der angegebenen Epochenzeit wird folglich um eine Stunde erhöht. Die Zeitzone wird im Ergebnis ausgegeben.

- **altova:dateTime-from-epoch**(34) gibt 1970-01-01T01:00:34+01:00 zurück.
- **altova:dateTime-from-epoch**(62) gibt 1970-01-01T01:01:02+01:00 zurück.

dateTime-from-epoch-no-TZ [altova:]

altova:dateTime-from-epoch-no-TZ(Epoch *als xs:decimal* **als xs:dateTime** XP3.1 XQ3.1
Die Epochenzeit ist ein auf Unix-Systemen verwendetes Zeitsystem. Darin wird jeder Zeitpunkt als Anzahl der Sekunden seit 00:00:00 UTC des 1. Januar 1970 definiert. Die Funktion **dateTime-from-epoch-no-TZ** gibt das **xs:dateTime**-Äquivalent einer Epochenzeit zurück, passt es an die lokale Zeitzone an, inkludiert die Zeitzoneninformation jedoch nicht im Ergebnis.

Die Funktion erhält ein xs:decimal-Argument und gibt einen xs:dateTime-Wert, der keinen **zz**-Teil (Zeitzone) enthält, zurück. Das Ergebnis wird durch Berechnung des UTC **dateTime**-Äquivalents der Epochenzeit und Hinzufügen der (anhand der Systemuhr ermittelten) lokalen Zeitzone ermittelt. Wenn die Funktion z.B. auf einem Rechner, der in der Zeitzone +01:00 (relativ zur UTC) konfiguriert wurde,

ausgeführt wird, so wird nach Berechnung des UTC-**dateTime**-Äquivalents eine Stunde zum Ergebnis addiert. Die Zeitzoneninformation, die einen optionalen lexikalischen Bestandteil des **xs:dateTime**-Ergebnisses bildet, wird nicht im **dateTime**-Ergebnis ausgegeben. Vergleichen Sie dieses Ergebnis mit dem von **dateTime-from-epoch** und auch der Funktion **epoch-from-dateTime**.

Beispiele

In den Beispielen unten wird eine lokale Zeitzone von UTC +01:00 angenommen. Das UTC **dateTime**-Äquivalent der angegebenen Epochenzeit wird folglich um eine Stunde erhöht. Die Zeitzone wird nicht im Ergebnis ausgegeben.

- **altova:dateTime-from-epoch**(34) gibt 1970-01-01T01:00:34 zurück.
- **altova:dateTime-from-epoch**(62) gibt 1970-01-01T01:01:02 zurück.

epoch-from-dateTime [altova:]

altova:epoch-from-dateTime(dateTimeValue *als xs:dateTime*) **als xs:decimal** XP3.1 XQ3.1

Die Epochenzeit ist ein auf Unix-Systemen verwendetes Zeitsystem. Darin wird jeder Zeitpunkt als Anzahl der Sekunden seit 00:00:00 UTC des 1. Januar 1970 definiert. Die Funktion **epoch-from-dateTime** gibt das Epochenzeitäquivalent von **xs:dateTime** zurück, welches als Argument der Funktion bereitgestellt wird. Beachten Sie, dass Sie den **xs:dateTime**-Wert eventuell explizit konstruieren müssen. Der angegebene **xs:dateTime**-Wert kann den optionalen **zz** (Zeitzonen)-Wert enthalten, muss ihn aber nicht enthalten.

Unabhängig davon, ob der Zeitzonenteil als Bestandteil des Arguments angegeben wird oder nicht, wird der (anhand der Systemuhr ermittelte) lokale Zeitzonenunterschied vom angegebenen **dateTimeValue**-Argument subtrahiert. Dadurch wird das UTC-Zeit-Äquivalent erzeugt, anhand dessen die entsprechende Epochenzeit berechnet wird. Wenn die Funktion z.B. auf einem Rechner, der für die Zeitzone +01:00 (relativ zur UTC) konfiguriert wurde, ausgeführt wird, so wird vor Berechnung des Epochenzeitwerts eine Stunde vom angegebenen **dateTimeValue** subtrahiert. Siehe dazu auch die Funktion **dateTime-from-epoch**.

Beispiele

In den Beispielen unten wird eine lokale Zeitzone von UTC +01:00 angenommen. Daher wird vor Berechnung der Epochenzeit eine Stunde vom angegebenen **dateTime**-Wert subtrahiert.

- **altova:epoch-from-dateTime**(xs:dateTime("1970-01-01T01:00:34+01:00")) gibt 34 zurück.
- **altova:epoch-from-dateTime**(xs:dateTime("1970-01-01T01:00:34")) gibt 34 zurück.
- **altova:epoch-from-dateTime**(xs:dateTime("2021-04-01T11:22:33")) gibt 1617272553 zurück.

[Nach oben 2148]

© 2016-2022 Altova GmbH

30.2.1.3 XPath/XQuery-Funktionen: Standort

Die folgenden XPath/XQuery-Erweiterungsfunktionen zu Standortdaten werden in der aktuellen Version von XMLSpy unterstützt und können in (i) in einem XSLT-Kontext in XPath-Ausdrücken oder (ii) in einem XQuery-Dokument in einem XQuery-Ausdruck verwendet werden.

Anmerkung zur Benennung von Funktionen und zur Anwendbarkeit der Sprache

Altova-Erweiterungsfunktionen können in XPath/XQuery-Ausdrücken verwendet werden. Dadurch stehen neben den Funktionen in der Standardbibliothek der XPath-, XQuery- und XSLT-Funktionen zusätzliche Funktionen zur Verfügung. Die Altova-Erweiterungsfunktionen befinden sich im **Altova-Erweiterungsfunktions-Namespace, `http://www.altova.com/xslt-extensions`** und sind in diesem Abschnitt mit dem Präfix **`altova:`**, das als an diesen Namespace gebunden angenommen wird, gekennzeichnet. Beachten Sie, dass manche Funktionen in zukünftigen Versionen Ihres Produkts eventuell nicht mehr unterstützt werden oder dass sich das Verhalten einzelner Funktionen ändern kann. Um zu sehen, welche Altova-Erweiterungsfunktionen unterstützt werden, lesen Sie bitte die Dokumentation zur jeweiligen Release.

XPath-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XP1 XP2 XP3.1
XSLT-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XSLT1 XSLT2 XSLT3
XQuery-Funktionen (in XQuery-Ausdrücken in XQuery verwendet):	XQ1 XQ3.1

▼ format-geolocation [altova:]

`altova:format-geolocation(Latitude` *als xs:decimal*, **`Longitude`** *als xs:decimal*, **`GeolocationOutputStringFormat`** *als xs:integer*`)` als **`xs:string`** XP3.1 XQ3.1
Erhält als die ersten beiden Argumente die geografische Breite und Länge und gibt den Standort als String zurück. Das dritte Argument, **`GeolocationOutputStringFormat`**, ist das Format des Ausgabestring für den Standort; darin werden zum Identifizieren des Ausgabestringformats Ganzzahlwerte von `1` bis `4` verwendet (*siehe 'Format des Ausgabestrings für die geografische Position' weiter unten*). Die Werte für die Breite liegen im Bereich von `+90` bis `-90` (`N` nach `S`). Die Werte für die Länge liegen im Bereich von `+180` bis `-180` (`O` nach `W`).

Anmerkung: Zur Bereitstellung der Input-Strings können die Funktion image-exif-data[2179] und die Attribute der Exif-Metadaten verwendet werden.

- *Beispiele*
  - **`altova:format-geolocation`**`(33.33, -22.22, 4)` gibt `xs:string "33.33 -22.22"` zurück
  - **`altova:format-geolocation`**`(33.33, -22.22, 2)` gibt `xs:string "33.33N 22.22W"` zurück
  - **`altova:format-geolocation`**`(-33.33, 22.22, 2)` gibt `xs:string "33.33S 22.22E"` zurück
  - **`altova:format-geolocation`**`(33.33, -22.22, 1)` gibt `xs:string "33°19'48.00"S 22° 13'12.00"E"` zurück

- *Ausgabestringformate für die geografische Position:*

  Die bereitgestellte Breite und Länge ist in einem der unten aufgelisteten Ausgabeformate formatiert. Das gewünschte Format wird anhand seiner Ganzzahl-ID (`1` bis `4`) identifiziert. Die Breitenwerte liegen im Bereich von `+90` bis `-90` (`N` nach `S`). Die Längenwerte liegen im Bereich von `+180` bis `-180` (`O` nach `W`).

1
Grad, Minuten, Dezimalsekunden, mit nachgestellter Orientierung (N/S, E/W) **D°M'S.SS"N/S D°M'S.SS"E/W** *Beispiel:* **33°55'11.11"N 22°44'66.66"W**

2
Dezimalgrad, mit nachgestellter Orientierung (N/S, E/W) **D.DDN/S D.DDE/W** *Beispiel:* **33.33N 22.22W**

3
Grad, Minuten, Dezimalsekunden mit Vorzeichen (+/-); Plus-Zeichen für (**N/E**) ist optional **+/-D°M'S.SS" +/-D°M'S.SS"** *Beispiel:* **33°55'11.11" -22°44'66.66"**

4
Dezimalgrad, mit Vorzeichen (+/-); Plus-Zeichen für (**N/E**) ist optional **+/-D.DD +/-D.DD** *Beispiel:* **33.33 -22.22**

- *Altova Exif-Attribut: Geolocation*

  Der Altova XPath/XQuery-Prozessor generiert anhand der Exif-Standard-Metadaten-Tags das benutzerdefinierte Attribut **Geolocation**. **Geolocation** ist eine Verkettung von vier Exif-Tags: GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef mit hinzugefügten Werten (*siehe Tabelle unten*).

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

parse-geolocation [altova:]

altova:parse-geolocation(GeolocationInputString *als xs:string*) als **xs:decimal+** **XP3.1** **XQ3.1**

Parst das bereitgestellte GeolocationInputString-Argument und gibt die geografische Breite und Länge (in dieser Reihenfolge) als Sequenz aus zwei xs:decimal Elementen zurück. Die Formate, in denen der Input-String für die geografische Position bereitgestellt werden kann, sind unten aufgelistet.

Anmerkung: Zur Bereitstellung des Input-String für die geografische Position können die Funktion image-exif-data[2179] und das @Geolocation[2179]-Attribut der Exif-Metadaten verwendet werden (*siehe Beispiel*

© 2016-2022 Altova GmbH

unten).

- *Beispiele*
  - `altova:parse-geolocation("33.33 -22.22")` gibt die Sequenz bestehend aus zwei `xs:decimals (33.33, 22.22)` Elementen zurück
  - `altova:parse-geolocation("48°51'29.6""N 24°17'40.2""")` gibt die Sequenz bestehend aus zwei `xs:decimals (48.8582222222222, 24.2945)` Elementen zurück
  - `altova:parse-geolocation('48°51''29.6"N 24°17''40.2"')` gibt die Sequenz bestehend aus zwei `xs:decimals (48.8582222222222, 24.2945)` Elementen zurück
  - `altova:parse-geolocation( image-exif-data(//MyImages/Image20141130.01)/@Geolocation )` gibt die Sequenz bestehend aus zwei `xs:decimals` Elementen zurück

- *Input-String-Formate der Standortdaten:*

  Der Input-String für die geografische Position muss die Breite und Länge (in dieser Reihenfolge) getrennt durch ein Leerzeichen enthalten. Beide Werte können jedes der folgenden Formate haben. Auch Kombinationen sind zulässig, d.h. die Breite kann in einem anderen Format als die Länge angegeben werden. Die Breitenwerte liegen im Bereich `+90` bis `-90` (`N` nach `S`). Die Längenwerte liegen im Bereich von `+180` bis `-180` (`O` nach `W`).

  Anmerkung: Wenn als Trennzeichen für den Input-String einfache oder doppelte Anführungszeichen verwendet werden, kann dies zu einer Fehlinterpretation der einfachen bzw. doppelten Anführungszeichen als Minuten- bzw. Sekundenwerte führen. In solchen Fällen müssen die zur Angabe der Minuten- und Sekundenwerte verwendeten Anführungszeichen durch Verdoppelung mit einem Escape-Zeichen versehen werden. In den Beispielen in diesem Abschnitt sind Anführungszeichen, die als Trennzeichen für den Input-String dienen, gelb markiert (`"`), während Maßeinheitsangaben blau (`""`) markiert sind.

  - Grad, Minuten, Dezimalsekunden, mit nachgestellter Orientierung (`N/S, O/W`)
    `D°M'S.SS"N/S D°M'S.SS"W/E`
    Beispiel: **`33°55'11.11"N 22°44'55.25"W`**

  - Grad, Minuten, Dezimalsekunden mit Vorzeichen (`+/-`); das Plus-Zeichen für (**`N`**/`O`) ist optional
    `+/-D°M'S.SS" +/-D°M'S.SS"`
    Beispiel: **`33°55'11.11" -22°44'55.25"`**

  - Grad, Dezimalminuten mit nachgestellter Orientierung (`N/S`, `O/W`)
    `D°M.MM'N/S D°M.MM'W/E`
    Beispiel: **`33°55.55'N 22°44.44'W`**

  - Grad, Dezimalminuten mit Vorzeichen (`+/-`); das Plus-Zeichen für (**`N`**/`O`) ist optional
    `+/-D°M.MM' +/-D°M.MM'`
    Beispiel: **`+33°55.55' -22°44.44'`**

  - Dezimalgrade, mit nachgestellter Orientierung (`N/S`, `O/W`)
    `D.DDN/S D.DDW/E`
    Beispiel: **`33.33N 22.22W`**

  - Dezimalgrade mit Vorzeichen (`+/-`); das Plus-Zeichen für (**`N/S O/W`**) ist optional

`+/-D.DD +/-D.DD`
Beispiel: `33.33 -22.22`

Beispiele für Formatkombinationen:
`33.33N -22°44'55.25"`
`33.33 22°44'55.25"W`
`33.33 22.45`

- *Altova Exif-Attribut: Geolocation*

  Der Altova XPath/XQuery-Prozessor generiert anhand der Exif-Standard-Metadaten-Tags das benutzerdefinierte Attribut **Geolocation**. **Geolocation** ist eine Verkettung von vier Exif-Tags: GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef mit hinzugefügten Werten (*siehe Tabelle unten*).

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

- geolocation-distance-km [altova:]

altova:geolocation-distance-km(GeolocationInputString-1 *als xs:string*, **GeolocationInputString-2** *als xs:string*) als **xs:decimal** **XP3.1** **XQ3.1**
Berechnet die Entfernung zwischen zwei geografischen Positionen in Kilometern. Die Formate, in denen der Input-String für die geografischen Position angegeben werden kann, sind unten aufgelistet. Die Breitenwerte liegen im Bereich von +90 bis -90 (N nach S). Die Längenwerte liegen im Bereich von +180 bis -180 (O nach W).

Anmerkung: Zur Bereitstellung des Input-String für die geografische Position können die Funktion image-exif-data[2179] und das @Geolocation[2179]-Attribut der Exif-Metadaten verwendet werden.

- *Beispiele*
  - **altova:geolocation-distance-km**("33.33 -22.22", "48°51'29.6""N 24°17'40.2""") gibt xs:decimal 4183.08132372392 zurück

- *Input-String-Formate der Standortdaten:*

  Der Input-String für die geografische Position muss die Breite und Länge (in dieser Reihenfolge) getrennt durch ein Leerzeichen enthalten. Beide Werte können jedes der folgenden Formate haben. Auch Kombinationen sind zulässig, d.h. die Breite kann in einem anderen Format als die Länge angegeben werden. Die Breitenwerte liegen im Bereich +90 bis -90 (N nach S). Die Längenwerte liegen im Bereich von +180 bis -180 (O nach W).

  Anmerkung: Wenn als Trennzeichen für den Input-String einfache oder doppelte Anführungszeichen verwendet werden, kann dies zu einer Fehlinterpretation der einfachen bzw. doppelten Anführungszeichen als Minuten- bzw. Sekundenwerte führen. In solchen Fällen müssen die zur Angabe der Minuten- und Sekundenwerte verwendeten Anführungszeichen durch Verdoppelung mit

© 2016-2022 Altova GmbH

einem Escape-Zeichen versehen werden. In den Beispielen in diesem Abschnitt sind Anführungszeichen, die als Trennzeichen für den Input-String dienen, gelb markiert (**"**), während Maßeinheitsangaben blau (**""**) markiert sind.

- Grad, Minuten, Dezimalsekunden, mit nachgestellter Orientierung (N/S, O/W)
  D°M'S.SS"N/S D°M'S.SS"W/E
  Beispiel: **33°55'11.11"N 22°44'55.25"W**

- Grad, Minuten, Dezimalsekunden mit Vorzeichen (+/-); das Plus-Zeichen für (**N**/O) ist optional
  +/-D°M'S.SS" +/-D°M'S.SS"
  Beispiel: **33°55'11.11" -22°44'55.25"**

- Grad, Dezimalminuten mit nachgestellter Orientierung (N/S, O/W)
  D°M.MM'N/S D°M.MM'W/E
  Beispiel: **33°55.55'N 22°44.44'W**

- Grad, Dezimalminuten mit Vorzeichen (+/-); das Plus-Zeichen für (**N**/O) ist optional
  +/-D°M.MM' +/-D°M.MM'
  Beispiel: **+33°55.55' -22°44.44'**

- Dezimalgrade, mit nachgestellter Orientierung (N/S, O/W)
  D.DDN/S D.DDW/E
  Beispiel: **33.33N 22.22W**

- Dezimalgrade mit Vorzeichen (+/-); das Plus-Zeichen für (**N/S O/W**) ist optional
  +/-D.DD +/-D.DD
  Beispiel: **33.33 -22.22**

Beispiele für Formatkombinationen:
33.33N -22°44'55.25"
33.33 22°44'55.25"W
33.33 22.45

Altova Exif-Attribut: Geolocation

Der Altova XPath/XQuery-Prozessor generiert anhand der Exif-Standard-Metadaten-Tags das benutzerdefinierte Attribut **Geolocation**. **Geolocation** ist eine Verkettung von vier Exif-Tags: GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef mit hinzugefügten Werten (*siehe Tabelle unten*).

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-distance-mi [altova:]

altova:geolocation-distance-mi(GeolocationInputString-1 *als xs:string*, **GeolocationInputString-2** *als xs:string*) als **xs:decimal** **XP3.1** **XQ3.1**
Berechnet die Entfernung zwischen zwei geografischen Positionen in Meilen. Die Formate, in denen der Input-String für die geografischen Position angegeben werden kann, sind unten aufgelistet. Die Breitenwerte liegen im Bereich von `+90` bis `-90` (`N` nach `S`). Die Längenwerte liegen im Bereich von `+180` bis `-180` (`O` nach `W`).

Anmerkung: Zur Bereitstellung des Input-String für die geografische Position können die Funktion image-exif-data[2179] und das @Geolocation[2179]-Attribut der Exif-Metadaten verwendet werden.

- *Beispiele*
  - **altova:geolocation-distance-mi**(`"33.33 -22.22", "48°51'29.6""N 24°17'40.2"""`) gibt `xs:decimal 2599.40652340653` zurück

- *Input-String-Formate der Standortdaten:*

  Der Input-String für die geografische Position muss die Breite und Länge (in dieser Reihenfolge) getrennt durch ein Leerzeichen enthalten. Beide Werte können jedes der folgenden Formate haben. Auch Kombinationen sind zulässig, d.h. die Breite kann in einem anderen Format als die Länge angegeben werden. Die Breitenwerte liegen im Bereich `+90` bis `-90` (`N` nach `S`). Die Längenwerte liegen im Bereich von `+180` bis `-180` (`O` nach `W`).

  Anmerkung: Wenn als Trennzeichen für den Input-String einfache oder doppelte Anführungszeichen verwendet werden, kann dies zu einer Fehlinterpretation der einfachen bzw. doppelten Anführungszeichen als Minuten- bzw. Sekundenwerte führen. In solchen Fällen müssen die zur Angabe der Minuten- und Sekundenwerte verwendeten Anführungszeichen durch Verdoppelung mit einem Escape-Zeichen versehen werden. In den Beispielen in diesem Abschnitt sind Anführungszeichen, die als Trennzeichen für den Input-String dienen, gelb markiert (`"`), während Maßeinheitsangaben blau (`""`) markiert sind.

  - Grad, Minuten, Dezimalsekunden, mit nachgestellter Orientierung (`N/S, O/W`)
    `D°M'S.SS"N/S D°M'S.SS"W/E`
    Beispiel: **`33°55'11.11"N 22°44'55.25"W`**

  - Grad, Minuten, Dezimalsekunden mit Vorzeichen (`+/-`); das Plus-Zeichen für (**`N`**/**`O`**) ist optional
    `+/-D°M'S.SS" +/-D°M'S.SS"`
    Beispiel: **`33°55'11.11" -22°44'55.25"`**

  - Grad, Dezimalminuten mit nachgestellter Orientierung (`N/S`, `O/W`)
    `D°M.MM'N/S D°M.MM'W/E`
    Beispiel: **`33°55.55'N 22°44.44'W`**

  - Grad, Dezimalminuten mit Vorzeichen (`+/-`); das Plus-Zeichen für (**`N`**/**`O`**) ist optional
    `+/-D°M.MM' +/-D°M.MM'`
    Beispiel: **`+33°55.55' -22°44.44'`**

  - Dezimalgrade, mit nachgestellter Orientierung (`N/S`, `O/W`)
    `D.DDN/S D.DDW/E`
    Beispiel: **`33.33N 22.22W`**

© 2016-2022 Altova GmbH

- Dezimalgrade mit Vorzeichen (+/-); das Plus-Zeichen für (**N/S O/W**) ist optional
  +/-D.DD +/-D.DD
  Beispiel: **33.33 -22.22**

Beispiele für Formatkombinationen:
33.33N -22°44'55.25"
33.33 22°44'55.25"W
33.33 22.45

- *Altova Exif-Attribut: Geolocation*

  Der Altova XPath/XQuery-Prozessor generiert anhand der Exif-Standard-Metadaten-Tags das benutzerdefinierte Attribut **Geolocation**. **Geolocation** ist eine Verkettung von vier Exif-Tags: GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef mit hinzugefügten Werten (*siehe Tabelle unten*).

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocations-bounding-rectangle [altova:]

altova:geolocations-bounding-rectangle(Geolocations *als xs:sequence*, **GeolocationOutputStringFormat** *als xs:integer*) als **xs:string** **XP3.1** **XQ3.1**
Erhält als erstes Argument eine Sequenz von Strings, wobei es sich bei jedem String in der Sequenz um eine geografische Position handelt. Die Funktion gibt eine Sequenz von zwei Strings zurück, die die geografischen Positionskoordinaten der linken oberen bzw. rechten unteren Ecke eines Rechtecks bilden, dessen Größe so angepasst ist, dass es alle im ersten Argument angegebenen Positionskoordinaten enthält. Die Formate, in denen der Input-String für die geografischen Position angegeben werden kann, sind unten aufgelistet (*siehe 'Input-String-Formate der Standortdaten'*). Die Breitenwerte liegen im Bereich von +90 bis -90 (N nach S). Die Längenwerte liegen im Bereich von +180 bis -180 (O nach W).

Im zweiten Argument der Funktion ist das Format der beiden Geolocation-Strings in der Ausgabesequenz angegeben. Das Argument erhält einen Ganzzahlwert von 1 bis 4, wobei die einzelnen Werte ein jeweils unterschiedliches String-Format definieren (*siehe 'Ausgabestringsformate für die geografische Position' weiter unten).*

Anmerkung: Zur Bereitstellung der Input-Strings können die Funktion image-exif-data 2179 und die Attribute der Exif-Metadaten verwendet werden.

- *Beispiele*
  - **altova:geolocations-bounding-rectangle**(("48.2143531 16.3707266", "51.50939 -0.11832"), 1) gibt die Sequenz ("51°30'33.804"N 0°7'5.952"W", "48°12'51.67116"N 16°22'14.61576"E") zurück.
  - **altova:geolocations-bounding-rectangle**(("48.2143531 16.3707266", "51.50939 -0.11832", "42.5584577 -70.8893334"), 4) gibt die Sequenz ("51.50939 -70.8893334", "42.5584577 16.3707266") zurück.

Input-String-Formate der Standortdaten:

Der Input-String für die geografische Position muss die Breite und Länge (in dieser Reihenfolge) getrennt durch ein Leerzeichen enthalten. Beide Werte können jedes der folgenden Formate haben. Auch Kombinationen sind zulässig, d.h. die Breite kann in einem anderen Format als die Länge angegeben werden. Die Breitenwerte liegen im Bereich `+90` bis `-90` (`N` nach `S`). Die Längenwerte liegen im Bereich von `+180` bis `-180` (`O` nach `W`).

Anmerkung: Wenn als Trennzeichen für den Input-String einfache oder doppelte Anführungszeichen verwendet werden, kann dies zu einer Fehlinterpretation der einfachen bzw. doppelten Anführungszeichen als Minuten- bzw. Sekundenwerte führen. In solchen Fällen müssen die zur Angabe der Minuten- und Sekundenwerte verwendeten Anführungszeichen durch Verdoppelung mit einem Escape-Zeichen versehen werden. In den Beispielen in diesem Abschnitt sind Anführungszeichen, die als Trennzeichen für den Input-String dienen, gelb markiert (`"`), während Maßeinheitsangaben blau (`""`) markiert sind.

- Grad, Minuten, Dezimalsekunden, mit nachgestellter Orientierung (`N/S, O/W`)
  `D°M'S.SS"N/S D°M'S.SS"W/E`
  Beispiel: **`33°55'11.11"N 22°44'55.25"W`**

- Grad, Minuten, Dezimalsekunden mit Vorzeichen (`+/-`); das Plus-Zeichen für (**`N/o`**) ist optional
  `+/-D°M'S.SS" +/-D°M'S.SS"`
  Beispiel: **`33°55'11.11" -22°44'55.25"`**

- Grad, Dezimalminuten mit nachgestellter Orientierung (`N/S, O/W`)
  `D°M.MM'N/S D°M.MM'W/E`
  Beispiel: **`33°55.55'N 22°44.44'W`**

- Grad, Dezimalminuten mit Vorzeichen (`+/-`); das Plus-Zeichen für (**`N/o`**) ist optional
  `+/-D°M.MM' +/-D°M.MM'`
  Beispiel: **`+33°55.55' -22°44.44'`**

- Dezimalgrade, mit nachgestellter Orientierung (`N/S, O/W`)
  `D.DDN/S D.DDW/E`
  Beispiel: **`33.33N 22.22W`**

- Dezimalgrade mit Vorzeichen (`+/-`); das Plus-Zeichen für (**`N/S O/W`**) ist optional
  `+/-D.DD +/-D.DD`
  Beispiel: **`33.33 -22.22`**

Beispiele für Formatkombinationen:

```
33.33N  -22°44'55.25"
33.33  22°44'55.25"W
33.33  22.45
```

Ausgabestringformate für die geografische Position:

Die bereitgestellte Breite und Länge ist in einem der unten aufgelisteten Ausgabeformate formatiert. Das gewünschte Format wird anhand seiner Ganzzahl-ID (`1` bis `4`) identifiziert. Die Breitenwerte

© 2016-2022 Altova GmbH

liegen im Bereich von `+90` bis `-90` (`N` nach `S`). Die Längenwerte liegen im Bereich von `+180` bis `-180` (`O` nach `W`).

1
Grad, Minuten, Dezimalsekunden, mit nachgestellter Orientierung (`N/S, E/W`) **`D°M'S.SS"N/S D°M'S.SS"E/W`** *Beispiel:* **`33°55'11.11"N 22°44'66.66"W`**

2
Dezimalgrad, mit nachgestellter Orientierung (`N/S`, `E/W`) **`D.DDN/S D.DDE/W`** *Beispiel:* **`33.33N 22.22W`**

3
Grad, Minuten, Dezimalsekunden mit Vorzeichen (`+/-`); Plus-Zeichen für (**`N/E`**) ist optional **`+/-D°M'S.SS" +/-D°M'S.SS"`** *Beispiel:* **`33°55'11.11" -22°44'66.66"`**

4
Dezimalgrad, mit Vorzeichen (`+/-`); Plus-Zeichen für (**`N/E`**) ist optional **`+/-D.DD +/-D.DD`** *Beispiel:* **`33.33 -22.22`**

Altova Exif-Attribut: Geolocation

Der Altova XPath/XQuery-Prozessor generiert anhand der Exif-Standard-Metadaten-Tags das benutzerdefinierte Attribut **`Geolocation`**. **`Geolocation`** ist eine Verkettung von vier Exif-Tags: `GPSLatitude`, `GPSLatitudeRef`, `GPSLongitude`, `GPSLongitudeRef` mit hinzugefügten Werten (*siehe Tabelle unten*).

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-within-polygon [altova:]

`altova:geolocation-within-polygon(Geolocation` *als xs:string*, **`((PolygonPoint`** *als xs:string*`)+))` als **`xs:boolean`** **XP3.1** **XQ3.1**

Ermittelt ob sich **`Geolocation`** (das erste Argument) innerhalb des durch die **`PolygonPoint`**-Argumente beschriebenen Polygonbereichs befindet. Wenn die `PolygonPoint`-Argumente keine geschlossene Form (wenn der erste und der letzte Punkt identisch sind) bilden, so wird der erste Punkt implizit zum letzten

Punkt hinzugefügt, um die Form zu schließen. Alle Argumente (`Geolocation` und `PolygonPoint+`) werden durch Input-Strings für die geografische Position (*Formatliste siehe unten*) angegeben. Wenn sich das `Geolocation` Argument innerhalb des Polygons befindet, gibt die Funktion `true()` zurück; andernfalls gibt sie `false()` zurück. Die Breitenwerte liegen im Bereich von `+90` bis `-90` (`N` nach `S`). Die Längenwerte liegen im Bereich von `+180` bis `-180` (`O` nach `W`).

Anmerkung: Zur Bereitstellung des Input-String für die geografische Position können die Funktion `image-exif-data`[2179] und das `@Geolocation`[2179]-Attribut der Exif-Metadaten verwendet werden.

Beispiele

- **`altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48 24", "58 -32"))`** gibt `true()` zurück
- **`altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48 24"))`** gibt `true()` zurück
- **`altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48°51'29.6""N 24°17'40.2""))`** gibt `true()` zurück

Input-String-Formate der Standortdaten:

Der Input-String für die geografische Position muss die Breite und Länge (in dieser Reihenfolge) getrennt durch ein Leerzeichen enthalten. Beide Werte können jedes der folgenden Formate haben. Auch Kombinationen sind zulässig, d.h. die Breite kann in einem anderen Format als die Länge angegeben werden. Die Breitenwerte liegen im Bereich `+90` bis `-90` (`N` nach `S`). Die Längenwerte liegen im Bereich von `+180` bis `-180` (`O` nach `W`).

Anmerkung: Wenn als Trennzeichen für den Input-String einfache oder doppelte Anführungszeichen verwendet werden, kann dies zu einer Fehlinterpretation der einfachen bzw. doppelten Anführungszeichen als Minuten- bzw. Sekundenwerte führen. In solchen Fällen müssen die zur Angabe der Minuten- und Sekundenwerte verwendeten Anführungszeichen durch Verdoppelung mit einem Escape-Zeichen versehen werden. In den Beispielen in diesem Abschnitt sind Anführungszeichen, die als Trennzeichen für den Input-String dienen, gelb markiert (`"`), während Maßeinheitsangaben blau (`""`) markiert sind.

- Grad, Minuten, Dezimalsekunden, mit nachgestellter Orientierung (`N/S, O/W`)
  `D°M'S.SS"N/S D°M'S.SS"W/E`
  Beispiel: **`33°55'11.11"N 22°44'55.25"W`**

- Grad, Minuten, Dezimalsekunden mit Vorzeichen (`+/-`); das Plus-Zeichen für (**`N`/`O`**) ist optional
  `+/-D°M'S.SS" +/-D°M'S.SS"`
  Beispiel: **`33°55'11.11" -22°44'55.25"`**

- Grad, Dezimalminuten mit nachgestellter Orientierung (`N/S`, `O/W`)
  `D°M.MM'N/S D°M.MM'W/E`
  Beispiel: **`33°55.55'N 22°44.44'W`**

- Grad, Dezimalminuten mit Vorzeichen (`+/-`); das Plus-Zeichen für (**`N`/`O`**) ist optional
  `+/-D°M.MM' +/-D°M.MM'`
  Beispiel: **`+33°55.55' -22°44.44'`**

- Dezimalgrade, mit nachgestellter Orientierung (`N/S`, `O/W`)

© 2016-2022 Altova GmbH

D.DDN/S D.DDW/E
Beispiel: **33.33N 22.22W**

- Dezimalgrade mit Vorzeichen (+/-); das Plus-Zeichen für (**N/S O/W**) ist optional
  +/-D.DD +/-D.DD
  Beispiel: **33.33 -22.22**

Beispiele für Formatkombinationen:
33.33N -22°44'55.25"
33.33 22°44'55.25"W
33.33 22.45

Altova Exif-Attribut: Geolocation

Der Altova XPath/XQuery-Prozessor generiert anhand der Exif-Standard-Metadaten-Tags das benutzerdefinierte Attribut **Geolocation**. **Geolocation** ist eine Verkettung von vier Exif-Tags: GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef mit hinzugefügten Werten (*siehe Tabelle unten*).

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

geolocation-within-rectangle [altova:]

altova:geolocation-within-rectangle(Geolocation *als xs:string*, **RectCorner-1** *als xs:string*, **RectCorner-2** *als xs:string*) als **xs:boolean** **XP3.1** **XQ3.1**
Ermittelt, ob sich **Geolocation** (das erste Argument) innerhalb des durch das zweite und dritte Argument, **RectCorner-1** und **RectCorner-2**, definierten Rechtecks befindet. **RectCorner-1** und **RectCorner-2** definieren gegenüberliegende Eckpunkte des Rechtecks. Alle Argumente (Geolocation, **RectCorner-1** und **RectCorner-2**) werden durch Input-Strings für die geografische Position (*Formatliste siehe unten*) angegeben. Wenn sich das Geolocation-Argument innerhalb des Rechtecks befindet, gibt die Funktion true() zurück; andernfalls gibt sie false() zurück. Die Breitenwerte liegen im Bereich von +90 bis -90 (N nach S). Die Längenwerte liegen im Bereich von +180 bis -180 (O nach W).

Anmerkung: Zur Bereitstellung des Input-String für die geografische Position können die Funktion image-exif-data[2179] und das @Geolocation[2179]-Attribut der Exif-Metadaten verwendet werden.

Beispiele

- **altova:geolocation-within-rectangle**("33 -22", "58 -32", "-48 24") gibt true() zurück
- **altova:geolocation-within-rectangle**("33 -22", "58 -32", "48 24") gibt false() zurück
- **altova:geolocation-within-rectangle**("33 -22", "58 -32", "48°51'29.6""S 24° 17'40.2""") gibt true() zurück

Input-String-Formate der Standortdaten:

Der Input-String für die geografische Position muss die Breite und Länge (in dieser Reihenfolge) getrennt durch ein Leerzeichen enthalten. Beide Werte können jedes der folgenden Formate haben. Auch Kombinationen sind zulässig, d.h. die Breite kann in einem anderen Format als die Länge angegeben werden. Die Breitenwerte liegen im Bereich `+90` bis `-90` (`N` nach `S`). Die Längenwerte liegen im Bereich von `+180` bis `-180` (`O` nach `W`).

Anmerkung: Wenn als Trennzeichen für den Input-String einfache oder doppelte Anführungszeichen verwendet werden, kann dies zu einer Fehlinterpretation der einfachen bzw. doppelten Anführungszeichen als Minuten- bzw. Sekundenwerte führen. In solchen Fällen müssen die zur Angabe der Minuten- und Sekundenwerte verwendeten Anführungszeichen durch Verdoppelung mit einem Escape-Zeichen versehen werden. In den Beispielen in diesem Abschnitt sind Anführungszeichen, die als Trennzeichen für den Input-String dienen, gelb markiert (`"`), während Maßeinheitsangaben blau (`""`) markiert sind.

- Grad, Minuten, Dezimalsekunden, mit nachgestellter Orientierung (`N/S, O/W`)
  `D°M'S.SS"N/S D°M'S.SS"W/E`
  Beispiel: **`33°55'11.11"N 22°44'55.25"W`**

- Grad, Minuten, Dezimalsekunden mit Vorzeichen (`+/-`); das Plus-Zeichen für (**`N/O`**) ist optional
  `+/-D°M'S.SS" +/-D°M'S.SS"`
  Beispiel: **`33°55'11.11" -22°44'55.25"`**

- Grad, Dezimalminuten mit nachgestellter Orientierung (`N/S`, `O/W`)
  `D°M.MM'N/S D°M.MM'W/E`
  Beispiel: **`33°55.55'N 22°44.44'W`**

- Grad, Dezimalminuten mit Vorzeichen (`+/-`); das Plus-Zeichen für (**`N/O`**) ist optional
  `+/-D°M.MM' +/-D°M.MM'`
  Beispiel: **`+33°55.55' -22°44.44'`**

- Dezimalgrade, mit nachgestellter Orientierung (`N/S`, `O/W`)
  `D.DDN/S D.DDW/E`
  Beispiel: **`33.33N 22.22W`**

- Dezimalgrade mit Vorzeichen (`+/-`); das Plus-Zeichen für (**`N/S O/W`**) ist optional
  `+/-D.DD +/-D.DD`
  Beispiel: **`33.33 -22.22`**

Beispiele für Formatkombinationen:

```
33.33N  -22°44'55.25"
33.33  22°44'55.25"W
33.33  22.45
```

Altova Exif-Attribut: Geolocation

Der Altova XPath/XQuery-Prozessor generiert anhand der Exif-Standard-Metadaten-Tags das benutzerdefinierte Attribut **`Geolocation`**. **`Geolocation`** ist eine Verkettung von vier Exif-Tags: `GPSLatitude`, `GPSLatitudeRef`, `GPSLongitude`, `GPSLongitudeRef` mit hinzugefügten Werten (*siehe Tabelle unten*).

© 2016-2022 Altova GmbH

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

[Nach oben 2167]

30.2.1.4 XPath/XQuery-Funktionen: Bildbezogene

Die folgenden XPath/XQuery-Erweiterungsfunktionen im Zusammenhang mit Bildern werden in der aktuellen Version von XMLSpy unterstützt und können in (i) in einem XSLT-Kontext in XPath-Ausdrücken oder (ii) in einem XQuery-Dokument in einem XQuery-Ausdruck verwendet werden.

Anmerkung zur Benennung von Funktionen und zur Anwendbarkeit der Sprache

Altova-Erweiterungsfunktionen können in XPath/XQuery-Ausdrücken verwendet werden. Dadurch stehen neben den Funktionen in der Standardbibliothek der XPath-, XQuery- und XSLT-Funktionen zusätzliche Funktionen zur Verfügung. Die Altova-Erweiterungsfunktionen befinden sich im **Altova-Erweiterungsfunktions-Namespace, `http://www.altova.com/xslt-extensions`** und sind in diesem Abschnitt mit dem Präfix **`altova:`**, das als an diesen Namespace gebunden angenommen wird, gekennzeichnet. Beachten Sie, dass manche Funktionen in zukünftigen Versionen Ihres Produkts eventuell nicht mehr unterstützt werden oder dass sich das Verhalten einzelner Funktionen ändern kann. Um zu sehen, welche Altova-Erweiterungsfunktionen unterstützt werden, lesen Sie bitte die Dokumentation zur jeweiligen Release.

XPath-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XP1 XP2 XP3.1
XSLT-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XSLT1 XSLT2 XSLT3
XQuery-Funktionen (in XQuery-Ausdrücken in XQuery verwendet):	XQ1 XQ3.1

▼ suggested-image-file-extension [altova:]

`altova:suggested-image-file-extension(Base64String` *als string*`)` als **`string?`** XP3.1 XQ3.1
Erhält die Base64-Kodierung einer Bilddatei als Argument und gibt die darin enthaltene Dateierweiterung des Bilds zurück. Der Rückgabewert ist ein Vorschlag, basierend auf den in der Kodierung enthaltenen Bilddateitypinformationen. Wenn diese Informationen nicht verfügbar sind, wird ein leerer String zurückgegeben. Diese Funktion ist nützlich, wenn Sie ein Base64-Bild als Datei speichern und die entsprechende Dateierweiterung dynamisch abrufen möchten.

▣ *Beispiele*

- **`altova:suggested-image-file-extension`**`(/MyImages/MobilePhone/Image20141130.01)` gibt `'jpg'` zurück
- **`altova:suggested-image-file-extension`**`($XML1/Staff/Person/@photo)` gibt `''` zurück

In den Beispielen oben wird von den als Argument der Funktion bereitgestellten Nodes angenommen, dass sie ein Base64-kodiertes Bild enthalten. Im ersten Beispiel wird `jpg` als Dateityp bzw. Dateierweiterung abgerufen. Im zweiten Beispiel enthält die angegebene Base54-Kodierung keine brauchbaren Dateierweiterungsinformationen.

image-exif-data [altova:]

altova:image-exif-data(Base64BinaryString *als string*) als **element? XP3.1 XQ3.1**

Erhält ein Base64-kodiertes JPEG-Bild als Argument und gibt ein Element namens **Exif** zurück, das die Exif-Metadaten des Bilds enthält. Die Exif-Metadaten werden als Attribut-Wert-Paare des `Exif`-Elements erstellt. Bei den Attributnamen handelt es sich um die Exif-Daten-Tags aus der Base64-Kodierung. Weiter unten sehen Sie eine Liste der Exif-Tags. Wenn die Exif-Daten einen anbieterspezifischen Tag enthalten, so wird auch dieser Tag und sein Wert als Attribut-Wert-Paar zurückgegeben. Zusätzlich zu den Standard-Exif-Metadatentags (*siehe Liste unten*) werden auch Altova-spezifische Attribut-Wert-Paare generiert. Diese Altova Exif-Attribute sind unten aufgelistet.

Beispiele

- Um ein einziges Attribut abzurufen, verwenden Sie die Funktion folgendermaßen:
  ```
  image-exif-data(//MyImages/Image20141130.01)/@GPSLatitude
  image-exif-data(//MyImages/Image20141130.01)/@Geolocation
  ```
- Um alle Attribute abzurufen, verwenden Sie die Funktion folgendermaßen:
  ```
  image-exif-data(//MyImages/Image20141130.01)/@*
  ```
- Um die Namen aller Attribute abzurufen, verwenden Sie den folgenden Ausdruck:
  ```
  for $i in image-exif-data(//MyImages/Image20141130.01)/@* return name($i)
  ```
  Auf diese Art können Sie die Namen der von der Funktion zurückgegebenen Attribute eruieren.

Altova Exif-Attribut: Geolocation

Der Altova XPath/XQuery-Prozessor generiert anhand der Exif-Standard-Metadaten-Tags das benutzerdefinierte Attribut **Geolocation**. **Geolocation** ist eine Verkettung von vier Exif-Tags: `GPSLatitude`, `GPSLatitudeRef`, `GPSLongitude`, `GPSLongitudeRef` mit hinzugefügten Werten (*siehe Tabelle unten*).

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151° 13'11.73"E

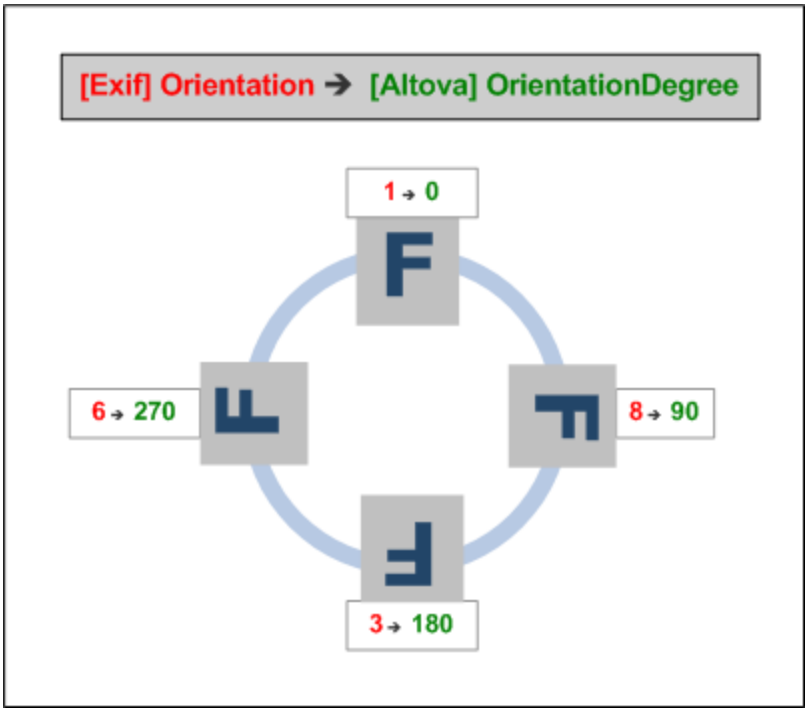
Altova Exif-Attribut: OrientationDegree

Der Altova XPath/XQuery-Prozessor generiert anhand des Exif-Metadaten-Tags **Orientation** das benutzerdefinierte Attribut **OrientationDegree**.

OrientationDegree übersetzt den Standard-Exif-Tag `Orientation` von einem Ganzzahlwert (`1`, `8`, `3` oder `6`) in die entsprechenden Gradwerte dafür (`0`, `90`, `180`, `270`) (*siehe Abbildung unten*). Beachten Sie dass es keine Übersetzung der `Orientation`-Werte `2`, `4`, `5`, `7` gibt. (Diese Ausrichtungen

© 2016-2022 Altova GmbH

werden durch Spiegelung des Bilds `1` an seiner senkrechten Mittelachse zur Erzeugung des Bilds mit dem Wert `2` und anschließende Drehung dieses Bilds um jeweils 90 Grad zur Erzeugung der Werte `7` bzw. `4` bzw. `5` erzielt).



- *Liste der Standard-Exif-Metatags*
  - `ImageWidth`
  - `ImageLength`
  - `BitsPerSample`
  - `Compression`
  - `PhotometricInterpretation`
  - `Orientation`
  - `SamplesPerPixel`
  - `PlanarConfiguration`
  - `YCbCrSubSampling`
  - `YCbCrPositioning`
  - `XResolution`
  - `YResolution`
  - `ResolutionUnit`
  - `StripOffsets`
  - `RowsPerStrip`
  - `StripByteCounts`
  - `JPEGInterchangeFormat`
  - `JPEGInterchangeFormatLength`
  - `TransferFunction`
  - `WhitePoint`
  - `PrimaryChromaticities`
  - `YCbCrCoefficients`
  - `ReferenceBlackWhite`

- DateTime
- ImageDescription
- Make
- Model
- Software
- Artist
- Copyright

- ExifVersion
- FlashpixVersion
- ColorSpace
- ComponentsConfiguration
- CompressedBitsPerPixel
- PixelXDimension
- PixelYDimension
- MakerNote
- UserComment
- RelatedSoundFile
- DateTimeOriginal
- DateTimeDigitized
- SubSecTime
- SubSecTimeOriginal
- SubSecTimeDigitized
- ExposureTime
- FNumber
- ExposureProgram
- SpectralSensitivity
- ISOSpeedRatings
- OECF
- ShutterSpeedValue
- ApertureValue
- BrightnessValue
- ExposureBiasValue
- MaxApertureValue
- SubjectDistance
- MeteringMode
- LightSource
- Flash
- FocalLength
- SubjectArea
- FlashEnergy
- SpatialFrequencyResponse
- FocalPlaneXResolution
- FocalPlaneYResolution
- FocalPlaneResolutionUnit
- SubjectLocation
- ExposureIndex
- SensingMethod
- FileSource
- SceneType
- CFAPattern
- CustomRendered
- ExposureMode
- WhiteBalance

 © 2016-2022 Altova GmbH

- DigitalZoomRatio
- FocalLengthIn35mmFilm
- SceneCaptureType
- GainControl
- Contrast
- Saturation
- Sharpness
- DeviceSettingDescription
- SubjectDistanceRange
- ImageUniqueID

- GPSVersionID
- GPSLatitudeRef
- GPSLatitude
- GPSLongitudeRef
- GPSLongitude
- GPSAltitudeRef
- GPSAltitude
- GPSTimeStamp
- GPSSatellites
- GPSStatus
- GPSMeasureMode
- GPSDOP
- GPSSpeedRef
- GPSSpeed
- GPSTrackRef
- GPSTrack
- GPSImgDirectionRef
- GPSImgDirection
- GPSMapDatum
- GPSDestLatitudeRef
- GPSDestLatitude
- GPSDestLongitudeRef
- GPSDestLongitude
- GPSDestBearingRef
- GPSDestBearing
- GPSDestDistanceRef
- GPSDestDistance
- GPSProcessingMethod
- GPSAreaInformation
- GPSDateStamp
- GPSDifferential

[Nach oben 2179]

30.2.1.5 XPath/XQuery-Funktionen: Numerische

Die numerischen Erweiterungsfunktionen von Altova können in XPath- und XQuery-Ausdrücken verwendet werden und stellen zusätzliche Funktionen für die Verarbeitung von Daten zur Verfügung. Die Funktionen in

diesem Abschnitt können mit dem **XPath 3.0- und XQuery 3.0**-Prozessor von Altova verwendet werden. Sie stehen im Zusammenhang mit XPath/XQuery zur Verfügung.

Anmerkung zur Benennung von Funktionen und zur Anwendbarkeit der Sprache

Altova-Erweiterungsfunktionen können in XPath/XQuery-Ausdrücken verwendet werden. Dadurch stehen neben den Funktionen in der Standardbibliothek der XPath-, XQuery- und XSLT-Funktionen zusätzliche Funktionen zur Verfügung. Die Altova-Erweiterungsfunktionen befinden sich im **Altova-Erweiterungsfunktions-Namespace, `http://www.altova.com/xslt-extensions`** und sind in diesem Abschnitt mit dem Präfix **`altova:`**, das als an diesen Namespace gebunden angenommen wird, gekennzeichnet. Beachten Sie, dass manche Funktionen in zukünftigen Versionen Ihres Produkts eventuell nicht mehr unterstützt werden oder dass sich das Verhalten einzelner Funktionen ändern kann. Um zu sehen, welche Altova-Erweiterungsfunktionen unterstützt werden, lesen Sie bitte die Dokumentation zur jeweiligen Release.

XPath-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XP1 XP2 XP3.1
XSLT-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XSLT1 XSLT2 XSLT3
XQuery-Funktionen (in XQuery-Ausdrücken in XQuery verwendet):	XQ1 XQ3.1

Funktionen zur automatischen Nummerierung

generate-auto-number [altova:]

`altova:generate-auto-number(ID` *`als xs:string`*`, `**`StartsWith`** *`als xs:double`*`, `**`Increment`** *`als xs:double`*`, `**`ResetOnChange`** *`als xs:string`*`)` als `xs:integer` XP1 XP2 XQ1 XP3.1 XQ3.1

Generiert jedes Mal, wenn die Funktion aufgerufen wird, eine Zahl. Die erste Zahl, die beim ersten Aufruf der Funktion generiert wird, wird durch das Argument `StartsWith` definiert. Bei jedem erneuten Aufruf der Funktion wird eine neue Zahl generiert. Diese Zahl wird durch den im Argument `Increment` definierten Wert anhand der zuvor generierten Zahl inkrementiert. Auf diese Art erstellt die Funktion `altova:generate-auto-number` einen Zähler, dessen Name durch das Argument `ID` definiert wird und der jedes Mal, wenn die Funktion aufgerufen wird, inkrementiert wird. Wenn sich der Wert des Arguments `ResetOnChange` seit dem vorherigen Funktionsaufruf geändert hat, so wird der Wert der zu generierenden Zahl auf den Wert `StartsWith` zurückgesetzt. Die Automatische Nummerierung kann auch mit der Funktion `altova:reset-auto-number` zurückgesetzt werden.

Beispiele

- **`altova:generate-auto-number`**`("ChapterNumber", 1, 1, "SomeString")` gibt bei jedem Aufruf der Funktion eine einzige Zahl beginnend mit `1` zurück, die bei jedem Aufruf der Funktion um `1` inkrementiert wird. Solange das vierte Argument in jedem anschließenden Aufruf `"SomeString"` bleibt, wird die Inkrementierung fortgesetzt. Wenn sich der Wert des vierten Arguments ändert, wird der Zähler (namens `ChapterNumber`) auf 1 zurückgesetzt. Der Wert von `ChapterNumber` kann auch folgendermaßen durch Aufruf der Funktion `altova:reset-auto-number` zurückgesetzt werden: `altova:reset-auto-number("ChapterNumber")`.

reset-auto-number [altova:]

`altova:reset-auto-number(ID` *`als xs:string`*`)` XP1 XP2 XQ1 XP3.1 XQ3.1

Diese Funktion setzt die Zahl des im `ID`-Argument angegebenen Zählers zur automatischen Nummerierung zurück. Die Zahl wird auf die Zahl zurückgesetzt, die durch das Argument `StartsWith` der Funktion `altova:generate-auto-number`, die den im `ID`-Argument genannten Zähler erstellt hat, definiert

© 2016-2022 Altova GmbH

ist

Beispiele

- `altova:reset-auto-number("ChapterNumber")` setzt die Zahl des Zählers zur automatischen Nummerierung (`ChapterNumber`), der durch die Funktion `altova:generate-auto-number` erstellt wurde, zurück. Die Zahl wird auf den Wert des Arguments `StartsWith` der Funktion `altova:generate-auto-number`, die `ChapterNumber` erstellt hat, zurückgesetzt.

[Nach oben 2183]

Numerische Funktionen

hex-string-to-integer [altova:]

`altova:hex-string-to-integer(HexString` *`als xs:string`*`)` als **`xs:integer`** **XP3.1** **XQ3.1**
Verwendet ein String-Argument, das das Base-16-Äquivalent einer Ganzzahl im Dezimalsystem (Base-10) ist, und gibt die dezimale Ganzzahl zurück.

Beispiele

- `altova:hex-string-to-integer('1')` gibt `1` zurück
- `altova:hex-string-to-integer('9')` gibt `9` zurück
- `altova:hex-string-to-integer('A')` gibt `10` zurück
- `altova:hex-string-to-integer('B')` gibt `11` zurück
- `altova:hex-string-to-integer('F')` gibt `15` zurück
- `altova:hex-string-to-integer('G')` gibt einen Fehler zurück
- `altova:hex-string-to-integer('10')` gibt `16` zurück
- `altova:hex-string-to-integer('01')` gibt `1` zurück
- `altova:hex-string-to-integer('20')` gibt `32` zurück
- `altova:hex-string-to-integer('21')` gibt `33` zurück
- `altova:hex-string-to-integer('5A')` gibt `90` zurück
- `altova:hex-string-to-integer('USA')` gibt einen Fehler zurück

integer-to-hex-string [altova:]

`altova:integer-to-hex-string(Integer` *`as xs:integer`*`)` **`as xs:string`** **XP3.1** **XQ3.1**
Verwendet ein Ganzzahlargument und gibt das Base-16-Äquivalent als String zurück.

Beispiele

- `altova:integer-to-hex-string(1)` gibt `1` zurück
- `altova:integer-to-hex-string(9)` gibt `'9'` zurück
- `altova:integer-to-hex-string(10)` gibt `'A'` zurück
- `altova:integer-to-hex-string(11)` gibt `'B'` zurück
- `altova:integer-to-hex-string(15)` gibt `'F'` zurück
- `altova:integer-to-hex-string(16)` gibt `'10'` zurück
- `altova:integer-to-hex-string(32)` gibt `'20'` zurück
- `altova:integer-to-hex-string(33)` gibt `'21'` zurück
- `altova:integer-to-hex-string(90)` gibt `'5A'` zurück

[Nach oben 2183]

Funktionen zur Formatierung von Zahlen

[nach oben]

30.2.1.6 XPath/XQuery-Funktionen: Schema

Die unten aufgelisteten Altova-Erweiterungsfunktionen geben Schemainformationen zurück. Weiter unten finden Sie Beschreibungen der Funktionen zusammen mit (i) Beispielen und (ii) einer Liste von Schemakomponenten und den dazugehörigen Eigenschaften. Diese Funktionen können mit dem **XPath 3.0-** und dem **XQuery 3.0-**Prozessor von Altova verwendet werden und stehen im Zusammenhang mit XPath/XQuery zur Verfügung.

Schemainformationen aus Schema-Dokumenten
Die Funktion **altova:schema** hat zwei Argumente: eines mit null Argumenten und das andere mit zwei Argumenten. Die Funktion mit null Argumenten gibt das gesamte Schema zurück. Sie können anschließend von diesem Ausgangspunkt aus durch das Schema navigieren und zu den gewünschten Schemakomponenten gehen. Die Funktion mit zwei Argumenten gibt eine bestimmte, durch Ihren QName identifizierte Komponentenart zurück. Der Rückgabewert ist in beiden Fällen eine Funktion. Um durch die zurückgegebene Komponente zu navigieren, müssen Sie eine Eigenschaft dieser spezifischen Komponente auswählen. Wenn es sich bei der Eigenschaft um ein nicht atomares Element handelt (d.h. wenn es sich um eine Komponente handelt), können Sie weiter navigieren, indem Sie eine Eigenschaft dieser Komponente auswählen. Wenn es sich bei der ausgewählten Eigenschaft um ein atomares Element handelt, wird der Wert des Elements zurückgegeben und Sie können nicht weiter navigieren.

Anmerkung: In XPath-Ausdrücken muss das Schema in die Verarbeitungsumgebung, z.B. in XSLT mit der Anweisung **xslt:import-schema** importiert werden. In XQuery-Ausdrücken muss das Schema explizit mit Hilfe eines Schemaimports importiert werden.

Schemainformationen aus XML-Nodes
Die Funktion **altova:type** übermittelt den Node eines XML-Dokuments und gibt die Typinformationen des Node aus dem PSVI zurück.

Anmerkung zur Benennung von Funktionen und zur Anwendbarkeit der Sprache

Altova-Erweiterungsfunktionen können in XPath/XQuery-Ausdrücken verwendet werden. Dadurch stehen neben den Funktionen in der Standardbibliothek der XPath-, XQuery- und XSLT-Funktionen zusätzliche Funktionen zur Verfügung. Die Altova-Erweiterungsfunktionen befinden sich im **Altova-Erweiterungsfunktions-Namespace, http://www.altova.com/xslt-extensions** und sind in diesem Abschnitt mit dem Präfix **altova:**, das als an diesen Namespace gebunden angenommen wird, gekennzeichnet. Beachten Sie, dass manche Funktionen in zukünftigen Versionen Ihres Produkts eventuell nicht mehr unterstützt werden oder dass sich das Verhalten einzelner Funktionen ändern kann. Um zu sehen, welche Altova-Erweiterungsfunktionen unterstützt werden, lesen Sie bitte die Dokumentation zur jeweiligen Release.

XPath-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XP1 XP2 XP3.1
XSLT-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XSLT1 XSLT2 XSLT3
XQuery-Funktionen (in XQuery-Ausdrücken in XQuery verwendet):	XQ1 XQ3.1

© 2016-2022 Altova GmbH

▼ Schema (null Argumente)

altova:schema() als (function(xs:string) als item()*)? XP3.1 XQ3.1
Gibt die **schema**-Komponente als ganzes zurück. Durch Auswahl einer der Eigenschaften der **schema**-Komponente können Sie weiter in die **schema**-Komponente navigieren.

- Wenn es sich bei dieser Eigenschaft um eine Komponente handelt, können Sie durch Auswahl einer dieser Komponenteneigenschaften einen Schritt tiefer navigieren. Dieser Schritt kann wiederholt werden, um weiter in das Schema zu navigieren.
- Wenn es sich bei der Komponente um einen atomaren Wert handelt, wird der atomare Wert zurückgegeben und Sie können nicht tiefer navigieren.

Die Eigenschaften der **schema**-Komponente sind:

```
"type definitions"
"attribute declarations"
"element declarations"
"attribute group definitions"
"model group definitions"
"notation declarations"
"identity-constraint definitions"
```

Die Eigenschaften aller anderen Komponentenarten (neben **schema**) sind unten aufgelistet.

Anmerkung: In XQuery-Ausdrücken muss das Schema explizit importiert werden. In XPath-Ausdrücken muss das Schema in die Verarbeitungsumgebung, z.B. in XSLT mit der Anweisung **xslt:import** importiert worden sein.

▣ *Beispiele*

- `import schema "" at "C:\Test\ExpReport.xsd"; for $typedef in altova:schema() ("type definitions") return $typedef ("name")` gibt die Namen aller simpleTypes oder complexTypes im Schema zurück.

- `import schema "" at "C:\Test\ExpReport.xsd"; altova:schema() ("type definitions")[1]("name")` gibt den Namen des ersten aller simpleTypes oder complexTyps im Schema zurück.

Komponenten und ihre Eigenschaften

▣ Assertion

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Assertion"
test	XPath-Eigenschaftsdatensatz	

▣ Attribute Declaration

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Attribute Declaration"
name	String	Lokaler Name des Attributs
target namespace	String	Namespace URI des Attributs
type definition	SimpleType oder ComplexType	
scope	Eine Funktion mit Eigenschaften ("class":"Scope", "variety": "global" oder "local", "parent": der enthaltende ComplexTyp bzw. die enthaltende Attributgruppe)	
value constraint	Falls vorhanden, eine Funktion mit Eigenschaften ("class": "Value Constraint", "variety": "fixed" oder "default", "value": atomarer Wert, "lexical form": String. Beachten Sie, dass die Eigenschaft "value" für Namespace-sensitive Typen nicht zur Verfügung steht.	
inheritable	Boolean	

- Attribute Group Declaration

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Attribute Group Definition"
name	String	Lokaler Name der Attributgruppe
target namespace	String	Namespace URI der Attributgruppe
attribute uses	Sequenz von (Attribute Use)	
attribute wildcard	Optionale Attribut-Wildcard	

- Attribute Use

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Attribute Use"
required	Boolean	true, wenn das Attribut obligatorisch ist, false, wenn es optional ist.
value constraint	Siehe Attribute Declaration	
inheritable	Boolean	

- Attribute Wildcard

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert

© 2016-2022 Altova GmbH

kind	string	"Wildcard"
namespace constraint	Funktion mit Eigenschaften ("class": "Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": Sequenz von xs:anyURI, "disallowed names": Liste mit QNames und/oder den Strings "defined" und "definedSiblings"	
process contents	String ("strict"|"lax"|"skip")	

Complex Type

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Complex Type"
name	String	Lokaler Name des Typs (leer, wenn anonym)
target namespace	String	Namespace URI des Typs (leer, wenn anonym)
base type definition	Complex Type Definition	
final	String-Sequenz ("restriction"|"extension")	
context	Leere Sequenz (nicht implementiert)	
derivation method	String ("restriction"|"extension")	
abstract	Boolean	
attribute uses	Attribute Use-Sequenz	
attribute wildcard	Optionale Attribute Wildcard	
content type	Funktion mit Eigenschaften: ("class":"Content Type", "variety":string ("element-only"|"empty"|"mixed"|"simple"), particle: optionales Partikel, "open content": Funktion mit Eigenschaften ("class":"Open Content", "mode": string ("interleave"|"suffix"), "wildcard": Wildcard), "simple type definition": Simple Type)	
prohibited substitutions	String-Sequenz ("restriction"|"extension")	
assertions	Assertion-Sequenz	

Element Declaration

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert

kind	String	"Complex Type"
name	String	Lokaler Name des Typs (leer, wenn anonym)
target namespace	String	Namespace URI des Typs (leer, wenn anonym)
type definition	Simple Type oder Complex Type	
type table	Funktion mit Eigenschaften ("class":"Type Table", "alternatives": Type Alternative-Sequenz, "default type definition": Simple Type oder Complex Type)	
scope	Funktion mit Eigenschaften ("class":"Scope", "variety": ("global"|"local"), "parent": optionaler Complex Type)	
value constraint	siehe Attribute Declaration	
nillable	Boolean	
identity-constraint definitions	Identity Constraint-Sequenz	
substitution group affiliations	Element Declaration-Sequenz	
substitution group exclusions	String-Sequenz ("restriction"|"extension")	
disallowed substitutions	String-Sequenz ("restriction"|"extension"|"substitution")	
abstract	Boolean	

- Element Wildcard

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Wildcard"
namespace constraint	Funktion mit Eigenschaften ("class": "Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": xs:anyURI-Sequenz, "disallowed names": Liste mit QNames und/oder den Strings "defined" und "definedSiblings"	
process contents	String ("strict"|"lax"|"skip")	

- Facet

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	Der Name des Facet, z.B.

© 2016-2022 Altova GmbH

		"minLength" oder "enumeration"
value	abhängig vom Facet	Der Wert des Facet
fixed	Boolean	
typed-value	Nur für das Enumeration Facet, Array(xs:anyAtomicType*)	Ein Array, das Enumeration-Werte, von denen jeder im Allgemeinen eine Sequenz atomarer Werte sein kann, enthält. (Anmerkung: Die Eigenschaft "value" ist unabhängig vom tatsächlichen Typ für das Enumeration Facet eine String-Sequenz)

- Identity Constraint

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Identity-Constraint Definition"
name	String	Lokaler Name des Constraint
target namespace	String	Namespace URI des Constraint
identity-constraint category	String ("key"|"unique"|"keyRef")	
selector	XPath-Eigenschaftsdatensatz	
fields	Sequenz von XPath-Eigenschaftsdatensätzen	
referenced key	(nur für keyRef): Identity Constraint	Der entsprechende Key Constraint

- Model Group

Property name	Eigenschaftstyp	Eigenschaftswert
kind	String	"Model Group"
compositor	String ("sequence"|"choice"|"all")	
particles	Partikel-Sequenz	

- Model Group Definition

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Model Group Definition"
name	String	Lokaler Name der Model Group
target namespace	String	Namespace URI der Model Group
model group	Model Group	

- Notation

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Notation Declaration"
name	String	Lokaler Name der Notation
target namespace	String	Namespace URI der Notation
system identifier	anyURI	
public identifier	String	

Particle

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Particle"
min occurs	Integer	
max occurs	Integer oder String("unbounded")	
term	Element Declaration, Element Wildcard oder ModelGroup	

Simple Type

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Simple Type Definition"
name	String	Lokaler Name des Typs (leer, wenn anonym)
target namespace	String	Namespace URI des Typs (leer, wenn anonym)
final	String-Sequenz("restriction"|"extension"|"list"|"union")	
context	enthaltende Komponente	
base type definition	Simple Type	
facets	Facet-Sequenz	
fundamental facets	Leere Sequenz (nicht implementiert)	
variety	String ("atomic"|"list"|"union")	
primitive type definition	Simple Type	
item type definition	(nur für Listentypen) Simple Type	
member type definitions	(nur für Union-Typen) Simple Type-Sequenz	

Type Alternative

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert

© 2016-2022 Altova GmbH

kind	String	"Type Alternative"
test	XPath-Eigenschaftsdatensatz	
type definition	Simple Type oder Complex Type	

XPath Property Record

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
namespace bindings	Sequenz von Funktionen mit Eigenschaften ("prefix": string, "namespace": anyURI)	
default namespace	anyURI	
base URI	anyURI	Die statische Basis-UI des XPath-Ausdrucks
expression	String	Der XPath-Ausdruck als String

Schema (zwei Argumente)

altova:schema(ComponentKind *als xs:string*, **Name** *als xs:QName*) **als (function(xs:string) als item()*)? XP3.1 XQ3.1**

Gibt die im ersten Argument angegebene Komponentenart zurück, welche einen Namen hat, der mit dem im zweiten Argument angegebenen Namen übereinstimmt. Durch Auswahl einer der Eigenschaften der Komponente können Sie weiter navigieren.

- Wenn es sich bei dieser Eigenschaft um eine Komponente handelt, können Sie durch Auswahl einer dieser Komponenteneigenschaften einen Schritt tiefer navigieren. Dieser Schritt kann wiederholt werden, um weiter in das Schema zu navigieren.
- Wenn es sich bei der Komponente um einen atomaren Wert handelt, wird der atomare Wert zurückgegeben und Sie können nicht tiefer navigieren.

Anmerkung: In XQuery-Ausdrücken muss das Schema explizit importiert werden. In XPath-Ausdrücken muss das Schema in die Verarbeitungsumgebung, z.B. in XSLT mit der Anweisung **xslt:import** importiert worden sein.

Beispiele

- `import schema "" at "C:\Test\ExpReport.xsd";`
  `altova:schema("element declaration", xs:QName("OrgChart"))("type definition")`
  `("content type")("particles")[3]!.("term")("kind")`
  gibt die `kind`-Eigenschaft des Terms der dritten `particles`-Komponente zurück. Diese particles-Komponente ist ein Nachfahr der Element-Deklaration mit dem QName `QName OrgChart`.

- `import schema "" at "C:\Test\ExpReport.xsd";`
  `let $typedef := altova:schema("type definition", xs:QName("emailType"))`
  `for $facet in $typedef ("facets")`
  `return [$facet ("kind"), $facet("value")]`
  gibt für jedes **facet** jeder **emailType**-Komponente ein Array mit der Art und dem Wert des Facet zurück.

Komponenten und ihre Eigenschaften

- Assertion

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Assertion"
test	XPath-Eigenschaftsdatensatz	

- Attribute Declaration

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Attribute Declaration"
name	String	Lokaler Name des Attributs
target namespace	String	Namespace URI des Attributs
type definition	SimpleType oder ComplexType	
scope	Eine Funktion mit Eigenschaften ("class":"Scope", "variety": "global" oder "local", "parent": der enthaltende ComplexTyp bzw. die enthaltende Attributgruppe)	
value constraint	Falls vorhanden, eine Funktion mit Eigenschaften ("class": "Value Constraint", "variety": "fixed" oder "default", "value": atomarer Wert, "lexical form": String. Beachten Sie, dass die Eigenschaft "value" für Namespace-sensitive Typen nicht zur Verfügung steht.	
inheritable	Boolean	

- Attribute Group Declaration

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Attribute Group Definition"
name	String	Lokaler Name der Attributgruppe
target namespace	String	Namespace URI der Attributgruppe
attribute uses	Sequenz von (Attribute Use)	
attribute wildcard	Optionale Attribut-Wildcard	

- Attribute Use

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert

© 2016-2022 Altova GmbH

kind	String	"Attribute Use"
required	Boolean	true, wenn das Attribut obligatorisch ist, false, wenn es optional ist.
value constraint	Siehe Attribute Declaration	
inheritable	Boolean	

Attribute Wildcard

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	string	"Wildcard"
namespace constraint	Funktion mit Eigenschaften ("class": "Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": Sequenz von xs:anyURI, "disallowed names": Liste mit QNames und/oder den Strings "defined" und "definedSiblings"	
process contents	String ("strict"|"lax"|"skip")	

Complex Type

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Complex Type"
name	String	Lokaler Name des Typs (leer, wenn anonym)
target namespace	String	Namespace URI des Typs (leer, wenn anonym)
base type definition	Complex Type Definition	
final	String-Sequenz ("restriction"|"extension")	
context	Leere Sequenz (nicht implementiert)	
derivation method	String ("restriction"|"extension")	
abstract	Boolean	
attribute uses	Attribute Use-Sequenz	
attribute wildcard	Optionale Attribute Wildcard	
content type	Funktion mit Eigenschaften: ("class":"Content Type", "variety":string ("element-only"|"empty"|"mixed"|"simple"), particle: optionales Partikel, "open content": Funktion mit Eigenschaften ("class":"Open Content", "mode": string	

	("interleave"|"suffix"), "wildcard": Wildcard), "simple type definition": Simple Type)	
prohibited substitutions	String-Sequenz ("restriction"|"extension")	
assertions	Assertion-Sequenz	

Element Declaration

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Complex Type"
name	String	Lokaler Name des Typs (leer, wenn anonym)
target namespace	String	Namespace URI des Typs (leer, wenn anonym)
type definition	Simple Type oder Complex Type	
type table	Funktion mit Eigenschaften ("class":"Type Table", "alternatives": Type Alternative-Sequenz, "default type definition": Simple Type oder Complex Type)	
scope	Funktion mit Eigenschaften ("class":"Scope", "variety": ("global"|"local"), "parent": optionaler Complex Type)	
value constraint	siehe Attribute Declaration	
nillable	Boolean	
identity-constraint definitions	Identity Constraint-Sequenz	
substitution group affiliations	Element Declaration-Sequenz	
substitution group exclusions	String-Sequenz ("restriction"|"extension")	
disallowed substitutions	String-Sequenz ("restriction"|"extension"|"substitution")	
abstract	Boolean	

Element Wildcard

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Wildcard"
namespace constraint	Funktion mit Eigenschaften ("class": "Namespace Constraint", "variety":	

© 2016-2022 Altova GmbH

	"any"|"enumeration"|"not", "namespaces": xs:anyURI-Sequenz, "disallowed names": Liste mit QNames und/oder den Strings "defined" und "definedSiblings"	
process contents	String ("strict"|"lax"|"skip")	

Facet

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	Der Name des Facet, z.B. "minLength" oder "enumeration"
value	abhängig vom Facet	Der Wert des Facet
fixed	Boolean	
typed-value	Nur für das Enumeration Facet, Array(xs:anyAtomicType*)	Ein Array, das Enumeration-Werte, von denen jeder im Allgemeinen eine Sequenz atomarer Werte sein kann, enthält. (Anmerkung: Die Eigenschaft "value" ist unabhängig vom tatsächlichen Typ für das Enumeration Facet eine String-Sequenz)

Identity Constraint

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Identity-Constraint Definition"
name	String	Lokaler Name des Constraint
target namespace	String	Namespace URI des Constraint
identity-constraint category	String ("key"|"unique"|"keyRef")	
selector	XPath-Eigenschaftsdatensatz	
fields	Sequenz von XPath-Eigenschaftsdatensätzen	
referenced key	(nur für keyRef): Identity Constraint	Der entsprechende Key Constraint

Model Group

Property name	Eigenschaftstyp	Eigenschaftswert
kind	String	"Model Group"
compositor	String ("sequence"|"choice"|"all")	
particles	Partikel-Sequenz	

Model Group Definition

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Model Group Definition"
name	String	Lokaler Name der Model Group
target namespace	String	Namespace URI der Model Group
model group	Model Group	

Notation

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Notation Declaration"
name	String	Lokaler Name der Notation
target namespace	String	Namespace URI der Notation
system identifier	anyURI	
public identifier	String	

Particle

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Particle"
min occurs	Integer	
max occurs	Integer oder String("unbounded")	
term	Element Declaration, Element Wildcard oder ModelGroup	

Simple Type

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Simple Type Definition"
name	String	Lokaler Name des Typs (leer, wenn anonym)
target namespace	String	Namespace URI des Typs (leer, wenn anonym)
final	String-Sequenz("restriction"|"extension"|"list"|"union")	
context	enthaltende Komponente	
base type definition	Simple Type	
facets	Facet-Sequenz	
fundamental facets	Leere Sequenz (nicht implementiert)	

© 2016-2022 Altova GmbH

variety	String ("atomic"|"list"|"union")	
primitive type definition	Simple Type	
item type definition	(nur für Listentypen) Simple Type	
member type definitions	(nur für Union-Typen) Simple Type-Sequenz	

Type Alternative

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Type Alternative"
test	XPath-Eigenschaftsdatensatz	
type definition	Simple Type oder Complex Type	

XPath Property Record

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
namespace bindings	Sequenz von Funktionen mit Eigenschaften ("prefix": string, "namespace": anyURI)	
default namespace	anyURI	
base URI	anyURI	Die statische Basis-UI des XPath-Ausdrucks
expression	String	Der XPath-Ausdruck als String

Typ

`altova:type(Node` *`als item?`*`)` **`als (function(xs:string) als item()*)?`** **XP3.1** **XQ3.1**
Die Funktion **`altova:type`** übermittelt einen Element- oder Attribut-Node eines XML-Dokuments und gibt die Typinformationen des Node aus dem PSVI zurück.

Anmerkung: Das XML-Dokument muss eine Schema-Deklaration haben, damit das Schema referenziert werden kann.

Beispiele

- ```
  for $element in //Email
  let $type := altova:type($element)
  return $type
  ```
  gibt eine Funktion zurück, die die Typinformationen des Node `Email` enthält.

- ```
  for $element in //Email
  let $type := altova:type($element)
  return $type ("kind")
  ```
  ermittelt anhand der Typ-Komponente des Node (Simple Type oder Complex Type) den Wert der

Eigenschaft **kind** der Komponente.

Der Parameter "**_props**" gibt die Eigenschaften der ausgewählten Komponente zurück, z.B:

- ```
  for $element in //Email
  let $type := altova:type($element)
  return ($type ("kind"), $type ("_props"))
  ```
  nimmt die Typkomponente des Node `Email` (Simple Type oder Complex Type) und gibt (i) den Wert der Eigenschaft **kind** der Komponente zurück und anschließend (ii) die Eigenschaften dieser Komponente.

Komponenten und ihre Eigenschaften

Assertion

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Assertion"
test	XPath-Eigenschaftsdatensatz	

Attribute Declaration

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Attribute Declaration"
name	String	Lokaler Name des Attributs
target namespace	String	Namespace URI des Attributs
type definition	SimpleType oder ComplexType	
scope	Eine Funktion mit Eigenschaften ("class":"Scope", "variety": "global" oder "local", "parent": der enthaltende ComplexTyp bzw. die enthaltende Attributgruppe)	
value constraint	Falls vorhanden, eine Funktion mit Eigenschaften ("class": "Value Constraint", "variety": "fixed" oder "default", "value": atomarer Wert, "lexical form": String. Beachten Sie, dass die Eigenschaft "value" für Namespace-sensitive Typen nicht zur Verfügung steht.	
inheritable	Boolean	

Attribute Group Declaration

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Attribute Group Definition"

© 2016-2022 Altova GmbH
  ```

name	String	Lokaler Name der Attributgruppe
target namespace	String	Namespace URI der Attributgruppe
attribute uses	Sequenz von (Attribute Use)	
attribute wildcard	Optionale Attribut-Wildcard	

Attribute Use

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Attribute Use"
required	Boolean	true, wenn das Attribut obligatorisch ist, false, wenn es optional ist.
value constraint	Siehe Attribute Declaration	
inheritable	Boolean	

Attribute Wildcard

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	string	"Wildcard"
namespace constraint	Funktion mit Eigenschaften ("class": "Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": Sequenz von xs:anyURI, "disallowed names": Liste mit QNames und/oder den Strings "defined" und "definedSiblings"	
process contents	String ("strict"|"lax"|"skip")	

Complex Type

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Complex Type"
name	String	Lokaler Name des Typs (leer, wenn anonym)
target namespace	String	Namespace URI des Typs (leer, wenn anonym)
base type definition	Complex Type Definition	
final	String-Sequenz ("restriction"|"extension")	
context	Leere Sequenz (nicht implementiert)	
derivation method	String ("restriction"|"extension")	

abstract	Boolean	
attribute uses	Attribute Use-Sequenz	
attribute wildcard	Optionale Attribute Wildcard	
content type	Funktion mit Eigenschaften: ("class":"Content Type", "variety":string ("element-only"|"empty"|"mixed"|"simple"), particle: optionales Partikel, "open content": Funktion mit Eigenschaften ("class":"Open Content", "mode": string ("interleave"|"suffix"), "wildcard": Wildcard), "simple type definition": Simple Type)	
prohibited substitutions	String-Sequenz ("restriction"|"extension")	
assertions	Assertion-Sequenz	

Element Declaration

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Complex Type"
name	String	Lokaler Name des Typs (leer, wenn anonym)
target namespace	String	Namespace URI des Typs (leer, wenn anonym)
type definition	Simple Type oder Complex Type	
type table	Funktion mit Eigenschaften ("class":"Type Table", "alternatives": Type Alternative-Sequenz, "default type definition": Simple Type oder Complex Type)	
scope	Funktion mit Eigenschaften ("class":"Scope", "variety": ("global"|"local"), "parent": optionaler Complex Type)	
value constraint	siehe Attribute Declaration	
nillable	Boolean	
identity-constraint definitions	Identity Constraint-Sequenz	
substitution group affiliations	Element Declaration-Sequenz	
substitution group exclusions	String-Sequenz ("restriction"|"extension")	

© 2016-2022 Altova GmbH

disallowed substitutions	String-Sequenz ("restriction"|"extension"|"substitution")	
abstract	Boolean	

- Element Wildcard

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Wildcard"
namespace constraint	Funktion mit Eigenschaften ("class": "Namespace Constraint", "variety": "any"|"enumeration"|"not", "namespaces": xs:anyURI-Sequenz, "disallowed names": Liste mit QNames und/oder den Strings "defined" und "definedSiblings"	
process contents	String ("strict"|"lax"|"skip")	

- Facet

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	Der Name des Facet, z.B. "minLength" oder "enumeration"
value	abhängig vom Facet	Der Wert des Facet
fixed	Boolean	
typed-value	Nur für das Enumeration Facet, Array(xs:anyAtomicType*)	Ein Array, das Enumeration-Werte, von denen jeder im Allgemeinen eine Sequenz atomarer Werte sein kann, enthält. (Anmerkung: Die Eigenschaft "value" ist unabhängig vom tatsächlichen Typ für das Enumeration Facet eine String-Sequenz)

- Identity Constraint

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Identity-Constraint Definition"
name	String	Lokaler Name des Constraint
target namespace	String	Namespace URI des Constraint
identity-constraint category	String ("key"|"unique"|"keyRef")	
selector	XPath-Eigenschaftsdatensatz	
fields	Sequenz von XPath-Eigenschaftsdatensätzen	

referenced key	(nur für keyRef): Identity Constraint	Der entsprechende Key Constraint

Model Group

Property name	Eigenschaftstyp	Eigenschaftswert
kind	String	"Model Group"
compositor	String ("sequence"|"choice"|"all")	
particles	Partikel-Sequenz	

Model Group Definition

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Model Group Definition"
name	String	Lokaler Name der Model Group
target namespace	String	Namespace URI der Model Group
model group	Model Group	

Notation

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Notation Declaration"
name	String	Lokaler Name der Notation
target namespace	String	Namespace URI der Notation
system identifier	anyURI	
public identifier	String	

Particle

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Particle"
min occurs	Integer	
max occurs	Integer oder String("unbounded")	
term	Element Declaration, Element Wildcard oder ModelGroup	

Simple Type

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Simple Type Definition"
name	String	Lokaler Name des Typs (leer, wenn anonym)

© 2016-2022 Altova GmbH

target namespace	String	Namespace URI des Typs (leer, wenn anonym)
final	String-Sequenz("restriction"|"extension"|"list"|"union")	
context	enthaltende Komponente	
base type definition	Simple Type	
facets	Facet-Sequenz	
fundamental facets	Leere Sequenz (nicht implementiert)	
variety	String ("atomic"|"list"|"union")	
primitive type definition	Simple Type	
item type definition	(nur für Listentypen) Simple Type	
member type definitions	(nur für Union-Typen) Simple Type-Sequenz	

- Type Alternative

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
kind	String	"Type Alternative"
test	XPath-Eigenschaftsdatensatz	
type definition	Simple Type oder Complex Type	

- XPath Property Record

Eigenschaftsname	Eigenschaftstyp	Eigenschaftswert
namespace bindings	Sequenz von Funktionen mit Eigenschaften ("prefix": string, "namespace": anyURI)	
default namespace	anyURI	
base URI	anyURI	Die statische Basis-UI des XPath-Ausdrucks
expression	String	Der XPath-Ausdruck als String

30.2.1.7 XPath/XQuery-Funktionen: Sequenz

Die Sequenz-Erweiterungsfunktionen von Altova können in XPath- und XQuery-Ausdrücken verwendet werden und stellen zusätzliche Funktionen für die Verarbeitung von Daten zur Verfügung. Die Funktionen in diesem

Abschnitt können mit dem **XPath 3.0-** und **XQuery 3.0**-Prozessor von Altova verwendet werden. Sie stehen im Zusammenhang mit XPath/XQuery zur Verfügung.

Anmerkung zur Benennung von Funktionen und zur Anwendbarkeit der Sprache

Altova-Erweiterungsfunktionen können in XPath/XQuery-Ausdrücken verwendet werden. Dadurch stehen neben den Funktionen in der Standardbibliothek der XPath-, XQuery- und XSLT-Funktionen zusätzliche Funktionen zur Verfügung. Die Altova-Erweiterungsfunktionen befinden sich im **Altova-Erweiterungsfunktions-Namespace, `http://www.altova.com/xslt-extensions`** und sind in diesem Abschnitt mit dem Präfix **`altova:`**, das als an diesen Namespace gebunden angenommen wird, gekennzeichnet. Beachten Sie, dass manche Funktionen in zukünftigen Versionen Ihres Produkts eventuell nicht mehr unterstützt werden oder dass sich das Verhalten einzelner Funktionen ändern kann. Um zu sehen, welche Altova-Erweiterungsfunktionen unterstützt werden, lesen Sie bitte die Dokumentation zur jeweiligen Release.

XPath-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XP1 XP2 XP3.1
XSLT-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XSLT1 XSLT2 XSLT3
XQuery-Funktionen (in XQuery-Ausdrücken in XQuery verwendet):	XQ1 XQ3.1

attributes [altova:]

`altova:attributes(AttributeName` *`als xs:string`***`)`** als **`attribute()*`** XP3.1 XQ3.1

Gibt alle Attribute zurück, die einen lokalen Namen haben, der mit dem im Input-Argument `AttributeName` angegebenen Namen identisch ist. Die Groß- und Kleinschreibung wird bei der Suche, die entlang der `attribute::` Achse durchgeführt wird, beachtet. Das bedeutet, dass der Kontext-Node der Parent-Element-Node sein muss.

Beispiele

- **`altova:attributes`**`("MyAttribute")` gibt `MyAttribute()*` zurück

`altova:attributes(AttributeName` *`als xs:string`***`, SearchOptions`** *`als xs:string`***`)`** als **`attribute()*`** XP3.1 XQ3.1

Gibt alle Attribute zurück, die einen lokalen Namen haben, der mit dem im Input-Argument `AttributeName` angegebenen Namen identisch ist. Die Groß- und Kleinschreibung wird bei der Suche, die entlang der `attribute::` Achse durchgeführt wird, beachtet. Der Kontext-Node muss der Parent-Element-Node sein. Das zweite Argument ist ein String, der Options-Flags enthält. Zur Verfügung stehen die folgenden Flags:

`r` = wechselt zu einer Suche mittels Regular Expression; bei `AttributeName` muss es sich in diesem Fall um einen Regular Expression-Suchstring handeln;

`f` = Wenn diese Option definiert ist, liefert `AttributeName` eine vollständige Übereinstimmung; andernfalls muss `AttributeName` nur teilweise mit einem Attributnamen übereinstimmen, damit dieses Attribut zurückgegeben wird. Wenn **`f`** z.B. nicht definiert ist, gibt `MyAtt` `MyAttribute` zurück;

`i` = wechselt zu einer Suche ohne Berücksichtigung der Groß- und Kleinschreibung;

`p` = inkludiert das Namespace-Präfix in die Suche; `AttributeName` sollte in diesem Fall das Namespace-Präfix enthalten, z.B.: `altova:MyAttribute`.

Die Flags können in jeder Reihenfolge angegeben werden. Ungültige Flags erzeugen eine Fehlermeldung. Sie können ein oder mehrere Flags weglassen. Es ist auch der leere String zulässig. Das Resultat ist dasselbe wie bei Verwendung der Funktion mit nur einem Argument (*siehe vorherige Signatur*). Unzulässig ist jedoch die Verwendung einer leeren Sequenz als zweites Argument.

Beispiele

© 2016-2022 Altova GmbH

- **altova:attributes**("MyAttribute", "rfip") gibt MyAttribute()* zurück
- **altova:attributes**("MyAttribute", "pri") gibt MyAttribute()* zurück
- **altova:attributes**("MyAtt", "rip") gibt MyAttribute()* zurück
- **altova:attributes**("MyAttributes", "rfip") gibt keine Übereinstimmung zurück
- **altova:attributes**("MyAttribute", "") gibt MyAttribute()* zurück
- **altova:attributes**("MyAttribute", "Rip") gibt einen Fehler zurück, dass das Flag unbekannt ist.
- **altova:attributes**("MyAttribute",) gibt den Fehler zurück, dass das zweite Argument fehlt.

▼ elements [altova:]

altova:elements(ElementName *als xs:string***)** als **element()*** **XP3.1** **XQ3.1**
Gibt alle Elemente zurück, die einen lokalen Namen haben, der mit dem im Input-Argument ElementName angegebenen Namen identisch ist. Die Groß- und Kleinschreibung wird bei der Suche, die entlang der child:: Achse durchgeführt wird, beachtet. Der Kontext-Node muss der Parent-Node des gesuchten Elements sein.

⊟ *Beispiele*

- **altova:elements**("MyElement") gibt MyElement()* zurück

altova:elements(ElementName *als xs:string*, **SearchOptions** *als xs:string***)** als **element()*** **XP3.1** **XQ3.1**
Gibt alle Elemente zurück, die einen lokalen Namen haben, der mit dem im Input-Argument ElementName angegebenen Namen identisch ist. Die Groß- und Kleinschreibung wird bei der Suche, die entlang der child:: Achse durchgeführt wird, beachtet. Der Kontext-Node muss der Parent-Node des gesuchten Elements sein. Das zweite Argument ist ein String, der Options-Flags enthält. Zur Verfügung stehen die folgenden Flags:
r = wechselt zu einer Suche mittels Regular Expression; bei ElementName muss es sich in diesem Fall um einen Regular Expression-Suchstring handeln;
f = Wenn diese Option definiert ist, liefert ElementName eine vollständige Übereinstimmung; andernfalls muss ElementName nur teilweise mit einem Elementnamen übereinstimmen, damit dieses Element zurückgegeben wird. Wenn **f** z.B. nicht definiert ist, gibt MyElem MyElement zurück;
i = wechselt zu einer Suche ohne Berücksichtigung der Groß- und Kleinschreibung;
p = inkludiert das Namespace-Präfix in die Suche; ElementName sollte in diesem Fall das Namespace-Präfix enthalten, z.B.: altova:MyElement.
Die Flags können in jeder Reihenfolge angegeben werden. Ungültige Flags erzeugen eine Fehlermeldung. Sie können ein oder mehrere Flags weglassen. Es ist auch der leere String zulässig. Das Resultat ist dasselbe wie bei Verwendung der Funktion mit nur einem Argument (siehe vorherige Signatur). Unzulässig ist jedoch die Verwendung einer leeren Sequenz.

⊟ *Beispiele*

- **altova:elements**("MyElement", "rip") gibt MyElement()* zurück
- **altova:elements**("MyElement", "pri") gibt MyElement()* zurück
- **altova:elements**("MyElement", "") gibt MyElement()* zurück
- **altova:elements**("MyElem", "rip") gibt MyElement()* zurück
- **altova:elements**("MyElements", "rfip") gibt keine Übereinstimmung zurück
- **altova:elements**("MyElement", "Rip") gibt einen Fehler zurück, dass das Flag unbekannt ist.
- **altova:elements**("MyElement",) gibt den Fehler zurück, dass das zweite Argument fehlt.

▼ find-first [altova:]

altova:find-first((*item()), (CheckFunction(item() *als xs:boolean*))** als **item()? XP3.1 XQ3.1**

Diese Funktion verwendet zwei Argumente. Das erste Argument ist eine Sequenz von einem oder mehreren Elementen eines beliebigen Datentyps. Das zweite Argument, `Condition`, ist eine Referenz zu einer XPath-Funktion, die ein Argument erhält. (hat einen Stellenwert `1`) und einen Booleschen Wert zurückgibt. Jedes Element von **`Sequence`** wird der Reihe nach der in `Condition` referenzierten Funktion bereitgestellt. (*Beachten Sie:* Die Funktion hat ein einziges Argument.) Das erste **`Sequence`** Element, bei dem das Resultat von **`Condition true()`** ist, wird als das Ergebnis von `altova:find-first` zurückgegeben. Anschließend wird die Iteration gestoppt.

▣ *Beispiele*

- **`altova:find-first`**`(5 to 10, function($a) {$a mod 2 = 0})` gibt `xs:integer 6` zurück

Das Argument **`Condition`** referenziert die XPath 3.0 Inline-Funktion, **`function()`**, welche eine Inline-Funktion **`$a`** deklariert und diese anschließend definiert. Die einelnen Elemente im Argument **`Sequence`** von **`altova:find-first`** werden der Reihe nach an **`$a`** als sein Input-Wert übergeben. Der Input-Wert wird an der Bedingung in der Funktionsdefinition (`$a mod 2 = 0`) überprüft. Der erste Input-Wert, der diese Bedingung erfüllt, wird als das Ergebnis von **`altova:find-first`** (in diese Fall **`6`**) zurückgegeben.

- **`altova:find-first`**`((1 to 10), (function($a) {$a+3=7}))` gibt `xs:integer 4` zurück

Weitere Beispiele

Wenn die Datei **`C:\Temp\Customers.xml`** vorhanden ist:

- **`altova:find-first`**`( ("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1) )` gibt `C:\Temp\Customers.xml` zurück

Wenn die Datei **`C:\Temp\Customers.xml`** nicht vorhanden ist und **`http://www.altova.com/index.html`** vorhanden ist:

- **`altova:find-first`**`( ("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1) )` gibt `http://www.altova.com/index.html` zurück

Wenn weder die Datei **`C:\Temp\Customers.xml`** noch **`http://www.altova.com/index.html`** vorhanden ist:

- **`altova:find-first`**`( ("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1) )` gibt kein Ergebnis zurück

Anmerkungen zu den obigen Beispielen

- Die XPath 3.0-Funktion, `doc-available`, erhält ein einziges Argument, das als URI verwendet wird. Sie gibt nur dann `true` zurück, wenn unter der angegebenen URI ein Dokument-Node gefunden wird. Das Dokument unter der angegebenen URI muss daher ein XML-Dokument sein.
- Die Funktion `doc-available` kann für **`Condition`**, das zweite Argument von `altova:find-first` verwendet werden, da sie nur ein Argument erhält (Stelligkeit=1), da sie ein Element `item()` als Input erhält (ein String, der als URI verwendet wird) und einen Booleschen Wert zurückgibt.
- Beachten Sie, dass `doc-available` nur referenziert und nicht direkt aufgerufen wird. Das

© 2016-2022 Altova GmbH

angehängte Suffix `#1` gibt eine Funktion mit einer Stelligkeit 1 an. Als Ganzes bedeutet `doc-available#1`: *Verwende die Funktion doc-availabe(), welche die Stelligkeit=1 hat und übergib die einzelnen Elemente in der ersten Sequenz der Reihe nach als einziges Argument an die Funktion.* Als Ergebnis wird jeder der beiden Strings an **`doc-available()`** übergeben. Die Funktion verwendet den String als URI und überprüft, ob unter der URI ein Dokument-Node vorhanden ist. Wenn dies der Fall ist, wird **`doc-available()`** zu `true()` ausgewertet und der String wird als Ergebnis der Funktion **`altova:find-first`** zurückgegeben. *Beachten Sie zur Funktion doc-available(), dass relative Pfade relativ zu aktuellen Basis-URI aufgelöst werden. Die Basis-URI ist standardmäßig die URI des XML-Dokuments, von dem aus die Funktion geladen wird.*

▼ find-first-combination [altova:]

`altova:find-first-combination((Seq-01` *as item()**, **`(Seq-02`** *as item()**, **`(Condition( Seq-01-Item, Seq-02-Item`** *as xs:boolean*`)) as item()*`** **XP3.1** **XQ3.1**

Diese Funktion verwendet drei Argumente:

- Die ersten beiden Argumente, **`Seq-01`** und **`Seq-02`**, sind Sequenzen von einem oder mehreren Elementen eines beliebigen Datentyps.
- Das dritte Argument, **`Condition`**, ist eine Referenz auf eine XPath-Funktion, die zwei Argumente erhält (d.h. eine Stelligkeit **2** hat) und einen Booleschen Wert zurückgibt.

Die Elemente von **`Seq-01`** und **`Seq-02`** werden als die Argumente der Funktion **`Condition`** in geordneten Paaren übergeben (je ein Element aus jeder Sequenz bildet ein Paar). Die Paare sind folgendermaßen geordnet.

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X1 Y2), (X1 Y3) ... (X1 Yn), (X2 Y1), (X2 Y2) ... (Xn Yn)
```

Das erste geordnete Paar, bei dem die Funktion **`Condition`** zu **`true()`** ausgewertet wird, wird als Ergebnis von **`altova:find-first-combination`** zurückgegeben. Beachten Sie: (i) Wenn die Funktion **`Condition`** durch die bereitgestellten Argumentpaare iteriert und nicht ein einziges Mal zu **`true()`** ausgewertet wird, so gibt **`altova:find-first-combination`** *Keine Ergebnisse* zurück; (ii) Das Ergebnis von **`altova:find-first-combination`** ist immer ein Elementpaar (eines beliebigen Datentyps) oder gar kein Element.

▣ *Beispiele*

- **`altova:find-first-combination`**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32})` gibt die Sequenz `xs:integers (11, 21)` zurück
- **`altova:find-first-combination`**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33})` gibt die Sequenz `xs:integers (11, 22)` zurück
- **`altova:find-first-combination`**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 34})` gibt die Sequenz `xs:integers (11, 23)` zurück

▼ find-first-pair [altova:]

`altova:find-first-pair((Seq-01` *als item()**, **`(Seq-02`** *als item()**, **`(Condition( Seq-01-`**

Item, Seq-02-Item *als xs:boolean*)) als **item()*** **XP3.1** **XQ3.1**
Diese Funktion erhält drei Argumente:

- Die ersten beiden Argumente, **Seq-01** und **Seq-02**, sind Sequenzen von einem oder mehreren Elementen eines beliebigen Datentyps.
- Das dritte Argument, **Condition**, ist eine Referenz auf eine XPath-Funktion, die zwei Argumente erhält (d.h. eine Stelligkeit **2** hat) und einen Booleschen Wert zurückgibt.

Die Elemente von **Seq-01** und **Seq-02** werden als die Argumente der Funktion **Condition** in geordneten Paaren übergeben. Die Paare sind folgendermaßen geordnet.

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

Das erste geordnete Paar, bei dem die Funktion **Condition** zu **true()** ausgewertet wird, wird als Ergebnis von **altova:find-first-pair** zurückgegeben. Beachten Sie: (i) Wenn die Funktion **Condition** durch die bereitgestellten Argumentpaare iteriert und nicht ein einziges Mal zu **true()** ausgewertet wird, so gibt **altova:find-first-pair** *Keine Ergebnisse* zurück; (ii) Das Ergebnis von **altova:find-first-pair** ist immer ein Elementpaar (eines beliebigen Datentyps) oder gar kein Element.

Beispiele

- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32}) gibt die Sequenz xs:integers (11, 21) zurück
- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33}) gibt *Keine Ergebnisse* zurück

Beachten Sie anhand der zwei Beispiel oben, dass die Paare folgendermaßen geordnet sind: (11, 21) (12, 22) (13, 23)...(20, 30). Aus diesem Grund gibt das zweite Beispiel *Keine Ergebnisse* zurück (da keine geordnetes Paar die Summe 33 ergibt).

find-first-pair-pos [altova:]

altova:find-first-pair-pos((Seq-01 *as item()**), **(Seq-02** *as item()**), **(Condition(Seq-01-Item, Seq-02-Item** *as xs:boolean*)) **as xs:integer** **XP3.1** **XQ3.1**
Diese Funktion erhält drei Argumente:

- Die ersten beiden Argumente, **Seq-01** und **Seq-02**, sind Sequenzen von einem oder mehreren Elementen eines beliebigen Datentyps.
- Das dritte Argument, **Condition**, ist eine Referenz auf eine XPath-Funktion, die zwei Argumente erhält (d.h. eine Stelligkeit **2** hat) und einen Booleschen Wert zurückgibt.

Die Elemente von **Seq-01** und **Seq-02** werden als die Argumente der Funktion **Condition** in geordneten Paaren übergeben. Die Paare sind folgendermaßen geordnet.

```
If   Seq-01 = X1, X2, X3 ... Xn
And  Seq-02 = Y1, Y2, Y3 ... Yn
Then (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

Als Ergebnis von **altova:find-first-pair-pos** wird die Indexposition des ersten geordneten Paars, bei dem die Funktion **Condition** zu **true()** ausgewertet wird, zurückgegeben. Beachten Sie: Wenn die

Funktion **Condition** durch die bereitgestellten Argumentpaare iteriert und kein einziges Mal zu **true()** ausgewertet wird, so gibt **altova:find-first-pair-pos** *Keine Ergebnisse* zurück.

Beispiele

- **altova:find-first-pair-pos**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32}) gibt 1 zurück
- **altova:find-first-pair-pos**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33}) gibt *Keine Ergebnisse* zurück

Beachten Sie anhand der zwei Beispiel oben, dass die Paare folgendermaßen geordnet sind: (11, 21) (12, 22) (13, 23)...(20, 30). Im ersten Beispiel gibt die Funktion **Condition** bei Auswertung des ersten Paars **true()** zurück, daher wird dessen Indexposition in der Sequenz, 1, zurückgegeben. Das zweite Beispiel gibt *Keine Ergebnisse* zurück (da keine geordnetes Paar die Summe 33 ergibt).

find-first-pos [altova:]

altova:find-first-pos((*item()**)*, **(CheckFunction(item()** *als xs:boolean***))** als **xs:integer? XP3.1 XQ3.1**

Diese Funktion verwendet zwei Argumente. Das erste Argument ist eine Sequenz von einem oder mehreren Elementen eines beliebigen Datentyps. Das zweite Argument, Condition, ist eine Referenz zu einer XPath-Funktion, die ein Argument erhält. (hat einen Stellenwert 1) und einen Booleschen Wert zurückgibt. Jedes Element von **Sequence** wird der Reihe nach der in Condition referenzierten Funktion bereitgestellt. (*Beachten Sie:* Die Funktion hat ein einziges Argument.) Das erste **Sequence** Element, bei dem das Resultat von **Condition true()** ist, wird als das Ergebnis von **altova:find-first-pos** zurückgegeben. Anschließend wird die Iteration gestoppt.

Beispiele

- **altova:find-first-pos**(5 to 10, function($a) {$a mod 2 = 0}) gibt xs:integer 2 zurück

Das Argument **Condition** referenziert die XPath 3.0 Inline-Funktion, **function()**, welche eine Inline-Funktion **$a** deklariert und diese anschließend definiert. Die einelnen Elemente im Argument **Sequence** von **altova:find-first-pos** werden der Reihe nach an **$a** als sein Input-Wert übergeben. Der Input-Wert wird an der Bedingung in der Funktionsdefinition ($a mod 2 = 0) überprüft. Die Indexposition in der Sequenz des ersten Input-Werts, die diese Bedingung erfüllt, wird als das Ergebnis von **altova:find-first-pos** zurückgegeben (in diesem Fall 2, da **6,** der erste Wert in der Sequenz, der die Bedingung erfüllt, sich in der Sequenz an der Indexposition 2 befindet).

Weitere Beispiele

Wenn die Datei **C:\Temp\Customers.xml** vorhanden ist:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) gibt 1 zurück

Wenn die Datei **C:\Temp\Customers.xml** nicht vorhanden ist und **http://www.altova.com/index.html** vorhanden ist:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) gibt 2 zurück

Wenn weder die Datei **C:\Temp\Customers.xml** noch **http://www.altova.com/index.html** vorhanden ist:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) gibt kein Ergebnis zurück

Anmerkungen zu den obigen Beispielen

- Die XPath 3.0-Funktion, doc-available, erhält ein einziges Argument, das als URI verwendet wird. Sie gibt nur dann true zurück, wenn unter der angegebenen URI ein Dokument-Node gefunden wird. (Das Dokument unter der angegebenen URI muss daher ein XML-Dokument sein.)
- Die Funktion doc-available kann für **Condition,** das zweite Argument von altova:find-first-pos verwendet werden, da sie nur ein Argument erhält (Stelligkeit=1), da sie ein Element item() als Input erhält (ein String, der als URI verwendet wird) und einen Booleschen Wert zurückgibt.
- Beachten Sie, dass doc-available nur referenziert und nicht direkt aufgerufen wird. Das angehängte Suffix #1 gibt eine Funktion mit einer Stelligkeit 1 an. Als Ganzes bedeutet doc-available#1: *Verwende die Funktion doc-availabe(), welche die Stelligkeit=1 hat und übergib die einzelnen Elemente in der ersten Sequenz der Reihe nach als einziges Argument an die Funktion.* Als Ergebnis wird jeder der beiden Strings an **doc-available()** übergeben. Die Funktion verwendet den String als URI und überprüft, ob unter der URI ein Dokument-Node vorhanden ist. Wenn dies der Fall ist, wird **doc-available()** zu true() ausgewertet und der String wird als Ergebnis der Funktion **altova:find-first** zurückgegeben. *Beachten Sie zur Funktion doc-available(), dass relative Pfade relativ zu aktuellen Basis-URI aufgelöst werden. Die Basis-URI ist standardmäßig die URI des XML-Dokuments, von dem aus die Funktion geladen wird.*

for-each-attribute-pair [altova:]

altova:for-each-attribute-pair(Seq1 *als element()?*, **Seq2** *als element()?*, **Function** *als function()***)** als **item()*** **XP3.1** **XQ3.1**

Die beiden ersten Argumente identifizieren zwei Elemente, anhand deren Attribute Attributpaare gebildet werden, wobei das eine Attribut eines Paars aus dem ersten Element und das andere aus dem zweiten Element stammt. Die Attributpaare werden auf Basis ihres übereinstimmenden Namens ausgewählt und alphabetisch (nach ihren Namen) zu einer Gruppe geordnet. Falls es zu einem Attribut im anderen Element keine Entsprechung gibt, ist das Paar "nicht verbunden", d.h. es besteht nur aus einem Mitglied. Das Funktionselement (das dritte Argument Function) wird auf die einzelnen Paare (verbundene und nicht verbundene) in der Sequenz der Paare separat angewendet, wodurch als Ausgabe eine Sequenz von Einträgen erzeugt wird.

Beispiele

- **altova:for-each-attribute-pair**(/Example/Test-A, /Example/Test-B, function($a, $b) {$a+b}) gibt zurück ...

  (2, 4, 6) wenn

```
<Test-A att1="1" att2="2" att3="3" />
<Test-B att1="1" att2="2" att3="3" />
```

  (2, 4, 6) wenn

© 2016-2022 Altova GmbH

```
<Test-A att2="2" att1="1" att3="3" />
<Test-B att3="3" att2="2" att1="1" />
```

`(2, 6)` wenn

```
<Test-A att4="4" att1="1" att3="3" />
<Test-B att3="3" att2="2" att1="1" />
```

Anmerkung: *Das Ergebnis* `(2, 6)` *wird mit Hilfe der folgenden Aktion ermittelt:* `(1+1, ()+2, 3+3, 4+())`. *Wenn einer der Operanden eine leere Sequenz ist, wie dies bei Eintrag 2 und 4 der Fall ist, so ist das Ergebnis der Addition eine leere Sequenz.*

- **altova:for-each-attribute-pair**`(/Example/Test-A, /Example/Test-B, concat#2)` gibt zurück ...

  `(11, 22, 33)` wenn

  ```
  <Test-A att1="1" att2="2" att3="3" />
  <Test-B att1="1" att2="2" att3="3" />
  ```

  `(11, 2, 33, 4)` wenn

  ```
  <Test-A att4="4" att1="1" att3="3" />
  <Test-B att3="3" att2="2" att1="1" />
  ```

▼ for-each-combination [altova:]

altova:for-each-combination(FirstSequence *als item()**, **SecondSequence** *als item()**, **Function($i,$j){$i || $j})** als `item()*` XP3.1 XQ3.1

Die Elemente der zwei Sequenzen in den ersten beiden Argumenten werden miteinander kombiniert, so dass jedes Element in der ersten Sequenz der Reihe nach einmal mit jedem Element in der zweiten Sequenz kombiniert wird. Die als drittes Argument angegebene Funktion wird auf die einzelnen Kombinationen in der erzeugten Sequenz angewendet, wodurch als Ausgabe eine Sequenz von Elementen erzeugt wird (*siehe Beispiel*).

▣ *Beispiele*

- **altova:for-each-combination**`( ('a', 'b', 'c'), ('1', '2', '3'), function($i, $j) {$i || $j} )` gibt `('a1', 'a2', 'a3', 'b1', 'b2', 'b3', 'c1', 'c2', 'c3')` zurück

u

▼ for-each-matching-attribute-pair [altova:]

altova:for-each-matching-attribute-pair(Seq1 *als element()?*, **Seq2** *als element()?*, **Function** *als function()***)** als `item()*` XP3.1 XQ3.1

Die beiden ersten Argumente identifizieren zwei Elemente, anhand deren Attribute Attributpaare gebildet werden, wobei das eine Attribut eines Paars aus dem ersten Element und das andere aus dem zweiten Element stammt. Die Attributpaare werden auf Basis ihres übereinstimmenden Namens ausgewählt und alphabetisch (nach ihren Namen) zu einer Gruppe geordnet. Falls es zu einem Attribut im anderen Element keine Entsprechung gibt, wird kein Paar gebildet. Das Funktionselement (das dritte Argument `Function`) wird auf die einzelnen Paare in der Sequenz der Paare separat angewendet, wodurch als Ausgabe eine Sequenz von Einträgen erzeugt wird.

▣ *Beispiele*

- **altova:for-each-matching-attribute-pair**(/Example/Test-A, /Example/Test-B, function($a, $b){$a+b}) gibt zurück ...

  (2, 4, 6) wenn
  <Test-A att1="1" att2="2" att3="3" />
  <Test-B att1="1" att2="2" att3="3" />

  (2, 4, 6) wenn
  <Test-A att2="2" att1="1" att3="3" />
  <Test-B att3="3" att2="2" att1="1" />

  (2, 6) wenn
  <Test-A att4="4" att1="1" att3="3" />
  <Test-B att3="3" att2="2" att3="1" />

- **altova:for-each-matching-attribute-pair**(/Example/Test-A, /Example/Test-B, concat#2) gibt zurück ....

  (11, 22, 33) wenn
  <Test-A att1="1" att2="2" att3="3" />
  <Test-B att1="1" att2="2" att3="3" />

  (11, 33) wenn
  <Test-A att4="4" att1="1" att3="3" />
  <Test-B att3="3" att2="2" att1="1" />

▼ substitute-empty [altova:]

altova:substitute-empty(FirstSequence *als item()**, **SecondSequence** *als item()*) als **item()***
XP3.1 XQ3.1
Wenn FirstSequence leer ist, wird SecondSequence zurückgegeben. Wenn FirstSequence nicht leer ist, wird FirstSequence zurückgegeben.

Beispiele

- **altova:substitute-empty**((1,2,3), (4,5,6)) gibt (1,2,3) zurück
- **altova:substitute-empty**((), (4,5,6)) gibt (4,5,6)zurück

30.2.1.8 XPath/XQuery-Funktionen: String

Die folgenden XPath/XQuey-Erweiterungsfunktionen für Strings werden in der aktuellen Version Ihres Altova-Produkts unterstützt und bieten Zusatzfunktionalitäten für die Verarbeitung von Daten. Die Funktionen in diesem Abschnitt können mit dem **XPath 3.0-** und **XQuery 3.0**-Prozessor von Altova verwendet werden. Sie stehen im Zusammenhang mit XPath/XQuery zur Verfügung.

Anmerkung zur Benennung von Funktionen und zur Anwendbarkeit der Sprache

Altova-Erweiterungsfunktionen können in XPath/XQuery-Ausdrücken verwendet werden. Dadurch stehen neben den Funktionen in der Standardbibliothek der XPath-, XQuery- und XSLT-Funktionen zusätzliche

© 2016-2022 Altova GmbH

Funktionen zur Verfügung. Die Altova-Erweiterungsfunktionen befinden sich im **Altova-Erweiterungsfunktions-Namespace,** `http://www.altova.com/xslt-extensions` und sind in diesem Abschnitt mit dem Präfix `altova:`, das als an diesen Namespace gebunden angenommen wird, gekennzeichnet. Beachten Sie, dass manche Funktionen in zukünftigen Versionen Ihres Produkts eventuell nicht mehr unterstützt werden oder dass sich das Verhalten einzelner Funktionen ändern kann. Um zu sehen, welche Altova-Erweiterungsfunktionen unterstützt werden, lesen Sie bitte die Dokumentation zur jeweiligen Release.

XPath-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XP1 XP2 XP3.1
XSLT-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XSLT1 XSLT2 XSLT3
XQuery-Funktionen (in XQuery-Ausdrücken in XQuery verwendet):	XQ1 XQ3.1

camel-case [altova:]

altova:camel-case(InputString *als xs:string***)** als **xs:string** XP3.1 XQ3.1
Gibt den Input-String **InputString** in CamelCase zurück. Der String wird mit Hilfe der Regular Expression `'\s'` (welches ein Kürzel für das Leerzeichen ist) analysiert. Das erste Zeichen nach einem Leerzeichen oder einer Sequenz aufeinander folgender Leerzeichen, das kein Leerzeichen ist, wird mit einem Großbuchstaben geschrieben. Das erste Zeichen im Ausgabestring wird mit einem Großbuchstaben geschrieben.

Beispiele

- **altova:camel-case**("max") gibt Max zurück
- **altova:camel-case**("max max") gibt Max Max zurück
- **altova:camel-case**("file01.xml") gibt File01.xml zurück
- **altova:camel-case**("file01.xml file02.xml") gibt File01.xml File02.xml zurück
- **altova:camel-case**("file01.xml file02.xml") gibt File01.xml File02.xml zurück
- **altova:camel-case**("file01.xml -file02.xml") gibt File01.xml -file02.xml zurück

altova:camel-case(InputString *als xs:string*, **SplitChars** *als xs:string*, **IsRegex** *als xs:boolean***)** als **xs:string** XP3.1 XQ3.1
Konvertiert den Input-String **InputString** in CamelCase, indem anhand von **SplitChars** festgelegt wird, welche(s) Zeichen die nächste Konvertierung in Großbuchstaben auslöst. **SplitChars** wird als Regular Expression verwendet, wenn **IsRegex = true()** oder als einfache Zeichen, wenn **IsRegex = false()**. Das erste Zeichen im Ausgabestring wird mit einem Großbuchstaben geschrieben.

Beispiele

- **altova:camel-case**("setname getname", "set|get", true()) gibt setName getName zurück
- **altova:camel-case**("altova\documents\testcases", "\", false()) gibt Altova\Documents\Testcases zurück

char [altova:]

altova:char(Position *as xs:integer***)** als **xs:string** XP3.1 XQ3.1
Gibt einen String zurück, der das Zeichen an der durch das Argument Position definierten Position enthält. Dieses Zeichen wird durch Konvertierung des Werts des Kontextelements in xs:string ermittelt. Der Ergebnisstring ist leer, wenn an dem durch das Position Argument gelieferten Index kein Zeichen vorhanden ist.

Beispiele

Wenn das Kontextelement **1234ABCD** lautet:

- **altova:char**(2) gibt 2 zurück
- **altova:char**(5) gibt A zurück
- **altova:char**(9) gibt den leeren String zurück.
- **altova:char**(-2) gibt den leeren String zurück.

altova:char(InputString *als xs:string*, **Position** *als xs:integer*) als **xs:string** **XP3.1** **XQ3.1**
Gibt einen String zurück, der das Zeichen enthält, das sich in dem als InputString Argument gelieferten String an der durch das Argument Position definierten Position befindet. Der Ergebnisstring ist leer, wenn an dem durch das Position Argument gelieferten Index kein Zeichen vorhanden ist.

Beispiele

- **altova:char**("2014-01-15", 5) gibt - zurück
- **altova:char**("USA", 1) gibt U zurück
- **altova:char**("USA", 1) gibt den leeren String zurück.
- **altova:char**("USA", -2) gibt den leeren String zurück.

create-hash-from-string[altova:]

altova:create-hash-from-string(InputString *als xs:string*) als **xs:string** **XP2** **XQ1** **XP3.1** **XQ3.1**
altova:create-hash-from-string(InputString *als xs:string*, **HashAlgo** *als xs:string*) als **xs:string** **XP2** **XQ1** **XP3.1** **XQ3.1**
Generiert anhand von InputString mit Hilfe des durch das Argument HashAlgo definierten Hash-Algorithmus einen Hash-String. Es können die folgenden Hash-Algorithmen definiert werden (in Groß- oder Kleinbuchstaben): **MD5**, **SHA-1**, **SHA-224**, **SHA-256**, **SHA-384**, **SHA-512**. Wenn das zweite Argument nicht definiert ist (*siehe erste Signatur oben*), wird der Hash-Algorithmus **SHA-256** verwendet.

Beispiele

- **altova:create-hash-from-string**('abc') gibt einen Hash-String zurück, der mit Hilfe des Hash-Algorithmus **SHA-256** generiert wurde.
- **altova:create-hash-from-string**('abc', 'md5') gibt einen Hash-String zurück, der mit Hilfe des Hash-Algorithmus **MD5** generiert wurde.
- **altova:create-hash-from-string**('abc', 'MD5') gibt einen Hash-String zurück, der mit Hilfe des Hash-Algorithmus **MD5** generiert wurde.

first-chars [altova:]

altova:first-chars(X-Number *as xs:integer*) als **xs:string** **XP3.1** **XQ3.1**
Gibt einen String zurück, der die ersten x Zeichen (bezeichnet durch X-Number) des String enthält, der durch Konvertierung des Werts des Kontextelements in xs:string erzeugt wird.

Beispiele

Wenn das Kontextelement **1234ABCD** lautet:

- **altova:first-chars**(2) gibt 12 zurück
- **altova:first-chars**(5) gibt 1234A zurück
- **altova:first-chars**(9) gibt 1234ABCD zurück

© 2016-2022 Altova GmbH

altova:first-chars(InputString *als xs:string*, **X-Number** *als xs:integer*) als **xs:string** **XP3.1** **XQ3.1**
Gibt einen String zurück, der die ersten x Zeichen (bezeichnet durch `X-Number`) des String enthält, das als das Argument `InputString` angegeben ist.

Beispiele

- **altova:first-chars**("2014-01-15", 5) gibt 2014- zurück
- **altova:first-chars**("USA", 1) gibt U zurück

format-string [altova:]

altova:format-string(InputString *als xs:string*, **FormatSequence** *als item()**) als **xs:string** **XP3.1** **XQ3.1**
Der Input String (erstes Argument) enthält Positionsparameter (`%1`, `%2`, usw.). Jeder Parameter wird durch das String-Element ersetzt, das sich in der (als zweites Argument bereitgestellten) Formatsequenz an der entsprechenden Position befindet. Daher ersetzt des erste Element in der Formatsequenz den Positionsparameter `%1`, das zweite den Positionsparameter `%2`, usw. Die Funktion gibt diesen formatierten String zurück, der die Ersetzungen enthält. Wenn für einen Positionsparameter kein String existiert, wird der Positionsparameter selbst zurückgegeben. Dies kommt vor, wenn der Index eines Positionsparameters größer als die Anzahl der Elemente in der Formatsequenz ist.

Beispiele

- **altova:format-string**('Hello %1, %2, %3', ('Jane','John','Joe')) gibt "Hello Jane, John, Joe" zurück.
- **altova:format-string**('Hello %1, %2, %3', ('Jane','John','Joe', 'Tom')) gibt "Hello Jane, John, Joe" zurück.
- **altova:format-string**('Hello %1, %2, %4', ('Jane','John','Joe', 'Tom')) gibt "Hello Jane, John, Tom" zurück.
- **altova:format-string**('Hello %1, %2, %4', ('Jane','John','Joe')) gibt "Hello Jane, John, %4" zurück.

last-chars [altova:]

altova:last-chars(X-Number *als xs:integer*) als **xs:string** **XP3.1** **XQ3.1**
Gibt einen String zurück, der die letzten x Zeichen (bezeichnet durch `X-Number`) des String enthält, der durch Konvertierung des Werts des Kontextelements in `xs:string` erzeugt wird.

Beispiele

Wenn das Kontextelement **1234ABCD** lautet:

- **altova:last-chars**(2) gibt CD zurück
- **altova:last-chars**(5) gibt 4ABCD zurück
- **altova:last-chars**(9) gibt 1234ABCD zurück

altova:last-chars(InputString *als xs:string*, **X-Number** *als xs:integer*) als **xs:string** **XP3.1** **XQ3.1**

Gibt einen String zurück, der die letzten x Zeichen (bezeichnet durch `X-Number`) des String enthält, das als das Argument `InputString` angegeben ist.

Beispiele

- `altova:last-chars("2014-01-15", 5)` gibt `01-15-` zurück
- `altova:last-chars("USA", 10)` gibt `USA` zurück

▼ pad-string-left [altova:]

`altova:pad-string-left(StringToPad` *`als xs:string`*`, `**`Repeats`** *`als xs:integer`*`, `**`PadCharacter`** *`als xs:string`*`)` als **`xs:string`** **XP3.1** **XQ3.1**

Das Argument `PadCharacter` ist ein einzelnes Zeichen. Es wird links vom String als Auffüllzeichen eingefügt, um die Anzahl der Zeichen in `StringToPad` zu erhöhen, damit diese Anzahl dem Ganzzahlwert des Arguments `StringLength` entspricht. Das Argument `StringLength` kann jeden beliebigen (positiven oder negativen) Ganzzahlwert haben, Auffüllzeichen werden aber nur verwendet, wenn der Wert von `StringLength` größer als die Anzahl der Zeichen in `StringToPad` ist. Wenn `StringToPad` mehr Zeichen als der Wert von `StringLength` hat, bleibt `StringToPad` unverändert.

Beispiele

- `altova:pad-string-left('AP', 1, 'Z')` gibt `'AP'` zurück
- `altova:pad-string-left('AP', 2, 'Z')` gibt `'AP'` zurück
- `altova:pad-string-left('AP', 3, 'Z')` gibt `'ZAP'` zurück
- `altova:pad-string-left('AP', 4, 'Z')` gibt `'ZZAP'` zurück
- `altova:pad-string-left('AP', -3, 'Z')` gibt `'AP'` zurück
- `altova:pad-string-left('AP', 3, 'YZ')` gibt einen Fehler zurück, dass das Auffüllzeichen zu lang ist

▼ pad-string-right [altova:]

`altova:pad-string-right(StringToPad` *`als xs:string`*`, `**`Repeats`** *`als xs:integer`*`, `**`PadCharacter`** *`als xs:string`*`)` als **`xs:string`** **XP3.1** **XQ3.1**

Das Argument `PadCharacter` ist ein einzelnes Zeichen. Es wird rechts vom String als Auffüllzeichen eingefügt, um die Anzahl der Zeichen in `StringToPad` zu erhöhen, damit diese Anzahl dem Ganzzahlwert des Arguments `StringLength` entspricht. Das Argument `StringLength` kann jeden beliebigen (positiven oder negativen) Ganzzahlwert haben, Auffüllzeichen werden aber nur verwendet, wenn der Wert von `StringLength` größer als die Anzahl der Zeichen in `StringToPad` ist. Wenn `StringToPad` mehr Zeichen als der Wert von `StringLength` hat, bleibt `StringToPad` unverändert.

Beispiele

- `altova:pad-string-right('AP', 1, 'Z')` gibt `'AP'` zurück
- `altova:pad-string-right('AP', 2, 'Z')` gibt `'AP'` zurück
- `altova:pad-string-right('AP', 3, 'Z')` gibt `'ZAP'` zurück
- `altova:pad-string-right('AP', 4, 'Z')` gibt `'ZZAP'` zurück
- `altova:pad-string-right('AP', -3, 'Z')` gibt `'AP'` zurück
- `altova:pad-string-right('AP', 3, 'YZ')` gibt einen Fehler zurück, dass das Auffüllzeichen zu lang ist

▼ repeat-string [altova:]

`altova:repeat-string(InputString` *`als xs:string`*`, `**`Repeats`** *`als xs:integer`*`)` als **`xs:string`** **XP2** **XQ1** **XP3.1** **XQ3.1**

Generiert einen String, der sich zusammensetzt aus dem ersten `InputString`-Argument, das die Anzahl der `Repeats` wiederholt wird.

Beispiele

- **altova:repeat-string**("Altova #", 3) gibt "Altova #Altova #Altova #" zurück

substring-after-last [altova:]

altova:substring-after-last(MainString *als xs:string*, **CheckString** *als xs:string*) als **xs:string** **XP3.1** **XQ3.1**

Falls in MainString CheckString gefunden wird, so wird der Substring zurückgegeben, der in MainString nach CheckString steht. Falls CheckString in MainString nicht gefunden wird, so wird der leere String zurückgegeben. Wenn CheckString ein leerer String ist, so wird der gesamte MainString zurückgegeben. Falls CheckString mehrmals in MainString, vorkommt, so wird der Substring nach der letzten Instanz von CheckString zurückgegeben.

Beispiele

- **altova:substring-after-last**('ABCDEFGH', 'B') gibt 'CDEFGH' zurück
- **altova:substring-after-last**('ABCDEFGH', 'BC') gibt 'DEFGH'zurück
- **altova:substring-after-last**('ABCDEFGH', 'BD') gibt '' zurück
- **altova:substring-after-last**('ABCDEFGH', 'Z') gibt '' zurück
- **altova:substring-after-last**('ABCDEFGH', '') gibt 'ABCDEFGH' zurück
- **altova:substring-after-last**('ABCD-ABCD', 'B') gibt 'CD' zurück
- **altova:substring-after-last**('ABCD-ABCD-ABCD', 'BCD') gibt '' zurück

substring-before-last [altova:]

altova:substring-before-last(MainString *as xs:string*, **CheckString** *as xs:string*) **as xs:string** **XP3.1** **XQ3.1**

Falls in MainString CheckString gefunden wird, so wird der Substring zurückgegeben, der in MainString vor CheckString steht. Falls CheckString in MainString nicht gefunden wird, so wird der leere String zurückgegeben. Wenn CheckString ein leerer String ist, so wird der gesamte MainString zurückgegeben. Falls CheckString mehrmals in MainString, vorkommt, so wird der Substring vor der letzten Instanz von CheckString zurückgegeben.

Beispiele

- **altova:substring-before-last**('ABCDEFGH', 'B') gibt 'A' zurück
- **altova:substring-before-last**('ABCDEFGH', 'BC') gibt 'A' zurück
- **altova:substring-before-last**('ABCDEFGH', 'BD') gibt '' zurück
- **altova:substring-before-last**('ABCDEFGH', 'Z') gibt '' zurück
- **altova:substring-before-last**('ABCDEFGH', '') gibt '' zurück
- **altova:substring-before-last**('ABCD-ABCD', 'B') gibt 'ABCD-A' zurück
- **altova:substring-before-last**('ABCD-ABCD-ABCD', 'ABCD') gibt 'ABCD-ABCD-' zurück

substring-pos [altova:]

altova:substring-pos(StringToCheck *als xs:string*, **StringToFind** *als xs:string*) als **xs:integer** **XP3.1** **XQ3.1**

Gibt die Zeichenposition der ersten Instanz von StringToFind im String StringToCheck zurück. Die Zeichenposition wird in Form einer Ganzzahl angegeben. Das erste Zeichen von StringToCheck hat die Position 1. Wenn StringToFind in StringToCheck nicht vorkommt, wird die Ganzzahl 0 zurückgegeben.

Um den String auf eine zweite oder eine weiter hinten folgende Instanz von `StringToCheck` zu überprüfen, verwenden Sie die nächste Signatur dieser Funktion.

Beispiele

- **altova:substring-pos**('Altova', 'to') gibt 3 zurück
- **altova:substring-pos**('Altova', 'tov') gibt 3 zurück
- **altova:substring-pos**('Altova', 'tv') gibt 0 zurück
- **altova:substring-pos**('AltovaAltova', 'to') gibt 3 zurück

altova:substring-pos(StringToCheck *als xs:string*, **StringToFind** *als xs:string*, **Integer** *als xs:integer*) als **xs:integer** XP3.1 XQ3.1

Gibt die Zeichenposition von `StringToFind` im String `StringToCheck` zurück. Die Suche nach `StringToFind` beginnt an der durch das Argument `Integer` angegebenen Zeichenposition; der Zeichen-Substring vor dieser Position wird nicht durchsucht. Die zurückgegebene Ganzzahl gibt jedoch die Position des gefundenen String innerhalb des *gesamten* String `StringToCheck` an. Diese Signatur dient dazu, die zweite oder eine weiter hinten folgende Position eines String zu finden, der mehrmals in `StringToCheck` vorkommt. Wenn `StringToFind` in `StringToCheck` nicht vorkommt, wird die Ganzzahl `0` zurückgegeben.

Beispiele

- **altova:substring-pos**('Altova', 'to', 1) gibt 3 zurück
- **altova:substring-pos**('Altova', 'to', 3) gibt 3 zurück
- **altova:substring-pos**('Altova', 'to', 4) gibt 0 zurück
- **altova:substring-pos**('Altova-Altova', 'to', 0) gibt 3 zurück
- **altova:substring-pos**('Altova-Altova', 'to', 4) gibt 10 zurück

trim-string [altova:]

altova:trim-string(InputString *als xs:string*) als **xs:string** XP3.1 XQ3.1

Diese Funktion verwendet ein `xs:string` Argument, entfernt alle voran- und nachgestellten Leerzeichen und gibt einen "getrimmten" `xs:string` zurück.

Beispiele

- **altova:trim-string**(" Hello World ") gibt "Hello World" zurück
- **altova:trim-string**("Hello World ") gibt "Hello World" zurück
- **altova:trim-string**(" Hello World") gibt "Hello World" zurück
- **altova:trim-string**("Hello World") gibt "Hello World" zurück
- **altova:trim-string**("Hello World") gibt "Hello World" zurück

trim-string-left [altova:]

altova:trim-string-left(InputString *als xs:string*) als **xs:string** XP3.1 XQ3.1

Diese Funktion verwendet ein `xs:string` Argument, entfernt alle vorangestellten Leerzeichen und gibt einen "links getrimmten" `xs:string` zurück.

Beispiele

- **altova:trim-string-left**(" Hello World ") gibt "Hello World " zurück
- **altova:trim-string-left**("Hello World ") gibt "Hello World " zurück
- **altova:trim-string-left**(" Hello World") gibt "Hello World" zurück
- **altova:trim-string-left**("Hello World") gibt "Hello World" zurück

© 2016-2022 Altova GmbH

- **altova:trim-string-left**("Hello World") gibt "Hello World" zurück

▼ trim-string-right [altova:]

altova:trim-string-right(InputString *als xs:string***)** als **xs:string** XP3.1 XQ3.1
Diese Funktion verwendet ein xs:string Argument, entfernt alle nachgestellten Leerzeichen und gibt einen "rechts getrimmten" xs:string zurück.

Beispiele

- **altova:trim-string-right**(" Hello World ") gibt " Hello World" zurück
- **altova:trim-string-right**("Hello World ") gibt "Hello World" zurück
- **altova:trim-string-right**(" Hello World") gibt " Hello World" zurück
- **altova:trim-string-right**("Hello World") gibt "Hello World" zurück
- **altova:trim-string-right**("Hello World") gibt "Hello World" zurück

30.2.1.9 XPath/XQuery-Funktionen: Diverse Funktionen

Die folgenden XPath/XQuery-Funktionen für allgemeine Zwecke werden in der aktuellen Version von XMLSpy unterstützt und können in (i) in einem XSLT-Kontext in XPath-Ausdrücken oder (ii) in einem XQuery-Dokument in XQuery-Ausdrücken verwendet werden.

Anmerkung zur Benennung von Funktionen und zur Anwendbarkeit der Sprache

Altova-Erweiterungsfunktionen können in XPath/XQuery-Ausdrücken verwendet werden. Dadurch stehen neben den Funktionen in der Standardbibliothek der XPath-, XQuery- und XSLT-Funktionen zusätzliche Funktionen zur Verfügung. Die Altova-Erweiterungsfunktionen befinden sich im **Altova-Erweiterungsfunktions-Namespace, http://www.altova.com/xslt-extensions** und sind in diesem Abschnitt mit dem Präfix **altova:**, das als an diesen Namespace gebunden angenommen wird, gekennzeichnet. Beachten Sie, dass manche Funktionen in zukünftigen Versionen Ihres Produkts eventuell nicht mehr unterstützt werden oder dass sich das Verhalten einzelner Funktionen ändern kann. Um zu sehen, welche Altova-Erweiterungsfunktionen unterstützt werden, lesen Sie bitte die Dokumentation zur jeweiligen Release.

XPath-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XP1 XP2 XP3.1
XSLT-Funktionen (in XPath-Ausdrücken in XSLT verwendet):	XSLT1 XSLT2 XSLT3
XQuery-Funktionen (in XQuery-Ausdrücken in XQuery verwendet):	XQ1 XQ3.1

▼ decode-string [altova:]

altova:decode-string(Input *als xs:base64Binary***) als xs:string** XP3.1 XQ3.1
altova:decode-string(Input *als xs:base64Binary***, Encoding** *als xs:string***) als xs:string** XP3.1 XQ3.1
Dekodiert den angegebenen base64Binary-Input anhand der definierten Kodierung zu einem String. Wenn keine Kodierung definiert ist, wird die UTF-8-Kodierung verwendet. Die folgenden Kodierungen werden unterstützt: US-ASCII, ISO-8859-1, UTF-16, UTF-16LE, UTF-16BE, ISO-10646-UCS2, UTF-32,

UTF-32LE, UTF-32BE, ISO-10646-UCS4

Beispiele

- **altova:decode-string**($XML1/MailData/Meta/b64B) gibt den base64Binary-Input als UTF-8-kodierten String zurück.
- **altova:decode-string**($XML1/MailData/Meta/b64B, "UTF-8") gibt den base64Binary-Input als UTF-8-kodierten String zurück.
- **altova:decode-string**($XML1/MailData/Meta/b64B, "ISO-8859-1") gibt den base64Binary-Input als ISO-8859-1-kodierten String zurück.

encode-string [altova:]

altova:encode-string(InputString *als xs:string*) **als xs:base64Binaryinteger** XP3.1 XQ3.1
altova:encode-string(InputString *als xs:string*, **Encoding** *als xs:string*) **als xs:base64Binaryinteger** XP3.1 XQ3.1
Kodiert den angegebenen String gemäß der definierten Kodierung, falls eine angegeben wird. Wenn keine Kodierung definiert ist, wird die UTF-8-Kodierung verwendet. Der kodierte String wird in base64Binary-Zeichen konvertiert und es wird der konvertierte base64Binary-Wert zurückgegeben. Anfangs wird die UTF-8-Kodierung unterstützt. Die Unterstützung wird auf die folgenden Kodierungen ausgeweitet werden: US-ASCII, ISO-8859-1, UTF-16, UTF-16LE, UTF-16BE, ISO-10646-UCS2, UTF-32, UTF-32LE, UTF-32BE, ISO-10646-UCS4

Beispiele

- **altova:encode-string**("Altova") gibt das base64Binary-Äquivalent des UTF-8-kodierten String "Altova" zurück.
- **altova:encode-string**("Altova", "UTF-8") gibt das base64Binary-Äquivalent des UTF-8-kodierten String "Altova" zurück.

get-temp-folder [altova:]

altova:get-temp-folder() als xs:string XP2 XQ1 XP3.1 XQ3.1
Diese Funktion hat kein Argument. Sie gibt den Pfad zum temporären Ordner des aktuellen Benutzers zurück.

Beispiele

- **altova:get-temp-folder**() würde auf einem Windows-Rechner z.B. den folgenden Pfad als xs:string zurückgeben: C:\Users\<UserName>\AppData\Local\Temp\.

generate-guid [altova:]

altova:generate-guid() als **xs:string** XP2 XQ1 XP3.1 XQ3.1
Generiert einen eindeutigen String GUID-String.

Beispiele

- **altova:generate-guid**() gibt (z.B.) 85F971DA-17F3-4E4E-994E-99137873ACCD zurück

high-res-timer [altova:]

© 2016-2022 Altova GmbH

altova:high-res-timer() als **xs:double** XP3.1 XQ3.1
Gibt einen hochauflösenden System-Timer-Wert in Sekunden zurück. Wenn in einem System ein hochauflösender Timer zur Verfügung steht, können bei Bedarf (z.B. bei Animationen und zur Ermittlung des exakten Codeausführungszeitpunkts) hochauflösende Zeitmessungen vorgenommen werden. Diese Funktion stellt die Auflösung des Hochauflösungs-Timers des Systems zur Verfügung.

Beispiele

- **altova:high-res-timer**() gibt eine Wert wie '1.16766146154566E6' zurück.

parse-html [altova:]

altova:parse-html(HTMLText *als xs:string***)** als **node()** XP3.1 XQ3.1
Das Argument HTMLText ist ein String, der den Text eines HTML-Dokuments enthält. Die Funktion erstellt anhand des Strings eine HTML-Struktur. Der bereitgestellte String kann das HTML-Element enthalten, muss dies aber nicht tun. In beiden Fällen ist das Root-Element der Struktur ein Element namens **HTML**. Sie sollten sicher stellen, dass der HTML-Code im bereitgestellten String gültiger HTML-Code ist.

Beispiel

- **altova:parse-html**("<html><head/><body><h1>Header</h1></body></html>") erstellt anhand des bereitgestellten Strings eine HTML-Struktur.

sleep[altova:]

altova:sleep(Millisecs *als xs:integer***)** als **empty-sequence()** XP2 XQ1 XP3.1 XQ3.1
Unterbricht die Ausführung der aktuellen Operation für die Anzahl der durch das Argument Millisecs angegebenen Millisekunden.

Beispiel

- **altova:sleep**(1000) unterbricht die Ausführung der aktuellen Operation für 1000 Millisekunden.

[Nach oben 2221]

30.2.1.10 Diagrammfunktionen

Mit Hilfe der unten aufgelisteten Diagrammfunktionen können Sie Diagramme als Bilder erstellen, generieren und speichern. Sie werden in der aktuellen Version Ihres Altova-Produkts auf die unten beschriebene Art unterstützt. Beachten Sie jedoch, dass eine oder mehrere dieser Funktionen in zukünftigen Produktversionen eventuell nicht mehr unterstützt werden, bzw. dass sich das Verhalten einzelner Funktionen ändern kann. Um Informationen über die Unterstützung für Altova Erweiterungsfunktionen in der jeweiligen Release zu erhalten, schlagen Sie bitte in der Dokumentation der jeweils aktuellen Release nach.

Anmerkung: Diagrammfunktionen werden nur in **Altova Server-Produkten** und den **Enterprise-Editionen von Altova-Produkten** unterstützt.

Anmerkung: Unterstützte Bildformate für Diagramme in Server-Editionen sind jpg, png und bmp. Die beste Option ist png, da dabei keine Daten verloren gehen und es sich um ein komprimiertes Format handelt. In Enterprise-Editionen werden die folgenden Formate unterstützt: jpg. png, bmp und gif.

Funktionen zum Generieren und Speichern von Diagrammen

Diese Funktionen generieren anhand des (mit Hilfe der Diagrammerstellungsfunktionen) erzeugten Diagrammobjekts entweder ein Bild oder speichern ein Bild in einer Datei

altova:generate-chart-image ($chart, $width, $height, $encoding) als atomic

wobei

- $chart das Diagramm-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart erzeugt wurde
- $width und $height mit einer Längeneinheit definiert werden müssen
- $encoding kann x-binarytobase64 oder x-binarytobase16 sein

Die Funktion gibt das Diagrammbild in der definierten Kodierung zurück.

altova:generate-chart-image ($chart, $width, $height, $encoding, $imagetype) als atomic

wobei

- $chart das Diagramm-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart erzeugt wurde
- $width und $height mit einer Längeneinheit definiert werden müssen
- $encoding kann x-binarytobase64 oder x-binarytobase16 sein
- $imagetype eines der folgenden Bildformate sein kann: png, gif, bmp, jpg, jpeg. Beachten Sie, dass gif in Server-Produkten nicht unterstützt wird. *Siehe oben auf dieser Seite*.

Die Funktion gibt das Diagrammbild in der definierten Kodierung und im definierten Bildformat zurück.

altova:save-chart-image ($chart, $filename, $width, $height) als empty() ***(nur Windows)***

wobei

- $chart das Diagramm-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart erzeugt wurde
- $filename der Pfad und Name der Datei ist, unter dem das Diagrammbild gespeichert werden soll
- $width und $height mit einer Längeneinheit definiert werden müssen

Die Funktion speichert das Diagrammbild unter dem in $filename definierten Dateinamen. Alternativ zu dieser Funktion könnten Sie auch die Funktion xsl:result-document mit encoding="x-base64tobinary" verwenden, wobei der image-data-Inhalt entweder über die Funktion generate-chart-image() oder die Funktion chart() erhalten wird.

altova:save-chart-image ($chart, $filename, $width, $height, $imagetype) als empty() ***(nur Windows)***

wobei

© 2016-2022 Altova GmbH

- `$chart` das Diagramm-Erweiterungsobjekt ist, das mit der Funktion `altova:create-chart` erzeugt wurde
- `$filename` der Pfad und Name der Datei ist, unter dem das Diagrammbild gespeichert werden soll
- `$width` und `$height` mit einer Längeneinheit definiert werden müssen
- `$imagetype` eines der folgenden Bildformate sein kann: `png`, `gif`, `bmp`, `jpg`, `jpeg`. Beachten Sie, dass `gif` in Server-Produkten nicht unterstützt wird. *Siehe oben auf dieser Seite*.

Die Funktion speichert das Diagrammbild im definierten Bildformat unter dem in `$filename` definierten Dateinamen. Alternativ zu dieser Funktion könnten Sie auch die Funktion `xsl:result-document` mit `encoding="x-base64tobinary"` verwenden, wobei der image-data-Inhalt entweder über die Funktion `generate-chart-image()` oder die Funktion `chart()` erhalten wird.

Funktionen zur Erstellung von Diagrammen

Die folgenden Funktionen dienen zur Erstellung von Diagrammen.

`altova:create-chart``($chart-config, $chart-data-series*) als chart extension item`

wobei

- `$chart-config` das Diagrammkonfigurations-Erweiterungsobjekt ist, das mit der Funktion `altova:create-chart-config` oder über die Funktion `altova:create-chart-config-from-xml` erzeugt wurde
- `$chart-data-series` das chart-data-series-Erweiterungsobjekt ist, das mit der Funktion `altova:create-chart-data-series` oder der Funktion `altova:create-chart-data-series-from-rows` abgerufen wird

Die Funktion gibt ein Diagrammerweiterungsobjekt zurück, das anhand der über die Argumente gelieferten Daten erzeugt wird.

`altova:chart``($chart-config, $chart-data-series*) als chart extension item`

wobei

- `$chart-config` das Diagrammkonfigurations-Erweiterungsobjekt ist. Es handelt sich hierbei um eine nicht geordnete Reihe von vier Schlüssel:Wert-Paaren, wobei die vier Schlüssel `"width"`, `"height"`, `"title"` und `"kind"` sind. Die Werte von `width` und `height` sind Ganzzahlen und definieren die Höhe und Breite des Diagramms in Pixel. Der Wert von `kind` ist einer der folgenden: `Pie, Pie3d, BarChart, BarChart3d, BarChart3dGrouped, LineChart, ValueLineChart, RoundGauge, BarGauge`.
- `$chart-data-series` ist jeweils ein Array der Größe 3, wobei jedes Array eine Diagrammdatenreihe definiert. Jedes Array besteht aus: (i) dem Namen der Datenreihe, (ii) den Werten der x-Achse, (iii) den Werten der y-Achse. Es können mehrere Datenreihen angegeben werden. Im Beispiel unten enthalten die zwei Arrays Daten für die Temperatur-Monatsminima bzw. -maxima.

Die Funktion gibt ein Element vom Typ `xs:base64Binary` zurück, welches das Diagrammbild enthält. Dieses Bild wird anhand der über die Argumente der Funktion bereitgestellten Daten erstellt. Beachten Sie, dass diese Funktion, da sie Arrays und Zuordnungen verwendet, nur in XPath 3.1, XQuery 3.1 oder XSLT 3.0 verwendet werden kann.

Beispiel: `altova:chart( map{'width':800, 'height':600, "kind":"LineChart", "title":"Monthly Temperatures"}, (['Min', $temps/Month, $temps/Month/@min], ['Max', $temps/Month, $temps/Month/@max]) )`

altova:create-chart-config`($type-name, $title) als chart-config Erweiterungsobjekt`

wobei

- `$type-name` den Typ des zu erstellenden Diagramms definiert: `Pie, Pie3d, BarChart, BarChart3d, BarChart3dGrouped, LineChart, ValueLineChart, RoundGauge, BarGauge`
- `$title` der Name des Diagramms ist

Die Funktion gibt ein Diagrammkonfigurations-Erweiterungsobjekt zurück, das die Konfigurationsinformationen zum Diagramm enthält.

altova:create-chart-config-from-xml`($xml-struct) als chart-config Erweiterungsobjekt`

wobei

- `$xml-struct` die XML-Struktur ist, die die Konfigurationsinformationen des Diagramms enthält

Die Funktion gibt ein Diagrammkonfigurations-Erweiterungsobjekt zurück, das die Konfigurationsinformationen zum Diagramm enthält. Diese Informationen werden in einem XML-Datenfragment 2228 geliefert.

altova:create-chart-data-series`($series-name?, $x-values*, $y-values*) als chart-data-series Erweiterungsobjekt`

wobei

- `$series-name` der Name der Datenreihe ist
- `$x-values` die Liste der Werte für die X-Achse liefert
- `$y-values` die Liste der Werte für die Y-Achse liefert

Die Funktion gibt ein Diagrammdatenreihen-Erweiterungsobjekt zurück, das die Daten zur Erstellung des Diagramms, also die Namen der Datenreihen, und die Achsendaten enthält.

altova:create-chart-data-row`(x, y1, y2, y3, ...) als chart-data-x-`*N*`y-row Erweiterungsobjekt`

wobei

- `x` der Wert der X-Achsen-Spalte der Diagrammdatenzeile ist
- `yN` die Werte der Spalten für die Y-Achse sind

Die Funktion gibt ein chart-data-x-Ny-row Erweiterungsobjekt zurück, das die Daten für die X-Achsen-Spalte und die Y-Achsen-Spalten einer einzigen Datenreihe enthält.

© 2016-2022 Altova GmbH

altova:create-chart-data-series-from-rows($series-names as xs:string*, $row*) als chart-data-series-Erweiterungsobjekt

wobei

- $series-name der Name der zu erstellenden Datenreihen ist
- $row das chart-data-x-Ny-row Erweiterungsobjekt ist, das als Datenreihe erstellt werden soll

Die Funktion gibt ein chart-data-series Erweiterungsobjekt zurück, das die Daten für die X- und die Y-Achse der Datenreihe enthält.

altova:create-chart-layer($chart-config, $chart-data-series*) als chart-layer Erweiterungsobjekt

wobei

- $chart-config das Diagrammkonfigurations-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart-config oder über die Funktion altova:create-chart-config-from-xml erzeugt wurde
- $chart-data-series das chart-data-series-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart-data-series oder der Funktion altova:create-chart-data-series-from-rows abgerufen wird

Die Funktion gibt ein chart-layer Erweiterungsobjekt zurück, das chart-layer-Daten enthält.

altova:create-multi-layer-chart($chart-config, $chart-data-series*, $chart-layer*)

wobei

- $chart-config das Diagrammkonfigurations-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart-config oder über die Funktion altova:create-chart-config-from-xml erzeugt wurde
- $chart-data-series das chart-data-series-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart-data-series oder der Funktion altova:create-chart-data-series-from-rows abgerufen wird
- $chart-layer das chart-layer-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart-layer abgerufen wird

Die Funktion gibt ein multi-layer-chart-Objekt zurück.

altova:create-multi-layer-chart($chart-config, $chart-data-series*, $chart-layer*, xs:boolean $mergecategoryvalues)

wobei

- $chart-config das Diagrammkonfigurations-Erweiterungsobjekt ist, das mit der Funktion altova:create-chart-config oder über die Funktion altova:create-chart-config-from-xml erzeugt wurde

- `$chart-data-series` das chart-data-series-Erweiterungsobjekt ist, das mit der Funktion `altova:create-chart-data-series` oder der Funktion `altova:create-chart-data-series-from-rows` abgerufen wird
- `$chart-layer` das chart-layer-Erweiterungsobjekt ist, das mit der Funktion `altova:create-chart-layer` abgerufen wird
- `$mergecategoryvalues` bei `true` die Werte mehrerer Datenreihen zusammenführt und dies bei `false` nicht tut.

Die Funktion gibt ein multi-layer-chart-Objekt zurück.

30.2.1.10.1 XML-Struktur von Diagrammdaten

Unten sehen Sie die XML-Struktur von Diagrammdaten, wie sie für Altova-Erweiterungsfunktionen für Diagramme 2223 angezeigt werden könnte. Diese Funktionen beeinflussen das Aussehen der einzelnen Diagramme. Nicht alle Elemente werden für alle Diagrammarten verwendet, so wird z.B. das Element <Pie> bei Balkendiagrammen ignoriert.

Anmerkung: Diagrammfunktionen werden nur in **Enterprise und Server-Editionen** von Altova Produkten unterstützt.

```
<chart-config>
    <General
    SettingsVersion="1"  must be provided
    ChartKind="BarChart"  Pie, Pie3d, BarChart, StackedBarChart, BarChart3d, BarChart3dGrouped,
LineChart, ValueLineChart, AreaChart, StackedAreaChart, RoundGauge, BarGauge, CandleStick
    BKColor="#ffffff"  Color
    BKColorGradientEnd="#ffffff"  Color. In case of a gradient, BKColor and BKColorGradientEnd
define the gradient's colors
    BKMode="#ffffff"  Solid, HorzGradient, VertGradient
    BKFile="Path+Filename"  String. If file exists, its content is drawn over the background.
    BKFileMode="Stretch"  Stretch, ZoomToFit, Center, Tile
    ShowBorder="1"  Bool
    PlotBorderColor="#000000"  Color
    PlotBKColor="#ffffff"  Color
    Title=""  String
    ShowLegend="1"  Bool
    OutsideMargin="3.%"  PercentOrPixel
    TitleToPlotMargin="3.%"  PercentOrPixel
    LegendToPlotMargin="3.%"  PercentOrPixel
    Orientation="vert"  Enumeration: possible values are: vert, horz
    >
    <TitleFont
       Color="#000000"  Color
       Name="Tahoma"  String
       Bold="1"  Bool
       Italic="0"  Bool
       Underline="0"  Bool
       MinFontHeight="10.pt"  FontSize (only pt values)
       Size="8.%"  FontSize />
```

```
<LegendFont
   Color="#000000"
   Name="Tahoma"
   Bold="0"
   Italic="0"
   Underline="0"
   MinFontHeight="10.pt"
   Size="3.5%" />
<AxisLabelFont
   Color="#000000"
   Name="Tahoma"
   Bold="1"
   Italic="0"
   Underline="0"
   MinFontHeight="10.pt"
   Size="5.%" />
</General>

<Line
      ConnectionShapeSize="1.%"  PercentOrPixel
      DrawFilledConnectionShapes="1"  Bool
      DrawOutlineConnectionShapes="0"  Bool
      DrawSlashConnectionShapes="0"  Bool
      DrawBackslashConnectionShapes="0"  Bool
/>

<Bar
      ShowShadow="1"  Bool
      ShadowColor="#a0a0a0"  Color
      OutlineColor="#000000"  Color
      ShowOutline="1"  Bool
/>

<Area
      Transparency="0"  UINT ( 0-255 ) 255 is fully transparent, 0 is opaque
      OutlineColor="#000000"  Color
      ShowOutline="1"  Bool
/>

<CandleStick
      FillHighClose="0"  Bool. If 0, the body is left empty. If 1, FillColorHighClose is used for
the candle body
      FillColorHighClose="#ffffff"  Color. For the candle body when close > open
      FillHighOpenWithSeriesColor="1"  Bool. If true, the series color is used to fill the
candlebody when open > close
      FillColorHighOpen="#000000"  Color. For the candle body when open > close and
FillHighOpenWithSeriesColor is false
/>

<Colors  User-defined color scheme: By default this element is empty except for the style and has no
Color attributes
      UseSubsequentColors ="1"  Boolean. If 0, then color in overlay is used. If 1, then
subsequent colors from previous chart layer is used
      Style="User"  Possible values are: "Default", "Grayscale", "Colorful", "Pastel", "User"
```

```
            Colors="#52aca0"   Color: only added for user defined color set
            Colors1="#d3c15d"   Color: only added for user defined color set
            Colors2="#8971d8"   Color: only added for user defined color set
            ...
            ColorsN=""   Up to ten colors are allowed in a set: from Colors to Colors9
      </Colors>

      <Pie
            ShowLabels="1"   Bool
            OutlineColor="#404040"   Color
            ShowOutline="1"   Bool
            StartAngle="0."   Double
            Clockwise="1"   Bool
            Draw2dHighlights="1"   Bool
            Transparency="0"   Int (0 to 255: 0 is opaque, 255 is fully transparent)
            DropShadowColor="#c0c0c0"   Color
            DropShadowSize="5.%"   PercentOrPixel
            PieHeight="10.%"   PercentOrPixel. Pixel values might be different in the result because of
3d tilting
            Tilt="40.0"   Double (10 to 90: The 3d tilt in degrees of a 3d pie)
            ShowDropShadow="1"   Bool
            ChartToLabelMargin="10.%"   PercentOrPixel
            AddValueToLabel="0"   Bool
            AddPercentToLabel="0"   Bool
            AddPercentToLabels_DecimalDigits="0"   UINT ( 0 – 2 )
            >
            <LabelFont
                  Color="#000000"
                  Name="Arial"
                  Bold="0"
                  Italic="0"
                  Underline="0"
                  MinFontHeight="10.pt"
                  Size="4.%"    />
      </Pie>

      <XY>
            <XAxis   Axis
                  AutoRange="1"   Bool
                  AutoRangeIncludesZero="1"   Bool
                  RangeFrom="0."   Double: manual range
                  RangeTill="1."   Double : manual range
                  LabelToAxisMargin="3.%"   PercentOrPixel
                  AxisLabel=""   String
                  AxisColor="#000000"   Color
                  AxisGridColor="#e6e6e6"   Color
                  ShowGrid="1"   Bool
                  UseAutoTick="1"   Bool
                  ManualTickInterval="1."   Double
                  AxisToChartMargin="0.px"   PercentOrPixel
                  TickSize="3.px"   PercentOrPixel
                  ShowTicks="1"   Bool
                  ShowValues="1"   Bool
```

 © 2016-2022 Altova GmbH

```
                AxisPosition="LeftOrBottom"  Enums: "LeftOrBottom", "RightOrTop", "AtValue"
                AxisPositionAtValue = "0"  Double
                >
                <ValueFont
                        Color="#000000"
                        Name="Tahoma"
                        Bold="0"
                        Italic="0"
                        Underline="0"
                        MinFontHeight="10.pt"
                        Size="3.%"    />
        </XAxis>
        <YAxis  Axis (same as for XAxis)
                AutoRange="1"
                AutoRangeIncludesZero="1"
                RangeFrom="0."
                RangeTill="1."
                LabelToAxisMargin="3.%"
                AxisLabel=""
                AxisColor="#000000"
                AxisGridColor="#e6e6e6"
                ShowGrid="1"
                UseAutoTick="1"
                ManualTickInterval="1."
                AxisToChartMargin="0.px"
                TickSize="3.px"
                ShowTicks="1"  Bool
                ShowValues="1"  Bool
                AxisPosition="LeftOrBottom"  Enums: "LeftOrBottom", "RightOrTop", "AtValue"
                AxisPositionAtValue = "0"  Double
                >
                <ValueFont
                        Color="#000000"
                        Name="Tahoma"
                        Bold="0"
                        Italic="0"
                        Underline="0"
                        MinFontHeight="10.pt"
                        Size="3.%"/>
        </YAxis>
</XY>

<XY3d
        AxisAutoSize="1"  Bool: If false, XSize and YSize define the aspect ration of x and y axis.
If true, aspect ratio is equal to chart window
        XSize="100.%"  PercentOrPixel. Pixel values might be different in the result because of 3d
tilting and zooming to fit chart
        YSize="100.%"  PercentOrPixel. Pixel values might be different in the result because of 3d
tilting and zooming to fit chart
        SeriesMargin="30.%"  PercentOrPixel. Pixel values might be different in the result because
of 3d tilting and zooming to fit chart
        Tilt="20."  Double. -90 to +90 degrees
        Rot="20."  Double. -359 to +359 degrees
        FoV="50.">  Double. Field of view: 1-120 degree
        >
```

```
        <ZAxis
               AutoRange="1"
               AutoRangeIncludesZero="1"
               RangeFrom="0."
               RangeTill="1."
               LabelToAxisMargin="3.%"
               AxisLabel=""
               AxisColor="#000000"
               AxisGridColor="#e6e6e6"
               ShowGrid="1"
               UseAutoTick="1"
               ManualTickInterval="1."
               AxisToChartMargin="0.px"
               TickSize="3.px" >
               <ValueFont
                      Color="#000000"
                      Name="Tahoma"
                      Bold="0"
                      Italic="0"
                      Underline="0"
                      MinFontHeight="10.pt"
                      Size="3.%"/>
        </ZAxis>
</XY3d>

<Gauge
        MinVal="0."  Double
        MaxVal="100."  Double
        MinAngle="225"  UINT: -359-359
        SweepAngle="270"  UINT: 1-359
        BorderToTick="1.%"  PercentOrPixel
        MajorTickWidth="3.px"  PercentOrPixel
        MajorTickLength="4.%"  PercentOrPixel
        MinorTickWidth="1.px"  PercentOrPixel
        MinorTickLength="3.%"  PercentOrPixel
        BorderColor="#a0a0a0"  Color
        FillColor="#303535"  Color
        MajorTickColor="#a0c0b0"  Color
        MinorTickColor="#a0c0b0"  Color
        BorderWidth="2.%"  PercentOrPixel
        NeedleBaseWidth="1.5%"  PercentOrPixel
        NeedleBaseRadius="5.%"  PercentOrPixel
        NeedleColor="#f00000"  Color
        NeedleBaseColor="#141414"  Color
        TickToTickValueMargin="5.%"  PercentOrPixel
        MajorTickStep="10."  Double
        MinorTickStep="5."  Double
        RoundGaugeBorderToColorRange="0.%"  PercentOrPixel
        RoundGaugeColorRangeWidth ="6.%"  PercentOrPixel
        BarGaugeRadius="5.%"  PercentOrPixel
        BarGaugeMaxHeight="20.%"  PercentOrPixel
        RoundGaugeNeedleLength="45.%"  PercentOrPixel
        BarGaugeNeedleLength="3.%"  PercentOrPixel
        >
```


© 2016-2022 Altova GmbH

```
            <TicksFont
                  Color="#a0c0b0"
                  Name="Tahoma"
                  Bold="0"
                  Italic="0"
                  Underline="0"
                  MinFontHeight="10.pt"
                  Size="4.%"
            />
            <ColorRanges>   User-defined color ranges. By default empty with no child element entries
                  <Entry
                        From="50. "   Double
                        FillWithColor="1"   Bool
                        Color="#00ff00"   Color
                  />
                  <Entry
                        From="50.0"
                        FillWithColor="1"
                        Color="#ff0000"
                  />
                  ...
            </ColorRanges>
      </Gauge>
</chart-config>
```

30.2.1.10.2 Beispiel: Diagrammfunktionen

Anhand des XSLT-Beispieldokuments weiter unten sehen Sie, wie Altova-Erweiterungsfunktionen für Diagramme[2223] eingesetzt werden können. Weiter unten sehen Sie ein XML-Dokument und eine Abbildung des Ausgabebilds, das generiert wird, wenn ein XML-Dokument mit dem XSLT 2.0- oder XSLT 3.0-Prozessor anhand des XSLT-Dokuments verarbeitet wird.

Anmerkung: Diagrammfunktionen werden nur in **Enterprise und Server-Editionen** von Altova Produkten unterstützt.

Anmerkung: Weitere Informationen zur Erstellung von Diagrammdatentabellen finden Sie in der Dokumentation zu den Altova-Produkten XMLSpy und StyleVision.

XSLT-Dokument

In diesem (*unten aufgelisteten*) XSLT-Dokument wird mit Hilfe der Altova Diagramm-Erweiterungsfunktionen ein Kreisdiagramm generiert. Das XSLT-Dokument kann zur Verarbeitung des weiter unten angeführten XML-Dokuments verwendet werden.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:altovaext="http://www.altova.com/xslt-extensions"
   exclude-result-prefixes="#all">
   <xsl:output version="4.0" method="html" indent="yes" encoding="UTF-8"/>
```

```
   <xsl:template match="/">
      <html>
         <head>
            <title>
               <xsl:text>HTML Page with Embedded Chart</xsl:text>
            </title>
         </head>
         <body>
            <xsl:for-each select="/Data/Region[1]">
               <xsl:variable name="extChartConfig" as="item()*">
                  <xsl:variable name="ext-chart-settings" as="item()*">
                     <chart-config>
                        <General
                           SettingsVersion="1"
                           ChartKind="Pie3d"
                           BKColor="#ffffff"
                           ShowBorder="1"
                           PlotBorderColor="#000000"
                           PlotBKColor="#ffffff"
                           Title="{@id}"
                           ShowLegend="1"
                           OutsideMargin="3.2%"
                           TitleToPlotMargin="3.%"
                           LegendToPlotMargin="6.%"
                           >
                           <TitleFont
                              Color="#023d7d"
                              Name="Tahoma"
                              Bold="1"
                              Italic="0"
                              Underline="0"
                              MinFontHeight="10.pt"
                              Size="8.%" />
                        </General>
                     </chart-config>
                  </xsl:variable>
                  <xsl:sequence select="altovaext:create-chart-config-from-xml( $ext-
chart-settings )"/>
               </xsl:variable>
               <xsl:variable name="chartDataSeries" as="item()*">
                  <xsl:variable name="chartDataRows" as="item()*">
                     <xsl:for-each select="(Year)">
                        <xsl:sequence select="altovaext:create-chart-data-row( (@id),
( .) )"/>
                     </xsl:for-each>
                  </xsl:variable>
                  <xsl:variable name="chartDataSeriesNames" as="xs:string*"
select=" ( (&quot;Series 1&quot;), &apos;&apos; )[1]"/>
                  <xsl:sequence
                     select="altovaext:create-chart-data-series-from-
rows( $chartDataSeriesNames, $chartDataRows)"/>
               </xsl:variable>
```


© 2016-2022 Altova GmbH

```
                <xsl:variable name="ChartObj" select="altovaext:create-
chart( $extChartConfig, ( $chartDataSeries), false() )"/>
                <xsl:variable name="sChartFileName" select="'mychart1.png'"/>
                <img src="{$sChartFileName, altovaext:save-chart-image( $ChartObj,
$sChartFileName, 400, 400 ) }"/>
             </xsl:for-each>
          </body>
       </html>
   </xsl:template>
</xsl:stylesheet>
```

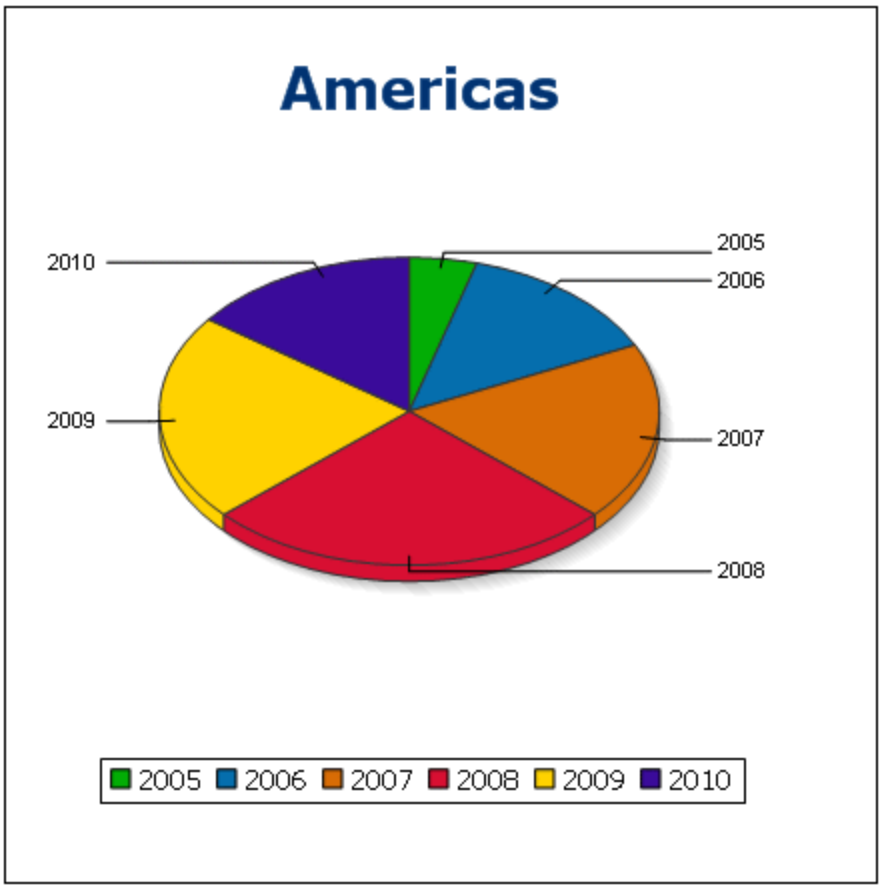
XML-Dokument

Dieses XML-Dokument kann mit dem oben stehenden XSLT-Dokument verarbeitet werden. Anhand der Daten im XML-Dokument wird das in der unten stehenden Abbildung gezeigte Kreisdiagramm generiert.

```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
   <ChartType>Pie Chart 2D</ChartType>
   <Region id="Americas">
      <Year id="2005">30000</Year>
      <Year id="2006">90000</Year>
      <Year id="2007">120000</Year>
      <Year id="2008">180000</Year>
      <Year id="2009">140000</Year>
      <Year id="2010">100000</Year>
   </Region>
   <Region id="Europe">
      <Year id="2005">50000</Year>
      <Year id="2006">60000</Year>
      <Year id="2007">80000</Year>
      <Year id="2008">100000</Year>
      <Year id="2009">95000</Year>
      <Year id="2010">80000</Year>
   </Region>
   <Region id="Asia">
      <Year id="2005">10000</Year>
      <Year id="2006">25000</Year>
      <Year id="2007">70000</Year>
      <Year id="2008">110000</Year>
      <Year id="2009">125000</Year>
      <Year id="2010">150000</Year>
   </Region>
</Data>
```

Ausgabebild

Das unten gezeigt Kreisdiagramm wird generiert, wenn das oben aufgelistete XML-Dokument mit Hilfe des XSLT-Dokuments verarbeitet wird.



30.2.2 Diverse Erweiterungsfunktionen

Es gibt in Programmiersprachen wie Java und C# eine Reihe von fertigen Funktionen, die nicht als XQuery / XPath 2.0- oder XSLT-Funktionen zur Verfügung stehen. Ein gutes Beispiel dafür sind die mathematischen in Java verfügbaren Funktionen wie z.B. `sin()` und `cos()`. Stünden diese Funktionen für die Erstellung von XSLT Stylesheets und XQuery-Abfragen zur Verfügung, würde sich der Einsatzbereich von Stylesheets und Abfragen erweitern und die Erstellung von Stylesheets wäre viel einfacher. Der in einer Reihe von Altova-Produkten verwendete XSLT- und XQuery-Prozessor von Altova unterstützt die Verwendung von Erweiterungsfunktionen in Java[2237] und .NET[2246] sowie MSXSL Skripts für XSLT[2253]. MSXSL-Skripts für XSLT[2253] und die Altova-Erweiterungsfunktionen[2144]. In diesem Abschnitt wird beschrieben, wie Sie Erweiterungsfunktionen und MSXSL-Skripts in Ihren XSLT Stylesheets und XQuery-Dokumenten verwenden können. Diese Beschreibungen finden Sie in den folgenden Abschnitten:

- Java-Erweiterungsfunktionen[2237]
- .NET-Erweiterungsfunktionen[2246]
- MSXSL-Skripts für XSLT[2253]

Hauptsächlich werden dabei die folgenden beiden Punkte behandelt: (i) Wie Funktionen in den entsprechenden Bibliotheken aufgerufen werden; und (ii) welche Regeln beim Konvertieren von Argumenten in einem

© 2016-2022 Altova GmbH

Funktionsaufruf in das erforderliche Format der Funktion befolgt werden und welche Regeln bei der Rückwärtskonvertierung (Funktionsresultat in XSLT/XQuery Datenobjekt) befolgt werden.

Voraussetzungen

Damit die Erweiterungsfunktionen unterstützt werden, muss auf dem Rechner, auf dem die XSLT-Transformation oder die XQuery-Ausführung stattfindet, eine Java Runtime-Umgebung (zum Aufrufen der Java-Funktionen) und ein .NET Framework 2.0 (Mindestvoraussetzung für Zugriff auf .NET-Funktionen) installiert sein oder es muss Zugriff auf eine solche bestehen.

30.2.2.1 Java-Erweiterungsfunktionen

Eine Java-Erweiterungsfunktion kann in einem XPath- oder XQuery-Ausdruck verwendet werden, um einen Java-Konstruktor oder eine Java-Methode (statisch oder Instanz) aufzurufen.

Ein Feld in einer Java-Klasse wird als Methode ohne Argument betrachtet. Bei einem Feld kann es sich um ein statisches Feld oder eine Instanz handeln. Wie man Felder aufruft, wird in den entsprechenden Unterabschnitten zu statischen Feldern und Instanzen beschrieben.

Dieser Abschnitt enthält die folgenden Unterabschnitte:

- Java: Konstruktoren 2243
- Java: Statische Methoden und statische Felder 2243
- Java: Instanzmethoden und Instanzfelder 2244
- Datentypen: XPath/XQuery in Java 2245
- Datentypen: Java in XPath/XQuery 2246

Beachten Sie die folgenden Punkte

- Wenn Sie ein Altova Desktop-Produkt verwenden, versucht die Altova-Applikation, den Pfad zur Java Virtual Machine automatisch zu ermitteln. Dazu wird zuerst (i) die Windows Registry und dann (ii) die **JAVA_HOME**-Umgebungsvariable gelesen. Sie können im Dialogfeld "Optionen" der Applikation auch einen benutzerdefinierten Pfad hinzufügen. Dieser Eintrag hat Vorrang vor allen anderen automatisch ermittelten Java VM-Pfaden.
- Wenn Sie ein Altova Server-Produkt auf einem Windows-Rechner ausführen, wird der Pfad zur Java Virtual Machine zuerst aus der Windows Registry ausgelesen. Falls dies nicht gelingt, wird die **JAVA_HOME**-Umgebungsvariable verwendet.
- Wenn Sie ein Altova Server-Produkt auf einem Linux- oder macOS-Rechner ausführen, sollten Sie sicherstellen, dass die **JAVA_HOME**-Umgebungsvariable ordnungsgemäß definiert ist und dass sich die Java Virtual Machine-Bibliothek (unter Windows die Datei **jvm.dll**) entweder im Verzeichnis **\bin\server** oder **\bin\client** befindet.

Form der Erweiterungsfunktion

Die Erweiterungsfunktion im XPath/XQuery-Ausdruck muss die folgenden Form haben `präfix:fname()`.

- Der Teil `präfix:` kennzeichnet die Erweiterungsfunktion als Java-Funktion, indem er die Erweiterungsfunktion mit einer in-scope Namespace-Deklaration verknüpft, deren URI mit `java:` beginnen muss (*Beispiele siehe unten*). Die Namespace-Deklaration sollte eine Java-Klasse bezeichnen, z.B:

`xmlns:myns="java:java.lang.Math"`. Sie könnte aber auch einfach lauten: `xmlns:myns="java"` (ohne Doppelpunkt), wobei die Identifizierung der Java-Klasse dem `fname()` Teil der Erweiterungsfunktion überlassen bleibt.

- Der Teil `fname()` identifiziert die aufgerufene Java-Methode und liefert die Argumente für die Methode (*Beispiele siehe unten*). Wenn die durch das `prefix:` Teil identifizierte Namespace URI jedoch keine Java-Klasse bezeichnet (*siehe vorheriger Punkt),* dann sollte die Java-Klasse im `fname()` Teil vor der Klasse identifiziert werden und von der Klasse durch einen Punkt getrennt sein (*siehe zweites XSLT-Beispiel unten*).

Anmerkung: Die aufgerufene Klasse muss sich unter dem Klassenpfad des Rechners befinden.

XSLT-Beispiel

Hier sehen Sie zwei Beispiele dafür, wie eine statische Methode aufgerufen werden kann. Im ersten Beispiel ist der Klassenname (`java.lang.Math`) in der Namespace URI enthalten und darf daher nicht im `fname()` Teil enthalten sein. Im zweiten Beispiel liefert der `prefix:` Teil das Präfix `java:`, während der `fname()` Teil die Klasse sowie die Methode identifiziert.

```
<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos(3.14)" />

<xsl:value-of xmlns:jmath="java"
              select="jmath:java.lang.Math.cos(3.14)" />
```

Die in der Erweiterungsfunktion (im Beispiel oben `cos()`) angegebene Methode muss mit dem Namen einer öffentlichen statischen Methode in der angegebenen Java-Klasse (im Beispiel oben `java.lang.Math`) übereinstimmen.

XQuery-Beispiel

Hier sehen Sie ein XQuery-Beispiel, das dem XSLT-Beispiel oben ähnlich ist:

```
<cosine xmlns:jMath="java:java.lang.Math">
   {jMath:cos(3.14)}
</cosine>
```

Benutzerdefinierte Java-Klassen

Wenn Sie Ihre eigenen Java-Klassen erstellt haben, werden die Methoden in diesen Klassen unterschiedlich aufgerufen, je nachdem: (i) ob die Klassen über eine JAR-Datei oder eine Klassendatei aufgerufen werden, und (ii) ob sich diese Dateien (JAR oder Klasse) im aktuellen Verzeichnis befinden (im selben Verzeichnis wie das XSLT- oder XQuery-Dokument) oder nicht. Wie Sie diese Dateien finden, wird in den Abschnitten Benutzerdefinierte Klassendateien 2239 und Benutzerdefinierte Jar-Dateien 2242 beschrieben. Pfade zu Klassendateien, die sich nicht im aktuellen Verzeichnis befinden, und Pfade zu allen JAR-Dateien müssen jedoch angegeben werden.

30.2.2.1.1 Benutzerdefinierte Klassendateien

Wenn der Zugriff über eine Klassendatei erfolgt, gibt es vier Möglichkeiten:

- Die Klassendatei befindet sich in einem Paket. Die XSLT-oder XQuery-Datei befindet sich im selben Ordner wie das Java-Paket. (*Siehe Beispiel unten* 2239.)
- Die Klassendatei befindet sich nicht in einem Paket. Die XSLT-oder XQuery-Datei befindet sich im selben Ordner wie die Klassendatei. (*Siehe Beispiel unten* 2240.)
- Die Klassendatei befindet sich in einem Paket. Die XSLT-oder XQuery-Datei befindet sich in irgendeinem beliebig gewählten Ordner. (*Siehe Beispiel unten* 2240.)
- Die Klassendatei befindet sich nicht in einem Paket. Die XSLT-oder XQuery-Datei befindet sich in irgendeinem beliebig gewählten Ordner. (*Siehe Beispiel unten* 2241.)

Gesetzt der Fall, die Klassendatei befindet sich nicht in einem Paket, sondern im selben Ordner wie das XSLT- oder XQuery-Dokument, so muss der Dateipfad nicht angegeben werden, da alle Klassen im Ordner gefunden werden. Die Syntax zum Identifizieren einer Klasse lautet:

```
java:classname
```

wobei

`java`: angibt, dass eine benutzerdefinierte Java-Funktion aufgerufen wird; (Java-Klassen im aktuellen Verzeichnis werden standardmäßig geladen)
`classname` der Name der Klasse der erforderlichen Methode ist

die Klasse in einer Namespace URI identifiziert wird und der Namespace einem Methodenaufruf als Präfix vorangestellt wird.

Klassendatei in einem Paket, XSLT/XQuery-Datei befindet sich im selben Ordner wie das Java-Paket

Im Beispiel unten wird die Methode `getVehicleType()` der Klasse `Car` des Pakets `com.altova.extfunc` aufgerufen. Das Paket `com.altova.extfunc` befindet sich im Ordner `JavaProject`. Die XSLT-Datei befindet sich ebenfalls im Ordner `JavaProject`.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:com.altova.extfunc.Car" >
<xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

<xsl:template match="/">
   <a>
   <xsl:value-of select="car:getVehicleType()"/>
   </a>
</xsl:template>

</xsl:stylesheet>
```

Die Klassendatei wird referenziert, die XSLT/XQuery-Datei befindet sich im selben Ordner wie die Klassendatei

Im Beispiel unten wird die Methode `getVehicleType()` der Klasse `Car` des Pakets `com.altova.extfunc` aufgerufen. Angenommen, (i) die Klassendatei `Car` class befindet sich im folgenden Ordner: `JavaProject/com/altova/extfunc` und (ii) dieser Ordner ist der aktuelle Ordner im Beispiel unten. Die XSLT-Datei befindet sich ebenfalls im Ordner `JavaProject/com/altova/extfunc`.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:Car" >
<xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

<xsl:template match="/">
   <a>
   <xsl:value-of select="car:getVehicleType()"/>
   </a>
</xsl:template>

</xsl:stylesheet>
```

Die Klassendatei befindet sich in einem Paket, die XSLT/XQuery-Datei befindet sich in einem beliebigen Ordner

Im Beispiel unten wird die Methode `getCarColor()` der Klasse `Car` des Pakets `com.altova.extfunc` aufgerufen. Das Paket `com.altova.extfunc` befindet sich im Ordner `JavaProject`. Die XSLT-Datei befindet sich in einem beliebigen Ordner. In diesem Fall muss der Pfad des Pakets mit der URI als Abfragestring definiert werden. Die Syntax lautet:

```
java:classname[?path=uri-of-classfile]
```

wobei

`java`: angibt, dass eine benutzerdefinierte Java-Funktion aufgerufen wird
`uri-of-classfile` die URI der Klassendatei ist
`classname` der Name der Klasse der benötigten Methode ist

die Klasse in einer Namespace URI identifiziert wird und der Namespace einem Methodenaufruf als Präfix vorangestellt wird. Im Beispiel unten sehen Sie, wie eine Klassendatei aufgerufen wird, die sich in einem anderen als dem aktuellen Verzeichnis befindet.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:com.altova.extfunc.Car?path=file:///C:/JavaProject/" >
```


© 2016-2022 Altova GmbH

```
<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
   <xsl:variable name="myCar" select="car:new('red')" />
   <a><xsl:value-of select="car:getCarColor($myCar)"/></a>
</xsl:template>

</xsl:stylesheet>
```

Die Klassendatei wird referenziert, die XSLT/XQuery-Datei befindet sich in einem beliebigen Ordner

Im Beispiel unten wird die Methode `getCarColor()` der Klasse `Car` aufgerufen. Angenommen, die Klassendatei `Car` befindet sich im Ordner `C:/JavaProject/com/altova/extfunc`. Die XSLT-Datei befindet sich in einem beliebigen Ordner. Der Pfad der Klassendatei muss dann in der Namespace-URI als Abfragestring definiert werden. Die Syntax lautet:

```
java:classname[?path=<uri-of-classfile>]
```

wobei

`java:` angibt, dass eine benutzerdefinierte Java-Funktion aufgerufen wird
`uri-of-classfile` die URI der Klassendatei ist
`classname` der Name der Klasse der benötigten Methode ist

die Klasse in einer Namespace URI identifiziert wird und der Namespace einem Methodenaufruf als Präfix vorangestellt wird. Im Beispiel unten sehen Sie, wie eine Klassendatei aufgerufen wird, die sich in einem anderen als dem aktuellen Verzeichnis befindet.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:Car?path=file:///C:/JavaProject/com/altova/extfunc/" >

<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
   <xsl:variable name="myCar" select="car:new('red')" />
   <a><xsl:value-of select="car:getCarColor($myCar)"/></a>
</xsl:template>

</xsl:stylesheet>
```

Anmerkung: Wenn ein Pfad über eine Erweiterungsfunktion angegeben wird, wird er zum ClassLoader hinzugefügt.

30.2.2.1.2 Benutzerdefinierte Jar-Dateien

JAR-Dateien
Wenn der Zugriff über eine JAR-Datei erfolgt, muss die URI der JAR-Datei mit Hilfe der folgenden Syntax definiert werden:

```
xmlns:classNS="java:classname?path=jar:uri-of-jarfile!/"
```

Die Methode wird anschließend durch Verwendung des Präfix der Namespace URI aufgerufen, der die Klasse bezeichnet: `classNS:method()`

wobei im obigen Beispiel:

`java:` angibt, dass eine Java-Funktion aufgerufen wird
`classname` der Name der Klasse der benutzerdefinierten Klasse ist
`?` das Trennzeichen zwischen dem Klassennamen und dem Pfad ist
`path=jar:` angibt, dass es sich um einen Pfad zu einer JAR-Datei handelt
`uri-of-jarfile` die URI der jar-Datei angibt
`!/` das Trennzeichen am Ende des Pfades ist
`classNS:method()` der Aufruf der Methode ist

Alternativ dazu kann der Klassenname mit dem Methodenaufruf angegeben werden. Hier sehen Sie zwei Beispiele für die Syntax:

```
xmlns:ns1="java:docx.layout.pages?
path=jar:file:///c:/projects/docs/docx.jar!/"
ns1:main()

xmlns:ns2="java?path=jar:file:///c:/projects/docs/docx.jar!/"
ns2:docx.layout.pages.main()
```

Hier sehen Sie ein komplettes XSLT-Beispiel, in dem eine JAR-Datei verwendet wird, um eine Java-Erweiterungsfunktion aufzurufen.:

```
<xsl:stylesheet version="2.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:fn="http://www.w3.org/2005/xpath-functions"
    xmlns:car="java?path=jar:file:///C:/test/Car1.jar!/" >
<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
    <xsl:variable name="myCar" select="car:Car1.new('red')" />
    <a><xsl:value-of select="car:Car1.getCarColor($myCar)"/></a>
</xsl:template>

<xsl:template match="car"/>

</xsl:stylesheet>
```

Anmerkung: Wenn ein Pfad über eine Erweiterungsfunktion angegeben wird, wird er zum ClassLoader hinzugefügt.

© 2016-2022 Altova GmbH

30.2.2.1.3 Java: Konstruktoren

Eine Erweiterungsfunktion kann dazu verwendet werden, um einen Java-Konstruktor aufzurufen. Alle Konstruktoren werden mit der Pseudofunktion `new()` aufgerufen.

Wenn das Ergebnis eines Java-Konstruktors implizit in XPath/XQuery-Datentypen konvertiert werden kann[2246], dann gibt die Java-Erweiterungsfunktion eine Sequenz zurück, bei der es sich um einem XPath/XQuery-Datentyp handelt. Wenn das Ergebnis eines Konstruktoraufrufs nicht in einen passenden XPath/XQuery-Datentyp konvertiert werden kann, dann erstellt der Konstruktor ein wrapped Java-Objekt mit einem Typ, der den Namen der Klasse hat, die dieses Java-Objekt zurückgibt. Wenn z.B. ein Konstruktor für die Klasse `java.util.Date` aufgerufen wird (`java.util.Date.new()`), so wird ein Objekt vom Typ `java.util.Date` zurückgegeben. Das lexikalische Format des zurückgegebenen Objekts stimmt unter Umständen nicht mit dem lexikalischen Format des XPath-Datentyps überein und der Wert müsste daher in das lexikalische Format des erforderlichen XPath-Datentyps und anschließend in den erforderlichen XPath-Datentyp konvertiert werden.

Ein von einem Konstruktor erstelltes Java-Objekt kann für zwei Zwecke verwendet werden:

- Es kann einer Variable zugewiesen werden:
```
<xsl:variable name="currentdate" select="date:new()"
xmlns:date="java:java.util.Date" />
```
- Es kann an eine Erweiterungsfunktion übergeben werden (*siehe Instanzmethode und Instanzfelder*[2244]):
```
<xsl:value-of select="date:toString(date:new())" xmlns:date="java:java.util.Date" />
```

30.2.2.1.4 Java: Statische Methoden und statische Felder

Eine statische Methode wird direkt über ihren Java-Namen und durch Angabe der Argumente für die Methode aufgerufen. Statische Felder (Methoden, die keine Argumente haben), wie z.B. die Konstantenwertfelder `E` und `PI` werden ohne Angabe eines Arguments aufgerufen.

XSLT-Beispiele

Hier sehen Sie einige Beispiele dafür, wie statische Methoden und Felder aufgerufen werden können:

```
<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos(3.14)" />

<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos( jMath:PI() )" />

<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:E() * jMath:cos(3.14)" />
```

Beachten Sie, dass die Erweiterungsfunktionen die Form `prefix:fname()` haben. Das Präfix ist in allen drei Fällen `jMath:`. Es ist mit der Namespace URI `java:java.lang.Math` verknüpft. (Die Namespace URI muss mit `java:`. beginnen. In den obigen Beispielen wurde es um den Klassennamen erweitert (`java.lang.Math`).) Der Teil `fname()` der Erweiterungsfunktionen muss mit dem Namen der öffentlichen Klasse (z.B.

`java.lang.Math`) gefolgt vom Namen einer öffentlichen statischen Methode mit ihrem/ihren Argument(en) (wie z.B. `(3.14)`) oder einem öffentlichen statischen Feld (z.B. `PI()`) übereinstimmen.

In den obigen Beispielen wurde der Klassenname in die Namespace URI inkludiert. Wäre sie nicht in der Namespace URI enthalten, müsste sie in den `fname()` Teil der Erweiterungsfunktion inkludiert werden. Z.B:

```
<xsl:value-of xmlns:java="java:"
              select="java:java.lang.Math.cos(3.14)" />
```

XQuery-Beispiel

Ein ähnliches Beispiel in XQuery wäre:

```
<cosine xmlns:jMath="java:java.lang.Math">
   {jMath:cos(3.14)}
</cosine>
```

30.2.2.1.5 Java: Instanzmethoden und Instanzfelder

Bei einer Instanzmethode wird als erstes Argument eines Methodenaufrufs ein Java-Objekt an die Methode übergeben. Ein solches Java-Objekt würde normalerweise mit Hilfe einer Erweiterungsfunktion (z.B. eines Konstruktoraufrufs) oder eines Stylesheet-Parameters/einer Stylesheet-Variablen erstellt. Ein XSLT-Beispiel dafür wäre:

```
<xsl:stylesheet version="1.0" exclude-result-prefixes="date"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:date="java:java.util.Date"
  xmlns:jlang="java:java.lang">
  <xsl:param name="CurrentDate" select="date:new()"/>
  <xsl:template match="/">
      <enrollment institution-id="Altova School"
                date="{date:toString($CurrentDate)}"
                type="{jlang:Object.toString(jlang:Object.getClass( date:new() ))}">
      </enrollment>
  </xsl:template>
</xsl:stylesheet>
```

Im Beispiel oben wird der Wert des Node `enrollment/@type` folgendermaßen erstellt:

1. Es wird ein Objekt mit einem Konstruktor für die Klasse `java.util.Date` (mit dem Konstruktor `date:new()`) erstellt.
2. Dieses Java-Objekt wird als das Argument der Methode `jlang.Object.getClass` übergeben.
3. Das mit der Methode `getClass` abgerufene Objekt wird als das Argument an die Methode `jlang.Object.toString` übergeben.

Das Ergebnis (der Wert von `@type`) ist ein String, der den Wert `java.util.Date` hat.

Ein Instanzfeld unterscheidet sich theoretisch insofern von einer Instanzmethode, als es sich nicht um ein Java-Objekt per se handelt, das als Argument an das Instanzfeld übergeben wird. Stattdessen wird ein Parameter oder eine Variable als Argument übergeben. Der Parameter/die Variable kann allerdings selbst den Wert

© 2016-2022 Altova GmbH

enthalten, der von einem Java-Objekt zurückgegeben wird. So erhält z.B. der Parameter `CurrentDate` den Wert, der von einem Konstruktor für die Klasse `java.util.Date` zurückgegeben wird. Dieser Wert wird anschließend als Argument an die Instanzmethode `date:toString` übergeben, um den Wert von `/enrollment/@date` bereitzustellen.

30.2.2.1.6 Datentypen: XPath/XQuery in Java

Wenn von einem XPath/XQuery-Ausdruck aus eine Java-Funktion aufgerufen wird, spielt der Datentyp der Argumente der Funktion eine wichtige Rolle, welche von mehreren Java-Klassen desselben Namens aufgerufen wird.

In Java gelten die folgenden Regeln:

- Wenn es mehr als eine Java-Methode mit demselben Namen gibt, jede aber eine andere Anzahl von Argumenten als die andere(n) hat, so wird die Java-Methode ausgewählt, die der Anzahl der Argumente im Funktionsaufruf am ehesten entspricht.
- Die XPath/XQuery-Datentypen "string", "number" und "boolean" (*siehe Liste unten*) werden implizit in einen entsprechenden Java-Datentyp konvertiert. Wenn der bereitgestellte XPath/XQuery-Datentyp in mehr als einen Java-Typ konvertiert werden kann (z.B: `xs:integer`), so wird jener Java-Typ ausgewählt, der für die ausgewählte Methode deklariert wurde. Wenn die aufgerufene Java-Methode z.B. `fx(decimal)` und der bereitgestellte XPath/XQuery-Datentyp `xs:integer` ist, so wird `xs:integer` in den Java-Datentyp `decimal` konvertiert.

In der Tabelle unten sehen Sie eine Liste der impliziten Konvertierungen der XPath/XQuery-Datentypen "string", "number" und "boolean"in Java-Datentypen.

`xs:string`	`java.lang.String`
`xs:boolean`	`boolean` (primitive), `java.lang.Boolean`
`xs:integer`	int, long, short, byte, float, double und die Wrapper-Klassen davon wie z.B. `java.lang.Integer`
`xs:float`	`float` (primitive), `java.lang.Float`, `double` (primitive)
`xs:double`	`double` (primitive), `java.lang.Double`
`xs:decimal`	`float` (primitive), `java.lang.Float`, `double`(primitive), `java.lang.Double`

Die oben aufgelisteten Subtypen von XML-Schema-Datentypen (die in XPath und XQuery verwendet werden) werden ebenfalls in den/die Java-Typ(en), der/die dem übergeordneten Subtyp entsprechen, konvertiert.

In einigen Fällen ist es nicht möglich, auf Basis der verfügbaren Informationen die richtige Java-Methode auszuwählen. Nehmen Sie als Beispiel den folgenden Fall.

- Das bereitgestellte Argument ist ein `xs:untypedAtomic` Wert 10 und ist für die Methode `mymethod(float)` bestimmt.

- Es gibt jedoch eine weitere Methode in der Klasse, die ein Argument eines anderen Datentyps erhält: `mymethod(double)`.
- Da die Methodennamen dieselben sind und der bereitgestellte Typ (`xs:untypedAtomic`) sowohl in `float` als auch `double` korrekt konvertiert werden könnte, kann es geschehen, dass `xs:untypedAtomic` in `double` anstelle von `float` konvertiert wird.
- Infolgedessen handelt es sich dann bei der ausgewählten Methode nicht um die benötigte Methode, sodass nicht das erwartete Ergebnis erzielt wird. Als Umgehungslösung können Sie eine benutzerdefinierte Methode mit einem anderen Namen erstellen und diese Methode verwenden.

Typen, die in der Liste oben nicht enthalten sind (z.B. `xs:date`), werden nicht konvertiert und generieren einen Fehler. Beachten Sie jedoch, dass es in einigen Fällen unter Umständen möglich ist, den benötigten Java-Typ mittels eines Java-Konstruktors zu erstellen.

30.2.2.1.7 Datentypen: Java in XPath/XQuery

Wenn eine Java-Methode einen Wert zurückgibt und der Datentyp des Werts "string", "numeric" oder "boolean" ist, wird anschließend in den entsprechenden XPath/XQuery-Typ konvertiert. So werden z.B. die Java-Datentypen `javaa.lang.Boolean` und `boolean` in `xsd:boolean` konvertiert.

Von Funktionen zurückgegebene eindimensionale Arrays werden zu einer Sequenz erweitert. Mehrdimensionale Arrays werden nicht konvertiert und sollten daher in einen Wrapper gesetzt werden.

Wenn ein wrapped Java-Objekt oder ein Datentyp zurückgegeben wird, bei dem es sich nicht um den Typ "string", "numeric" oder "boolean" handelt, können Sie sicherstellen, dass die Konvertierung in den benötigten XPath/XQuery-Typ erfolgt, indem Sie zuerst eine Java-Methode (e.g `toString`) verwenden, um das Java-Objekt in einen String zu konvertieren. In XPath/XQuery kann der String geändert werden, damit er der lexikalischen Darstellung des benötigten Typs entspricht, und anschließend z.B. mit Hilfe des Ausdrucks `cast as` in den benötigten Typ konvertiert werden.

30.2.2.2 .NET-Erweiterungsfunktionen

Wenn Sie auf einem Windows-Rechner mit der .NET-Plattform arbeiten, können Sie Erweiterungsfunktionen verwenden, die in jeder beliebigen der .NET-Sprachen geschrieben wurden (z.B. C#). Eine .NET Erweiterungsfunktion kann in einem XPath- oder XQuery-Ausdruck verwendet werden, um einen Konstruktor, eine Eigenschaft oder Methode (statische oder Instanz) in einer .NET-Klasse aufzurufen.

Eine Eigenschaft einer .NET-Klasse wird mit der Syntax `get_PropertyName()` aufgerufen.

Dieser Abschnitt ist in die folgenden Unterabschnitte gegliedert:

- .NET: Konstruktoren 2249
- .NET: Statische Methoden und statische Felder 2249
- .NET: Instanzmethoden und Instanzfelder 2250
- Datentypen: XPath/XQuery in .NET 2251
- Datentypen: .NET in XPath/XQuery 2252

 © 2016-2022 Altova GmbH

Form der Erweiterungsfunktion

Die Erweiterungsfunktion im XPath/XQuery-Ausdruck muss die folgende Form haben `präfix:fname()`.

- Der Teil `präfix:` ist mit einer URI verknüpft, die die benötigte .NET-Klasse definiert.
- Der Teil `fname()` identifiziert den Konstruktor, die Eigenschaft oder die Methode (statisch oder Instanz) innerhalb der .NET-Klasse und liefert alle gegebenenfalls benötigten Argumente.

- Die URI muss mit `clitype:` beginnen (welches die Funktion als .NET-Erweiterungsfunktion kennzeichnet).
- Die Form `prefix:fname()` der Erweiterungsfunktion kann mit Systemklassen und mit Klassen in einer geladenen Assembly verwendet werden. Wenn eine Klasse allerdings geladen werden muss, müssen zusätzliche Parameter mit den benötigten Informationen bereitgestellt werden.

Parameter

Zum Laden einer Assembly werden die folgenden Parameter verwendet:

`asm`	Der Name der zu ladenden Assembly
`ver`	Die Versionsnummer: eine Maximalzahl von vier Ganzzahlen, die durch Punkte getrennt sind
`sn`	Das Key Token des Strong Name der Assembly (16 Hex-Stellen).
`from`	Eine URI gibt den Pfad der zu ladenden Assembly (DLL) an. Wenn die URI relativ ist, ist sie relativ zum XSLT- oder XQuery-Dokument. Wenn dieser Parameter vorhanden ist, werden alle anderen Parameter ignoriert.
`partialname`	Der partielle Name der Assembly. Er wird für `Assembly.LoadWith.PartialName()` bereitgestellt, welches versuchen wird, die Assembly zu laden. Wenn `partialname` vorhanden ist, werden alle anderen Parameter ignoriert.
`loc`	Die Locale, z.B. `en-US`. Die Standardeinstellung ist `neutral`

Wenn die Assembly aus einer DLL geladen werden soll, verwenden Sie den `from` Parameter und lassen Sie den `sn` Parameter weg. Wenn die Assembly aus dem Global Assembly Cache (GAC) geladen werden soll, verwenden Sie den `sn` Parameter und lassen Sie den `from` Parameter weg.

Vor dem ersten Parameter muss ein Fragezeichen eingefügt werden. Parameter müssen durch ein Semikolon getrennt werden. Der Wert des Parameternamens wird durch ein Ist-Gleich-Zeichen angegeben (*siehe Beispiele unten*).

Beispiele für Namespace-Deklarationen

Ein Beispiel für eine Namespace Deklaration in XSLT, die die Systemklasse `System.Environment`: identifiziert.

```
xmlns:myns="clitype:System.Environment"
```

Ein Beispiel für eine Namespace Deklaration in XSLT, die die zu ladende Klasse als `Trade.Forward.Scrip`: identifiziert.

```
xmlns:myns="clitype:Trade.Forward.Scrip?asm=forward;version=10.6.2.1"
```

Ein Beispiel für eine Namespace-Deklaration in XQuery, die die Systemklasse `MyManagedDLL.testClass` identifiziert. Es werden zwei Klassen unterschieden:

1. Wenn die Assembly aus dem GAC geladen wird:
   ```
   declare namespace cs="clitype:MyManagedDLL.testClass?asm=MyManagedDLL;
          ver=1.2.3.4;loc=neutral;sn=b9f091b72dccfba8";
   ```

2. Wenn die Assembly aus der DLL geladen wird (vollständige und partielle Referenzen unten):
   ```
          declare namespace cs="clitype:MyManagedDLL.testClass?from=file:///C:/Altova
          Projects/extFunctions/MyManagedDLL.dll;

   declare namespace cs="clitype:MyManagedDLL.testClass?from=MyManagedDLL.dll;
   ```

XSLT-Beispiel

Hier sehen Sie ein vollständiges XSLT-Beispiel, in dem Funktionen in der Systemklasse `System.Math`: aufgerufen werden:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions">
   <xsl:output method="xml" omit-xml-declaration="yes" />
   <xsl:template match="/">
      <math xmlns:math="clitype:System.Math">
         <sqrt><xsl:value-of select="math:Sqrt(9)"/></sqrt>
         <pi><xsl:value-of select="math:PI()"/></pi>
         <e><xsl:value-of select="math:E()"/></e>
         <pow><xsl:value-of select="math:Pow(math:PI(), math:E())"/></pow>
      </math>
   </xsl:template>
</xsl:stylesheet>
```

Die Namespace-Deklaration für das Element `math` verknüpft das Präfix `math:` mit der URI `clitype:System.Math`. Der Beginn der URI `clitype:` gibt an, dass danach entweder eine Systemklasse oder eine geladene Klasse definiert wird. Das Präfix `math:` im XPath-Ausdruck verknüpft die Erweiterungsfunktionen mit der URI (und durch Erweiterung der Klasse) `System.Math`. Die Erweiterungsfunktionen identifizieren Methoden in der Klasse `System.Math` und stellen Argumente bereit, wo dies erforderlich ist.

XQuery-Beispiel

Hier sehen Sie ein XQuery-Beispielfragment ähnlich dem XSLT-Beispiel oben:

```
<math xmlns:math="clitype:System.Math">
   {math:Sqrt(9)}
</math>
```


© 2016-2022 Altova GmbH

Wie beim XSLT-Beispiel weiter oben identifiziert die Namespace-Deklaration die .NET-Klasse, in diesem Fall eine Systemklasse. Der XQuery-Ausdruck identifiziert die aufzurufenden Methode und liefert das Argument.

30.2.2.2.1 .NET: Konstruktoren

Eine Erweiterungsfunktion kann verwendet werden, um einen .NET-Konstruktor aufzurufen. Alle Konstruktoren werden mit der Pseudofunktion `new()` aufgerufen. Wenn es mehrere Konstruktoren für eine Klasse gibt, wird der Konstruktor ausgewählt, der der Anzahl der bereitgestellten Argumente am ehesten entspricht. Wenn kein passender Konstruktor gefunden wird, der den bereitgestellten Argumenten entspricht, wird die Fehlermeldung `'No constructor found'` zurückgegeben.

Konstruktoren, die XPath/XQuery-Datentypen zurückgeben

Wenn das Ergebnis eines .NET-Konstruktors implizit in XPath/XQuery-Datentypen konvertiert werden kann 2246, gibt die .NET-Erweiterungsfunktion eine Sequenz zurück, bei der es sich um einen XPath/XQuery-Datentyp handelt.

Konstruktoren, die .NET-Objekte zurückgeben

Wenn das Ergebnis eines .NET-Konstruktoraufrufs nicht in einen passenden XPath/XQuery-Datentyp konvertiert werden kann, erstellt der Konstruktor ein wrapped .NET-Objekt mit einem Typ, der der Name der Klasse ist, die dieses Objekt zurückgibt. Wenn z.B. ein Konstruktor für die Klasse `System.DateTime` aufgerufen wird (mit `System.DateTime.new()`), so wird ein Objekt mit dem Typ `System.DateTime` zurückgegeben.

Das lexikalische Format des zurückgegebenen Objekts stimmt unter Umständen nicht mit dem lexikalischen Format eines erforderlichen XPath-Datentyps überein. In solchen Fällen müsste der zurückgegebene Wert: (i) in das lexikalische Format des benötigten XPath-Datentyps konvertiert werden; und (ii) in den erforderlichen XPath-Datentyp konvertiert werden.

Ein von einem Konstruktor erstelltes .NET-Objekt kann für drei Zwecke verwendet werden:

- Es kann innerhalb einer Variable verwendet werden:
  ```
  <xsl:variable name="currentdate" select="date:new(2008, 4, 29)"
  xmlns:date="clitype:System.DateTime" />
  ```
- Es kann an eine Erweiterungsfunktion übergeben werden (*siehe Instanzmethode und Instanzfelder* 2244):
  ```
  <xsl:value-of select="date:ToString(date:new(2008, 4, 29))"
  xmlns:date="clitype:System.DateTime" />
  ```
- Es kann in einen String, eine Zahl oder einen Booleschen Ausdruck konvertiert werden:
- ```
  <xsl:value-of select="xs:integer(date:get_Month(date:new(2008, 4, 29)))"
  xmlns:date="clitype:System.DateTime" />
  ```

30.2.2.2.2 .NET: Statische Methoden und statische Felder

Eine statische Methode wird direkt über ihren Namen und durch Angabe der Argumente für die Methode aufgerufen. Der im Aufruf verwendete Name muss exakt mit einer öffentlichen statischen Methode in der
  ```

angegebenen Klasse übereinstimmen. Wenn der Methodenname und die Anzahl der in der Funktion angegebenen Argumente mit mehr als einer Methode in einer Klasse übereinstimmen, werden die Typen der bereitgestellten Argumente nach der besten Übereinstimmung überprüft. Wenn keine eindeutig passende Methode gefunden werden kann, wird ein Fehler ausgegeben.

Anmerkung: Ein Feld in einer .NET-Klasse wird als Methode ohne Argument betrachtet. Eine Eigenschaft wird mit der Syntax `get_PropertyName()` aufgerufen.

Beispiele

Ein XSLT-Beispiel, in dem Sie einen Methodenaufruf mit einem Argument (`System.Math.Sin(arg)`) sehen:

```
<xsl:value-of select="math:Sin(30)" xmlns:math="clitype:System.Math"/>
```

Ein XSLT-Beispiel, in dem Sie einen Aufruf eines Felds (wird als Methode ohne Argument betrachtet) sehen (`System.Double.MaxValue()`):

```
<xsl:value-of select="double:MaxValue()" xmlns:double="clitype:System.Double"/>
```

Ein XSLT-Beispiel, in dem Sie einen Aufruf einer Eigenschaft (Syntax ist `get_PropertyName()`) (`System.String()`) sehen:

```
<xsl:value-of select="string:get_Length('my string')"
xmlns:string="clitype:System.String"/>
```

Ein XQuery-Beispiel, in dem Sie einen Aufruf einer Methode mit einem Argument (`System.Math.Sin(arg)`) sehen:

```
<sin xmlns:math="clitype:System.Math">
   { math:Sin(30) }
</sin>
```

30.2.2.2.3 .NET: Instanzmethoden und Instanzfelder

Bei einer Instanzmethode wird als erstes Argument des Methodenaufrufs ein .NET-Objekt an die Methode übergeben. Dieses .NET-Objekt wird normalerweise mit Hilfe einer Erweiterungsfunktion (z.B. durch einen Konstruktoraufruf) oder einen Stylesheet-Parameter/eine Stylesheet-Variable erstellt. Ein XSLT-Beispiel dieser Art wäre:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions">
   <xsl:output method="xml" omit-xml-declaration="yes"/>
   <xsl:template match="/">
      <xsl:variable name="releasedate"
         select="date:new(2008, 4, 29)"
```

 © 2016-2022 Altova GmbH

```
            xmlns:date="clitype:System.DateTime"/>
      <doc>
         <date>
            <xsl:value-of select="date:ToString(date:new(2008, 4, 29))"
               xmlns:date="clitype:System.DateTime"/>
         </date>
         <date>
            <xsl:value-of select="date:ToString($releasedate)"
               xmlns:date="clitype:System.DateTime"/>
         </date>
      </doc>
   </xsl:template>
</xsl:stylesheet>
```

Im Beispiel oben wird ein `System.DateTime` Konstruktor (`new(2008, 4, 29)`) verwendet, um ein .NET-Objekt vom Typ `System.DateTime` zu erstellen. Diese Objekt wird zweimal erstellt, einmal als Wert der Variablen `releasedate`, ein zweites Mal als das erste und einzige Argument der Methode `System.DateTime.ToString()`. Die Instanzmethode `System.DateTime.ToString()` wird zwei Mal aufgerufen, beide Male mit dem `System.DateTime` Konstruktor (`new(2008, 4, 29)`) als erstem und einzigem Argument. In einer dieser Instanzen wird die Variable `releasedate` verwendet, um das .NET-Objekt abzurufen.

Instanzmethoden und Instanzfelder

Der Unterschied zwischen einer Instanzmethode und einem Instanzfeld ist ein theoretischer. In einer Instanzmethode wird ein .NET-Objekt direkt als Argument übergeben; in einem Instanzfeld wird stattdessen ein Parameter oder eine Variable übergeben - auch wenn der Parameter bzw. die Variable selbst ein .NET-Objekt enthalten kann. So enthält z.B. die Variable `releasedate` im Beispiel oben ein .NET-Objekt und es ist diese Variable, die als das Argument von `ToString()` an den zweiten `date` Elementkonstruktor übergeben wird. Die `ToString()` Instanz im ersten `date` Element ist daher eine Instanzmethode, während die zweite als Instanzfeld betrachtet wird. Das in beiden Instanzen erzeugte Ergebnis ist jedoch dasselbe.

30.2.2.2.4 Datentypen: XPath/XQuery in .NET

Wenn in einem XPath/XQuery-Ausdruck eine .NET-Erweiterungsfunktion verwendet wird, spielen die Datentypen der Argumente der Funktion eine wichtige Rolle bei der Entscheidung, welche der vielen .NET-Methoden mit demselben Namen aufgerufen werden soll.

In .NET gelten die folgenden Regeln:

- Wenn es mehr als eine Methode mit demselben Namen in einer Klasse gibt, so stehen nur die Methoden zur Auswahl, die dieselbe Anzahl von Argumenten wie der Funktionsaufruf haben.
- Die XPath/XQuery-Datentypen "string", "number" und "boolean" (*siehe Liste unten*) werden implizit in einen entsprechenden .NET-Datentyp konvertiert. Wenn der bereitgestellte XPath/XQuery-Datentyp in mehr als einen .NET-Typ konvertiert werden kann (z.B: `xs:integer`), so wird jener .NET-Typ ausgewählt, der für die ausgewählte Methode deklariert wurde. Wenn die aufgerufene .NET-Methode z.B. `fx(double)` und der bereitgestellte XPath/XQuery-Datentyp `xs:integer` ist, so wird `xs:integer` in den .NET-Datentyp `double`

In der Tabelle unten sehen Sie eine Liste der impliziten Konvertierungen der XPath/XQuery-Datentypen "string", "number" und "boolean"in .NET-Datentypen.

xs:string	StringValue, string
xs:boolean	BooleanValue, bool
xs:integer	IntegerValue, decimal, long, integer, short, byte, double, float
xs:float	FloatValue, float, double
xs:double	DoubleValue, double
xs:decimal	DecimalValue, decimal, double, float

Die oben aufgelisteten Subtypen von XML-Schema-Datentypen (die in XPath und XQuery verwendet werden) werden ebenfalls in den/die .NET-Typ(en), der/die dem übergeordneten Subtyp entsprechen, konvertiert.

In einigen Fällen ist es nicht möglich, auf Basis der verfügbaren Informationen die richtige .NET-Methode auszuwählen. Nehmen Sie als Beispiel den folgenden Fall.

- Das bereitgestellte Argument ist ein `xs:untypedAtomic` Wert 10 und ist für die Methode `mymethod(float)` bestimmt.
- Es gibt jedoch eine weitere Methode in der Klasse, die ein Argument eines anderen Datentyps erhält: `mymethod(double)`.
- Da die Methodennamen dieselben sind und der bereitgestellte Typ (`xs:untypedAtomic`) sowohl in `float` als auch `double` korrekt konvertiert werden könnte, kann es geschehen, dass `xs:untypedAtomic` in `double` anstelle von `float` konvertiert wird.
- Infolgedessen handelt es sich dann bei der ausgewählten Methode nicht um die benötigte Methode, sodass nicht das erwartete Ergebnis erzielt wird. Als Umgehungslösung können Sie eine benutzerdefinierte Methode mit einem anderen Namen erstellen und diese Methode verwenden.

Typen, die in der Liste oben nicht enthalten sind (z.B. `xs:date`), werden nicht konvertiert und generieren einen Fehler.

30.2.2.2.5 Datentypen: .NET in XPath/XQuery

Wenn eine .NET-Methode einen Wert zurückgibt und der Datentyp des Werts "string", "numeric" oder "boolean" ist, wird er anschließend in den entsprechenden XPath/XQuery-Typ konvertiert. So wird z.B. der .NET-Datentyp `decimal` in `xsd:decimal` konvertiert.

Wenn ein .NET-Objekt oder ein Datentyp zurückgegeben wird, bei dem es sich nicht um den Typ "string", "numeric" oder "boolean" handelt, können Sie sicherstellen, dass die Konvertierung in den benötigten XPath/XQuery-Typ erfolgt, indem Sie zuerst eine .NET-Methode (z.B. `System.DateTime.ToString()`) verwenden, um das .NET-Objekt in einen String zu konvertieren. In XPath/XQuery kann der String geändert werden, damit er der lexikalischen Darstellung des benötigten Typs entspricht, und anschließend z.B. mit Hilfe des Ausdrucks `cast as` in den benötigten Typ konvertiert werden.

© 2016-2022 Altova GmbH

30.2.2.3 MSXSL-Skripts für XSLT

Das Element `<msxsl:script>` enthält benutzerdefinierte Funktionen und Variablen, die von XPath-Ausdrücken im XSLT-Stylesheet aufgerufen werden können. Das Element `<msxsl:script>` ist ein Element der obersten Ebene, d.h. es muss ein Child-Element von `<xsl:stylesheet>` oder `<xsl:transform>` sein.

Das Element `<msxsl:script>` muss sich im Namespace `urn:schemas-microsoft-com:xslt` (*siehe Beispiel unten*) befinden.

Scripting-Sprache und Namespace

Die im Block verwendete Scripting-Sprache wird im Attribut `language` des Elements `<msxsl:script>` definiert und der für Funktionsaufrufe von XPath-Ausdrücken aus zu verwendende Namespace wird durch das Attribut `implements-prefix` (*siehe unten*) identifiziert.

```
<msxsl:script language="scripting-language implements-prefix="user-namespace-prefix">

   function-1 or variable-1
   ...
   function-n or variable-n

</msxsl:script>
```

Das Element `<msxsl:script>` interagiert mit der Windows Scripting Runtime. Daher können nur Sprachen, die auf Ihrem Rechner installiert sind, im Element `<msxsl:script>` verwendet werden. **Um MXSL Scripts verwenden zu können muss die Plattform .NET Framework 2.0 oder höher installiert sein.** Folglich können die .NET Scripting Sprachen innerhalb des Elements `<msxsl:script>` verwendet werden.

Das Attribut `language` akzeptiert dieselben Werte wie das Attribut `language` des HTML `<script>` Elements. Wenn das Attribut `language` nicht definiert ist, wird als Standardsprache Microsoft JScript verwendet.

Das Attribut `implements-prefix` erhält einen Wert, der ein Präfix eines deklarierten in-scope Namespace ist. Bei diesem Namespace handelt es sich normalerweise um einen Benutzer-Namespace, der für eine Funktionsbibliothek reserviert ist. Alle Funktionen und Variablen, die im Element `<msxsl:script>` definiert sind, werden sich im Namespace befinden, der durch das im Attribut `implements-prefix` definierte Präfixe identifiziert wird. Wenn eine Funktion von einem XPath-Ausdruck aus aufgerufen wird, muss sich der vollständig qualifizierte Funktionsname im selben Namespace wie die Funktionsdefinition befinden.

Beispiel

Hier sehen Sie ein Beispiel für ein vollständiges XSLT Stylesheet, in dem eine Funktion verwendet wird, die in einem `<msxsl:script>` Element definiert ist.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions"
   xmlns:msxsl="urn:schemas-microsoft-com:xslt"
   xmlns:user="http://mycompany.com/mynamespace">
```

```
  <msxsl:script language="VBScript" implements-prefix="user">
    <![CDATA[
    ' Input: A currency value: the wholesale price
    ' Returns: The retail price: the input value plus 20% margin,
    ' rounded to the nearest cent
    dim a as integer  = 13
    Function AddMargin(WholesalePrice) as integer
      AddMargin = WholesalePrice * 1.2 + a
    End Function
  ]]>
  </msxsl:script>

  <xsl:template match="/">
    <html>
      <body>
        <p>
          <b>Total Retail Price =
            $<xsl:value-of select="user:AddMargin(50)"/>
          </b>
          <br/>
          <b>Total Wholesale Price =
            $<xsl:value-of select="50"/>
          </b>
        </p>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

Datentypen

Die Werte von Parametern, die an und aus dem Script-Block heraus übergeben werden, sind auf XPath-Datentypen beschränkt. Diese Einschränkung gilt nicht für Daten, die zwischen Funktionen und Variablen innerhalb des Script-Blocks übergeben werden.

Assemblies

Eine Assembly kann über das Element **msxsl:assembly** in das Script importiert werden. Die Assembly wird über einen Namen oder eine URL identifiziert. Die Assembly wird beim Kompilieren des Stylesheet importiert. Hier sehen Sie ein einfaches Beispiel, wie das Element `msxsl:assembly` zu verwenden ist.

```
<msxsl:script>
       <msxsl:assembly name="myAssembly.assemblyName" />
       <msxsl:assembly href="pathToAssembly" />

       ...

</msxsl:script>
```

Der Assembly-Name kann ein vollständiger Name sein, wie z.B.:

© 2016-2022 Altova GmbH

```
"system.Math, Version=3.1.4500.1 Culture=neutral PublicKeyToken=a46b3f648229c514"
```

oder ein Kurzname wie z.B. `"myAssembly.Draw"`.

Namespaces

Namespaces können mit dem Element **`msxsl:using`** deklariert werden. Auf diese Art können Assembly-Klassen ohne ihre Namespaces in das Script geschrieben werden, wodurch Sie sich das mühsame Eintippen ersparen. Hier sehen Sie, wie das Element `msxsl:using` verwendet wird, um Namespaces zu deklarieren.

```
<msxsl:script>
       <msxsl:using namespace="myAssemblyNS.NamespaceName" />

       ...

</msxsl:script>
```

Der Wert des `namespace` Attributs ist der Name des Namespace.

30.3 Datentypen in DB-generierten XML-Schemas

Bei Generierung eines XML-Schemas anhand einer Datenbank (DB), werden die Datentypen, die für diese DB spezifisch sind, in XML-Schema-Datentypen konvertiert. In diesem Anhang finden Sie Informationen zum Mapping von DB-Datentypen auf XML-Schema-Datentypen für gebräuchliche Datenbanken. Wählen Sie die gewünschte Datenbank aus der Liste aus.

- ADO 2256
- MS Access 2257
- MS SQL Server 2258
- MySQL 2258
- ODBC 2259
- Oracle 2260
- Sybase 2261

30.3.1 ADO

Wenn anhand einer ADO-Datenbank ADO database (DB) ein XML-Schema generiert wird, werden die ADO-DB-Datentypen, wie in der Tabelle unten aufgelistet, in XML-Schema-Datentypen konvertiert.

ADO-Datentyp	XML-Schema-Datentyp
adGUID	xs:ID
adChar	xs:string
adWChar	xs:string
adVarChar	xs:string
adWVarChar	xs:string
adLongVarChar	xs:string
adWLongVarChar	xs:string
adVarWChar	xs:string
adBoolean	xs:boolean
adSingle	xs:float
adDouble	xs:double
adNumeric	xs:decimal
adCurrency	xs:decimal
adDBTimeStamp	xs:dateTime
adDate	xs:date
adBinary	xs:base64Binary

© 2016-2022 Altova GmbH

adVarBinary	xs:base64Binary
adLongVarBinary	xs:base64Binary
adInteger	xs:Integer
adUnsignedInt	xs:unsignedInt
adSmallInt	xs:short
adUnsignedSmallInt	xs:unsignedShort
adBigInt	xs:long
adUnsignedBigInt	xs:unsignedLong
adTinyInt	xs:byte
adUnsignedTinyInt	xs:unsignedByte

30.3.2 MS Access

Wenn anhand einer MS Access-Datenbank (DB) ein XML-Schema generiert wird, werden die MS Access-DB-Datentypen, wie in der Tabelle unten aufgelistet, in XML-Schema-Datentypen konvertiert.

MS Access-Datentyp	XML-Schema-Datentyp
GUID	xs:ID
char	xs:string
varchar	xs:string
memo	xs:string
bit	xs:boolean
Number(single)	xs:float
Number(double)	xs:double
Decimal	xs:decimal
Currency	xs:decimal
Date/Time	xs:dateTime
Number(Long Integer)	xs:integer
Number(Integer)	xs:short
Number(Byte)	xs:byte
OLE Object	xs:base64Binary

30.3.3 MS SQL Server

Wenn anhand einer MS SQL Server-Datenbank (DB) ein XML-Schema generiert wird, werden die MS SQL Server-DB-Datentypen, wie in der Tabelle unten aufgelistet, in XML-Schema-Datentypen konvertiert.

MS SQL Server-Datentyp	XML-Schema-Datentyp
uniqueidentifier	xs:ID
char	xs:string
nchar	xs:string
varchar	xs:string
nvarchar	xs:string
text	xs:string
ntext	xs:string
sysname	xs:string
bit	xs:boolean
real	xs:float
float	xs:double
decimal	xs:decimal
money	xs:decimal
smallmoney	xs:decimal
datetime	xs:dateTime
smalldatetime	xs:dateTime
binary	xs:base64Binary
varbinary	xs:base64Binary
image	xs:base64Binary
integer	xs:integer
smallint	xs:short
bigint	xs:long
tinyint	xs:byte

30.3.4 MySQL

Wenn anhand einer MySQL-Datenbank (DB) ein XML-Schema generiert wird, werden die MySQL-DB-Datentypen, wie in der Tabelle unten aufgelistet, in XML-Schema-Datentypen konvertiert.

© 2016-2022 Altova GmbH

MySQL-Datentyp	XML-Schema-Datentyp
char	xs:string
varchar	xs:string
text	xs:string
tinytext	xs:string
mediumtext	xs:string
longtext	xs:string
tinyint(1)	xs:boolean
float	xs:float
double	xs:double
decimal	xs:decimal
datetime	xs:dateTime
blob	xs:base64Binary
tinyblob	xs:base64Binary
mediumblob	xs:base64Binary
longblob	xs:base64Binary
smallint	xs:short
bigint	xs:long
tinyint	xs:byte

30.3.5 ODBC

Wenn anhand einer ODBC-Datenbank (DB) ein XML-Schema generiert wird, werden die ODBC-DB-Datentypen, wie in der Tabelle unten aufgelistet, in XML-Schema-Datentypen konvertiert.

ODBC-Datentyp	XML-Schema-Datentyp
SQL_GUID	xs:ID
SQL_CHAR	xs:string
SQL_VARCHAR	xs:string
SQL_LONGVARCHAR	xs:string
SQL_BIT	xs:boolean
SQL_REAL	xs:float

SQL_DOUBLE	xs:double
SQL_DECIMAL	xs:decimal
SQL_TIMESTAMP	xs:dateTime
SQL_DATE	xs:date
SQL_BINARY	xs:base64Binary
SQL_VARBINARY	xs:base64Binary
SQL_LONGVARBINARY	xs:base64Binary
SQL_INTEGER	xs:integer
SQL_SMALLINT	xs:short
SQL_BIGINT	xs:long
SQL_TINYINT	xs:byte

30.3.6 Oracle

Bei Generierung eines XML-Schemas anhand einer Oracle-Datenbank (DB) werden die Oracle DB-Datentypen wie unten in der Tabelle gezeigt in XML-Schema-Datentypen konvertiert.

Oracle-Datentyp	XML-Schema-Datentyp
ROWID	xs:ID
CHAR	xs:string
NCHAR	xs:string
VARCHAR2	xs:string
NVARCHAR2	xs:string
CLOB	xs:string
NCLOB	xs:string
NUMBER (mit angewendetem Check Constraint)*	xs:boolean
NUMBER	xs:decimal
FLOAT	xs:double
DATE	xs:dateTime
INTERVAL YEAR TO MONTH	xs:gYearMonth
BLOB	xs:base64Binary

* Wenn ein Check Constraint auf eine Spalte vom Datentyp `NUMBER` angewendet wird und der Check Constraint auf die Werte `0` oder `1` überprüft, wird der Datentyp `NUMBER` für diese Spalte in einen XML-

© 2016-2022 Altova GmbH

Schema-Datentyp `xs:boolean`. konvertiert. Dieser Mechanismus eignet sich zum Generieren eines Datentyps `xs:boolean` im generierten XML-Schema.

30.3.7 Sybase

Bei Generierung eines XML-Schemas anhand einer Sybase-Datenbank (DB) werden die Sybase DB-Datentypen, wie unten in der Tabelle gezeigt, in XML-Schema-Datentypen konvertiert.

Sybase-Datentyp	XML-Schema-Datentyp
char	xs:string
nchar	xs:string
varchar	xs:string
nvarchar	xs:string
text	xs:string
sysname-varchar(30)	xs:string
bit	xs:boolean
real	xs:float
float	xs:float
double	xs:double
decimal	xs:decimal
money	xs:decimal
smallmoney	xs:decimal
datetime	xs:dateTime
smalldatetime	xs:dateTime
timestamp	xs:dateTime
binary<=255	xs:base64Binary
varbinary<=255	xs:base64Binary
image	xs:base64Binary
integer	xs:integer
smallint	xs:short
tinyint	xs:byte

30.4 Datentypen in DB, die anhand von XML-Schemas generiert wurden

Bei Generierung einer DB-Struktur anhand eines XML-Schemas werden die für diese DB spezifischen Datentypen anhand von XML-Schema-Datentypen generiert. Informationen zum Mapping von XML-Schema-Datentypen auf DB-Datentypen bei gebräuchlichen Datenbanken finden Sie in diesem Anhang. Wählen Sie das gewünschte Kapitel aus der nachfolgenden Liste aus.

- MS Access 2262
- MS SQL Server 2263
- MySQL 2265
- Oracle 2267

30.4.1 MS Access

Bei Generierung eines XML-Schemas anhand einer MS Access-Datenbank (DB) werden die MS Access DB-Datentypen, wie unten in der Tabelle gezeigt, in XML-Schema-Datentypen konvertiert.

XML-Schema-Datentyp	MS Access-Datentyp
xs:ID	GUID
xs:string	If no facets varchar (255)
	Size = either length or maxLength
	If Size <= 255 varchar (n)
	else memo
xs:normalizedString	Same as xs:string
xs:token	Same as xs:string
xs:Name	Same as xs:string
xs:NCName	Same as xs:string
xs:anyURI	Same as xs:string
xs:QName	Same as xs:string
xs:NOTATION	Same as xs:string
xs:boolean	bit
xs:float	Number (single)
xs:double	Number (double)
xs:decimal	Decimal
xs:duration	Date/Time

© 2016-2022 Altova GmbH

xs:dateTime	Date/Time
xs:time	Date/Time
xs:date	Date/Time
xs:gYearMonth	Date/Time
xs:gYear	Date/Time
xs:gMonthDay	Date/Time
xs:gDay	Date/Time
xs:gMonth	Date/Time
xs:hexBinary	If no facets varbinary (255)
	Size = either length or maxLength
	If Size <= 8000 varbinary
	else image (OLE Object)
xs:base64Binary	Same as xs:hexBinary
xs:integer	Number (Long Integer)
xs:int	Number (Long Integer)
xs:negativeInteger	Number (Long Integer); value constraint
xs:positiveInteger	Number (Long Integer); value constraint
xs:nonNegativeInteger	Number (Long Integer); value constraint
xs:nonPositiveInteger	Number (Long Integer); value constraint
xs:unsignedInt	Number (Long Integer)
xs:short	-- no equivalent --
xs:unsignedShort	-- no equivalent --
xs:long	-- no equivalent --
xs:unsignedLong	-- no equivalent --
xs:byte	Number (Byte)
xs:unsignedByte	Number (Byte)

30.4.2 MS SQL Server

Bei Generierung eines XML-Schemas anhand einer MS SQL-Datenbank (DB) werden die MS SQL Server DB-Datentypen, wie unten in der Tabelle gezeigt, in XML-Schema-Datentypen konvertiert.

XML-Schema-Datentyp	MS SQL Server-Datentyp

ID	uniqueidentifier
xs:string	If no facets
	{ if UNICODE nvarchar (255)
	else varchar (255) }
	else
	{ if UNICODE
	(Size = either length or maxLength)
	If Size <= 4000
	if FacetLengthIsSet then nChar
	else nVarChar
	if Size <= 1073741823 then nText }
	else
	{ if NON-UNICODE
	(Size = either length or maxLength)
	If Size <= 8000
	if FacetLengthIsSet then char
	else varchar
	if Size <= 2147483647 then text }
xs:normalizedString	Same as xs:string
xs:token	Same as xs:string
xs:Name	Same as xs:string
xs:NCName	Same as xs:string
xs:anyURi	Same as xs:string
xs:QName	Same as xs:string
xs:NOTATION	Same as xs:string
xs:boolean	bit
xs:float	real
xs:double	float
xs:decimal	decimal
xs:duration	datetime
xs:dateTime	datetime
xs:time	datetime

© 2016-2022 Altova GmbH

xs:date	datetime
xs:gYearMonth	datetime
xs:gYear	datetime
xs:gMonthDay	datetime
xs:gDay	datetime
xs:gMonth	datetime
xs:hexBinary	If no facets varbinary (255)
	(Size = either length or maxLength
	If Size <= 8000
	if FacetLengthIsSet then binary
	else varbinary
	if Size <= 2147483647 then image
xs:base64Binary	Same as xs:hexBinary
xs:integer	int
xs:int	int
xs:negativeInteger	Int (constrained to {...,-2,-1})
xs:positiveInteger	Int (constrained to {1,2,...})
xs:nonNegativeInteger	int (constrained to {0,1,2,...})
xs:nonPositiveInteger	int (constrained to {...,-2,-1,0})
xs:unsignedInt	int (additional constraints)
xs:short	smallint
xs:unsignedShort	smallint (additional constraints)
xs:long	bigint
xs:unsignedLong	bigint (additional constraints)
xs:byte	tinyint
xs:unsignedByte	tinyint (additional constraints)

30.4.3 MySQL

Bei Generierung eines XML-Schemas anhand einer MySQL-Datenbank (DB) werden die MySQL-Datentypen wie unten in der Tabelle gezeigt in XML-Schema-Datentypen konvertiert.

XML-Schema-Datentyp	MySQL-Datentyp
xs:ID	varchar(255)
xs:string	If no facets then varchar (255)
	else if facet length is set and <= 255 then char
	else if facet maxLength set and <= 255 then varchar
	else if maxLength is set and <= 65545 then text
	else if maxlength is set and <= 16777215 then mediumtext
	else if maxlength is set and <= 429496295 then longtext
xs:normalizedString	*Same as* xs:string
xs:token	*Same as* xs:string
xs:Name	*Same as* xs:string
xs:NCName	*Same as* xs:string
xs:anyURI	*Same as* xs:string
xs:QName	*Same as* xs:string
xs:NOTATION	*Same as* xs:string
xs:boolean	tinyint(1)
xs:float	float
xs:double	double
xs:decimal	decimal
xs:duration	timestamp
xs:dateTime	datetime
xs:time	time
xs:date	date
xs:gYearMonth	timestamp(4)
xs:gYear	year(4)
xs:gMonthDay	timestamp(8); constraints to check month, day
xs:gDay	timestamp(8); constraints to check day
xs:gMonth	timestamp(8); constraints to check month
xs:hexBinary	If no facets then blob (255)

© 2016-2022 Altova GmbH

	else if facet length is set and <= 255 then blob
	else if facet maxLength is set and <= 255 then tinyblob
	else if maxlength is set and <= 65545 then blob
	else if maxlength is set and <= 16777215 then mediumblob
	else if maxlength is set and <= 429496295 then longblob
xs:base64Binary	*Same as* xs:hexBinary
xs:integer	Integer
xs:int	int
xs:negativeInteger	Integer (constrained to {...,-2,-1})
xs:positiveInteger	Integer (constrained to {1,2,...})
xs:nonNegativeInteger	Integer (constrained to {0,1,2,...})
xs:nonPositiveInteger	Integer (constrained to {...,-2,-1,0})
xs:unsignedInt	Int (additional constraints)
xs:short	Smallint
xs:unsignedShort	Smallint (additional constraints)
xs:long	Bigint
xs:unsignedLong	Bigint (additional constraints)
xs:byte	Tinyint
xs:unsignedByte	Tinyint (additional constraints)

30.4.4 Oracle

Bei Generierung eines XML-Schemas anhand einer Oracle-Datenbank (DB) werden die Oracle DB-Datentypen, wie unten in der Tabelle gezeigt, in XML-Schema-Datentypen konvertiert.

XML-Schema-Datentyp	Oracle-Datentyp
xs:ID	ROWID
xs:string	If no facets
	if UNICODE then NVARCHAR2 (255)
	else VARCHAR2 (255)

	else if UNICODE
	(Size = either length or maxLength)
	If Size <= 2000 then NCHAR
	if Size <= 4000 then NVARHCAR2
	if Size <= 4 Gigabytes then NCLOB
	else if NON-UNICODE
	(Size = either length or maxLength)
	If Size <= 2000 then CHAR
	if Size <= 4000 then VARCHAR2
	if Size <= 4 Gigabytes then CLOB
xs:normalizedString	*Same as* xs:string
xs:token	*Same as* xs:string
xs:Name	*Same as* xs:string
xs:NCName	*Same as* xs:string
xs:anyURI	*Same as* xs:string
xs:QName	*Same as* xs:string
xs:NOTATION	*Same as* xs:string
xs:boolean	NUMBER with constraint Boolean
xs:float	FLOAT
xs:double	FLOAT
xs:decimal	NUMBER
xs:duration	TIMESTAMP
xs:dateTime	TIMESTAMP
xs:time	DATE
xs:date	DATE
xs:gYearMonth	INTERVAL YEAR TO MONTH
xs:gYear	DATE
xs:gMonthDay	DATE
xs:gDay	DATE
xs:gMonth	DATE
xs:hexBinary	if no facets then RAW (255)
	(Size = either length or maxLength)

© 2016-2022 Altova GmbH

	If Size <= 2000 then RAW (X)
	else Size <= 2 Gigabytes then LONG RAW (X)
	if Size <= 4 Gigabytes then BLOB (X)
xs:base64Binary	BLOB
xs:integer	NUMBER
xs:int	NUMBER
xs:negativeInteger	NUMBER (constrained to {...,-2,-1})
xs:positiveInteger	NUMBER (constrained to {1,2,...})
xs:nonNegativeInteger	NUMBER (constrained to {0,1,2,...})
xs:nonPositiveInteger	NUMBER (constrained to {...,-2,-1,0})
xs:unsignedInt	NUMBER (additional constraints)
xs:short	NUMBER
xs:unsignedShort	NUMBER (additional constraints)
xs:long	NUMBER
xs:unsignedLong	NUMBER (additional constraints)
xs:byte	BLOB
xs:unsignedByte	BLOB (additional constraints)

30.5 Technische Daten

Dieses Kapitel enthält Informationen zu einigen technischen Aspekten Ihrer Software. Es ist in die folgenden Abschnitte gegliedert:

- OS- und Arbeitsspeicheranforderungen 2270
- Altova-Prozessoren 2270
- Unicode-Unterstützung 2271
- Internet-Verwendung 2271

30.5.1 OS- und Arbeitsspeicheranforderungen

Betriebssystem

Die Altova-Software-Applikationen stehen für die folgenden Plattformen zur Verfügung:

- Windows 7 SP1 mit Plattformupdate, Windows 8, Windows 10, Windows 11
- Windows Server 2008 R2 SP1 mit Plattformupdate oder höher

Arbeitsspeicher

Da die Software in C++ geschrieben wurde, wird dafür nicht so viel Platz wie in einer Java Runtime Umgebung benötigt und normalerweise wird dafür weniger Arbeitsspeicher als bei einer vergleichbaren Java-basierten Applikation benötigt. Es ist notwendig, dass die einzelnen Dokumente in den Hauptarbeitsspeicher geladen werden, damit jedes Dokument zur Gänze geparst und analysiert werden kann und die Anzeige- und Bearbeitungsgeschwindigkeit während der normalen Arbeit verbessert wird. Daher steigt mit der Größe des Dokuments auch der Arbeitsspeicherbedarf.

Auch die unbegrenzte Rückgängig-Funktion kann einiges an Arbeitsspeicher in Anspruch nehmen. Wenn Sie große Abschnitte in großen Dokumenten immer wieder ausschneiden und einfügen, kann dies enorm viel Speicherplatz verbrauchen.

30.5.2 Altova-Prozessoren

XML Validator

Wenn Sie ein XML-Dokument öffnen, verwendet die Applikation den integrierten XML Validator, um das Dokument auf Wohlgeformtheit zu prüfen, es anhand eines Schemas zu validieren (falls eines angegeben wurde) und Baumstrukturen und Infosets zu erstellen. Der Altova XML Validator dient auch dazu, beim Editieren von Dokumenten intelligente Eingabehilfen zur Verfügung zu stellen und etwaige Validierungsfehler dynamisch anzuzeigen.

Im integrierten XML Validator ist die Final Recommendation der W3C XML Schema Spezifikationen 1.0 und 1.1 implementiert. Neue Entwicklungen, die von der XML Schema-Arbeitsgruppe empfohlen werden, werden ständig in den XML Validator integriert, sodass Ihnen mit Altova-Produkten immer eine Entwicklungsumgebung auf dem neuesten Stand der Technik zur Verfügung steht.

© 2016-2022 Altova GmbH

XSLT- und XQuery-Prozessor

Die Altova Produkte arbeiten mit dem Altova XSLT 1.0, 2.0 und 3.0-Prozessor und dem Altova XQuery 1.0 und 3.1-Prozessor. Falls einer dieser Prozessoren im Produkt enthalten ist, finden Sie die Dokumentation zurm implementierungsspezifischen Verhalten der einzelnen Prozessoren in den Anhängen zu dieser Dokumentation.

Anmerkung: Altova MapForce verwendet den XSLT 1.0-, 2.0- und XQuery 1.0-Prozessor zur Codegenerierung.

30.5.3 Unicode-Unterstützung

Die XML-Produkte von Altova bieten vollständige Unicode-Unterstützung. Um ein XML-Dokument benötigen Sie eine Schriftart, die die von diesem Dokument verwendeten Unicode-Zeichen unterstützt.

Beachten Sie bitte, dass die meisten Schriftarten nur eine bestimmte Untergruppe des gesamten Unicode-Bereichs enthalten und normalerweise für das entsprechende Schriftsystem ausgelegt sind. Wenn Zeichen falsch dargestellt werden, könnte der Grund darin liegen, dass die gewählte Schriftart die erforderlichen Glyphen nicht enthält. Es ist daher nützlich, eine Schriftart zu verwenden, die den gesamten Unicode-Bereich abdeckt - v.a. wenn Sie XML-Dokumente in unterschiedlichen Sprachen oder Schriftsystemen editieren. Ein typische auf Windows PCs installierte Unicode-Schriftart ist Arial Unicode MS.

Im Ordner`/Examples` Ihres Applikationsordners finden Sie eine XHTML-Datei mit dem Namen `UnicodeUTF-8.html`, die den folgenden Satz in verschiedenen Sprachen und Schriftsystemen enthält:

- *When the world wants to talk, it speaks Unicode*
- *Wenn die Welt miteinander spricht, spricht sie Unicode*
- 世界的に話すなら、Unicode です。)

Wenn Sie diese XHTML-Datei öffnen, erhalten Sie einen kurzen Eindruck davon, was mit Unicode möglich ist und welche Schriftsysteme von den auf Ihrem PC verfügbaren Schriftarten unterstützt werden.

30.5.4 Internet-Verwendung

Altova-Applikationen können für Sie auch eine Verbindung mit dem Internet herstellen. Dies geschieht in den folgenden Fällen:

- Wenn Sie im Registrierungsdialogfeld (**Hilfe | Software-Aktivierung**) auf "Kostenlosen Evaluierungs-Key anfordern" klicken, werden die drei Felder im Registrierungsdialogfeld über eine normale HTTP-Verbindung (Port 80) an unseren Webserver übertragen. Sie erhalten dann per E-Mail (normales SMTP) den kostenlosen Evaluierungs-Keycode zugesandt.
- In einige Altova-Produkten können Sie eine Datei über das Internet öffnen (**Datei | Öffnen | Zu URL wechseln**). In diesem Fall wird das Dokument mittels einer der folgenden Protokollmethoden und Verbindungen aufgerufen: HTTP (normalerweise Port 80), FTP (nomalerweise Port 20/21), HTTPS (normalerweise Port 443). Sie können auch einen HTTP-Server auf Port 8080 verwendden. (Definieren Sie den Port im Dialogfeld "URL", indem Sie ihn durch ein Komma getrennt nach dem Servernamen angeben.*)*

- Wenn Sie ein XML-Dokument öffnen, das sich auf ein XML-Schema oder eine DTD bezieht und das Dokument durch eine URL definiert wird, wird das referenzierte Schema-Dokument ebenfalls über eine HTTP-Verbindung (Port 80) oder ein anderes in der URL definiertes Protokoll (siehe Punkt 2 oben) aufgerufen. Ebenso wird ein Schema-Dokument aufgerufen, wenn eine XML-Datei validiert wird. Beachten Sie, dass die Validierung automatisch beim Öffnen des Dokuments erfolgen kann, wenn Sie dies in der Applikation so konfiguriert haben (Register "Datei" des Dialogfelds "Optionen" **Extras | Optionen**).
- Webservice-Verbindungen werden in Altova Applikationen, die WSDL und SOAP verwenden, mittels WSDL-Dokumenten definiert.
- Wenn Sie den Befehl "**Als Mail senden**..." (**Datei | Als Mail senden**) in XMLSpy verwenden, wird die aktuelle Auswahl bzw. Datei über ein MAPI-kompatibles Mail-Programm, das auf dem PC des Benutzers installiert ist, versendet
- Im Rahmen der Altova-Software-Lizenzvereinbarung beschriebenen Software-Aktivierung und beim Live-Update.

© 2016-2022 Altova GmbH

30.6 Lizenzinformationen

Dieser Anhang enthält die folgenden Informationen:

- Informationen über den Vertrieb dieses Software-Produkts
- Informationen zur Software-Aktivierung und Lizenzüberwachung
- die Lizenzvereinbarung zu diesem Software-Produkt

Lesen Sie die Informationen bitte sorgfältig - sie sind rechtlich bindend, da Sie sich bei der Installation dieses Software-Produkts damit einverstanden erklärt haben.

Den Inhalt aller Altova-Lizenzenvereinbarungen finden Sie auf der Altova Website unter Rechtliches.

30.6.1 Electronic Software Distribution

Dieses Produkt ist über EDS (Electronic Software Distribution), also auf elektronischem Weg erhältlich, eine Methode, die die folgenden einzigartigen Vorteile bietet:

- Sie können die Software kostenlos 30 Tage lang testen, bevor Sie sich zu einem Kauf entscheiden. (*Anmerkung: Die Lizenz für MobileTogether Designer ist kostenlos*.)
- Wenn Sie sich entschieden haben, die Software zu kaufen, können Sie Ihre Bestellung online auf der Altova Website tätigen. Sie erhalten dann innerhalb weniger Minuten ein vollständig lizenziertes Produkt.
- Sie erhalten immer die neueste Version unserer Software
- Die Software enthält ein umfassendes Hilfesystem, das Sie von der Benutzeroberfläche der Appikation aus aufrufen können. Die neueste Version des Benutzerhandbuchs steht auf unserer Website www.altova.com (i) im HTML-Format zum Aufrufen online und (ii) im PDF-Format zum Download und Ausdrucken zur Verfügung.

30-Tage-Evaluierungszeitraum

Nachdem Sie dieses Software-Produkt heruntergeladen haben, können Sie es 30 Tage lang kostenlos testen. Während dieses Zeitraums werden Sie nach etwa 20 Tagen in regelmäßigen Abständen daran erinnert, dass die Software noch nicht lizenziert wurde. Diese Erinnerungsmeldung wird allerdings nur einmal, nämlich bei jedem Start des Programms, angezeigt. Wenn Sie das Programm nach Ablauf des 30-tägigen Evaluierungszeitraums weiterhin verwenden möchten, müssen Sie eine Produktlizenz erwerben, die Sie in Form einer Lizenzdatei mit einem Keycode erhalten. Laden Sie die Lizenzdatei über das Dialogfeld "Software-Aktivierung" Ihres Produkts hoch, um das Produkt freizuschalten.

Sie können Ihre Produktlizenz über https://shop.altova.com/ erwerben.

Weitergabe der Software an andere Mitarbeiter in Ihrem Unternehmen zu Testzwecken

Wenn Sie die Evaluierungsversion der Software auch anderen Personen in Ihrem Unternehmen über das Netzwerk zur Verfügung stellen möchten oder wenn Sie sie auf einem PC installieren möchten, der nicht mit dem Internet verbunden ist, dürfen Sie nur das Installationsprogramm weitergeben, vorausgesetzt es wurde nicht modifiziert. Jeder, der das von Ihnen zur Verfügung gestellte Installationsprogramm aufruft, muss einen eigenen Evaluierungs-Keycode für 30 Tage anfordern. Nach Ablauf des Testzeitraums, muss eine Lizenz erworben werden, damit das Produkt weiter verwendet werden kann.

30.6.2 Software-Aktivierung und Lizenzüberwachung

Im Rahmen der Aktivierung der Software durch Altova, verwendet die Software unter Umständen Ihr internes Netzwerk und Ihre Internetverbindung, um die Lizenzdaten während der Installation, Registrierung, der Verwendung oder der Aktualisierung an einen von Altova betriebenen Lizenzserver zu übertragen und die Authentizität der Lizenzdaten zu überprüfen, damit Altova-Software nicht ohne Lizenz oder auf unzulässige Art und Weise verwendet werden kann und um den Kundenservice gleichzeitig zu verbessern. Bei der Aktivierung werden zwischen Ihrem Computer und dem Altova-Lizenzserver für die Lizenzierung erforderliche Daten wie Informationen über Betriebssystem, IP-Adresse, Datum/Uhrzeit, Software-Version und Computername sowie andere Informationen ausgetauscht.

Ihr Altova-Produkt verfügt über ein integriertes Lizenzüberwachungsmodul, das ebenfalls dazu beiträgt, unbeabsichtigte Verletzungen der Lizenzvereinbarung zu vermeiden. Ihr Produkt kann entweder mit einer Einzelplatzlizenz oder einer Mehrfachlizenz erworben werden. Je nach Lizenz stellt das Lizenzüberwachungsmodul sicher, dass nicht mehr als die lizenzierte Anzahl an Benutzern die Applikation gleichzeitig verwendet.

Bei dieser Lizenzüberwachungsmethode wird Ihr LAN-Netzwerk verwendet, um die Kommunikation zwischen Instanzen der Applikation, die auf verschiedenen Computern laufen, zu überwachen.

Einzelplatzlizenz

Beim Start der Applikation wird im Rahmen der Lizenzüberprüfung ein kurzes Broadcast-Datagramm abgesendet, um andere Instanzen des Produkts, die auf anderen Computern im selben Netzwerk laufen, zu finden. Wenn keine Antwort einlangt, wird ein Port geöffnet, der Informationen von anderen Instanzen der Applikation empfangen kann.

Mehrplatzlizenz

Wenn Sie im selben LAN mehrere Instanzen der Applikation verwenden, kommunizieren diese beim Start kurz miteinander, um Keycode-Informationen auszutauschen, damit Sie sicher sein können, dass nicht mehr als die lizenzierte Anzahl an Lizenzen gleichzeitig in Verwendung ist. Dieselbe Lizenzüberwachungstechnologie wird auch bei Unix und vielen anderen Datenbankentwicklungstools verwendet. Sie gestattet Benutzern den Erwerb von Parallellizenzen für mehrere Benutzer zu vernünftigen Preisen.

Wir sind außerdem bestrebt, nur wenige, kleine Netzwerkpakete zu versenden, um Ihr Netzwerk nicht zu überlasten. Die von Ihrem Altova Produkt verwendeten TCP/IP Ports (2799) sind offiziell bei IANA registriert, (*nähere Informationen siehe IANA Service Name Registry*) und unser Lizenzüberwachungsmodul basiert auf einer bewährten und erprobten Technologie.

Wenn Sie eine Firewall verwenden, werden Sie unter Umständen feststellen, dass die Computer, auf denen Altova-Produkte laufen, über Port 2799 miteinander kommunizieren. Sie können diesen Netzwerkverkehr

© 2016-2022 Altova GmbH

zwischen verschiedenen Gruppen in Ihrem Unternehmen natürlich blockieren, solange Sie mit anderen Mitteln sicherstellen können, dass Ihre Lizenzvereinbarung eingehalten wird.

Anmerkung zu Zertifikaten

Ihre Altova Applikation kontaktiert den Altova Lizenzierungsserver über HTTPS (`link.altova.com`). Für diese Kommunikation verwendet Altova ein registriertes SSL-Zertifikat. Wenn dieses Zertifikat ersetzt wird (z.B. von Ihrer IT-Abteilung oder einer externen Agentur), werden Sie von Ihrer Altova Applikation gewarnt, dass die Verbindung nicht sicher ist. Sie könnten Ihre Altova Applikation mit dem Ersetzungszertifikat starten. Dies würde jedoch auf Ihr eigenes Risiko geschehen. Wenn Sie eine Warnung sehen, dass die *Verbindung nicht sicher* ist, überprüfen Sie den Ursprung des Zertifikats und wenden Sie sich an Ihr IT-Team (die in der Lage sein sollten, zu entscheiden, ob das Abfangen und die Ersetzung des Altova-Zertifkats fortgesetzt werden soll).

Wenn Ihr Unternehmen sein eigenes Zertifikat verwenden muss (z.B. um die Kommunikation zu und von Client-Rechnern zu überwachen), empfehlen wir Ihnen, Altova LicenseServer, die kostenlose Lizenzverwaltungssoftware von Altova in Ihrem Netzwerk zu installieren. Client-Rechner verwenden mit dieser Konfiguration weiterhin die Zertifikate Ihres Unternehmens, während der Altova LicenseServer für die Kommunikation mit Altova das Altova-Zertifikat das Altova-Zertifikat verwenden kann.

30.6.3 Altova Endbenutzer-Lizenzvereinbarung

- Die Altova-Endbenutzer-Lizenzvereinbarung kann unter https://www.altova.com/de/legal/eula eingesehen werden.
- Die Altova-Datenschutzbestimmungen finden Sie unter https://www.altova.com/de/privacy.

Index

$

$ref (JSON-Schemas), 733

.

.docx, 284, 955
.NET,
Integration von XMLSPY mit, 1141
und XSLT Debugger, 1146
Unterschiede zur XMLSPY Standalone Version, 1143
.NET Erweiterungsfunktionen,
Datentypkonvertierungen, .NET in XPath/XQuery, 2252
Datentypkonvertierungen, XPath/XQuery in .NET, 2251
für XSLT und XQuery, 2246
Instanzmethoden, Instanzfelder, 2250
Konstruktoren, 2249
statische Methoden, statische Felder, 2249
Übersicht, 2246
.pptx, 284, 955
.xslx, 284, 955

A

AAIDC-Bereich, 212
Abfrage,
siehe Datenbank abfragen, 1440
siehe Datenbankabfrage (Fenster(, 1440
siehe XQuery, 1440
Abfragefenster,
im Fenster Datenbankabfrage, 1449
Abfragen,
für DB-Anzeige in der Authentic-Ansicht, 651
Abgeleitete Types,
Base Type modifizieren, 243
Ableiten eines Schematyps, 244
ActiveX,
Integration auf Applikationsebene, 2076
Integration auf Dokumentebene, 2078
Voraussetzungen für die Integration, 2073
ActiveX Controls,
Unterstützung, 1691
zur Visual Studio Toolbox hinzufügen, 2075
ADO,
als Datenverbindungsschnittstelle, 969
Verbindung einrichten, 975
ADO.NET,
Verbindung einrichten, 982
Aktive Konfiguration,
für globale Ressourcen, 1579
Aktuelle Execution Nodes einblenden,
XSLT Debugger, 1424
Alias,
siehe Globale Ressourcen, 1061
Alles markieren,
Befehl, 1306
Als Bild speichern, 1305
Als reinen Text bearbeiten,
in Grid-Ansicht, 1350
Altova Globale Ressourcen,
siehe Globale Ressourcen, 1061
Altova Support, 138
Altova XML Parser,
Info, 2270
Altova-Erweiterungen,
Diagrammfunktionen(siehe Diagrammfunktionen), 2144
Altova-Produkte, 138
Änderungen verfolgen, 1603
Angedockte Fenster, 115
Anhängen,
Zeile (in Authentic-Ansicht), 1437
Anhängen (Befehl),
in Grid-Ansicht, 1349
Annotationen in der Schema-Ansicht, 182
Anpassen, 1593
Kontextmenü, 1593
Kontextmenü 'Anpassen', 1598
Makros, 1595
Menü, 1593
Symbolleiste/Menü Befehle, 1584
Anpassung, 135
Ansicht,
ausklappen, 1504
Browser-Ansicht, 1503
einklappen, 1504
Enhanced Grid-Ansicht, 1502
erweitern, 1504

© 2016-2022 Altova GmbH

Ansicht,
Gehe zu Datei, 1506
Gehe zu Zeile/Zeichen, 1505
Menü, 1501
Optimale Breite, 1505
Schema Design-Ansicht, 1502
Text-Ansicht, 1501
Ansicht konfigurieren,
Dialogfeld in der Content Model-Ansicht, 1395
Ansicht wechseln,
zur Authentic-Ansicht, 624
Ant,
Umgebungsvariablen definieren für, 1253
Anzeigen,
reduzieren, 1504
API, 2034
Dokumentation, 1706
JAVA, 2034
JAVA Classpath, 2034
Application,
ActiveDocument, 1749
AddMacroMenuItem, 1750
AddXSLT_XQParameter, 1750
Application, 1750
ClearMacroMenu, 1751
CurrentProject, 1751
Dialogs, 1752
Documents, 1752
GetDatabaseImportElementList, 1753
GetDatabaseSettings, 1753
GetDatabaseTables, 1754
GetExportSettings, 1754
GetTextImportElementList, 1755
GetTextImportExportSettings, 1756
GetXSLT_XQParameterCount, 1756
GetXSLT_XQParameterName, 1756
GetXSLT_XQParameterXPath, 1756
ImportFromDatabase, 1757
ImportFromSchema, 1758
ImportFromText, 1758
ImportFromWord, 1759
NewProject, 1760
OnBeforeOpenDocument, 1747
OnBeforeOpenProject, 1748
OnDocumentOpened, 1748
OnProjectOpened, 1749
OpenProject, 1761
Parent, 1761
Quit, 1761
ReloadSettings, 1762
RemoveXSLT_XQParameter, 1762
RunMacro, 1762
ScriptingEnvironment, 1763
ShowApplication, 1763
ShowForm, 1764
URLDelete, 1764
URLMakeDirectory, 1765
WarningNumber, 1765
WarningText, 1766
Arbeitsspeicher-Anforderungen, 2270
Archivansicht, 955
Archivansicht-Ansicht, 284
und EPUB-Dateien, 963
und OOXML-Dateien, 957
und ZIP-Dateien, 961
Arcroles in XBRL, 1533
Assertion-Meldungen, 241
Assertions in der Schema-Ansicht, 212, 217
Assistent für neue Taxonomien, 851
ATL,
Plug-In Beispieldateien, 1694
Attribute, 73, 182
in der Content Model-Ansicht ein- und ausblenden, 73
in Schemadefinitionen, 73
Attribute in der Schema-Ansicht, 212, 213
AttributeFormDefault,
Einstellungen in der Schema Design-Ansicht, 1386
Attribut-Eingabehilfe,
in der Authentic-Ansicht, 630
Attributgruppen in der Schema-Ansicht, 213
Attribut-Vorschau, 1617
Attributwerte,
in der Authentic-Ansicht eingeben, 618
Ausdruckserstellung, 598
Ausgabefenster,
Anzeige, 1650
Ausgabeformatierung, 1603
Ausnahmen wegen zu wenig Arbeitsspeicher,
auflösen, 1253
Ausschneiden (Befehl), 1298
Aussteigen,
XSLT Debugger, 1424
Auswahl des Designs für XMLSpy in Eclipse, 1153
Authentic,
CDATA-Abschnitte, 616
Daten eingeben, 616

© 2016-2022 Altova GmbH

Authentic,
Daten in einer XML-Datenbank bearbeiten, 1431
Dateneingabe-Steuerelemente, 616
Eingabehilfen, 610
Elemente anwenden, 613
Elemente löschen, 613
Entities, 616
Kontextmenü, 610
Markup-Tags anzeigen, 610
Menü, 1428
Nodes einfügen, 613
Nodes entfernen, 613
Nodes hinzufügen, 613
Sonderzeichen, 616
Tabellen, 613
Authentic (Menü),
Bearbeitung dynamischer Tabellen, 624
Markup-Anzeige, 624
Authentic Desktop,
Features, 138
Hilfe, 138
Authentic Skripterstellung,
Sicherheitseinstellungen, 1438
vertrauenswürdige Pfade, 1438
Authentic XML, 606
Authentic-Ansicht, 644
als XML/Text einfügen, 634
Attributwerte eingeben, 618
Ausgabedokumente anhand der PXF-Datei generieren, 1438
Bearbeiten von XML, 335
Benutzeroberfläche, 622
DB-Daten bearbeiten, 1429
Dokumentanzeige, 627
Eingabehilfen, 630
Entities einfügen, 619
Hauptfenster, 627
Kontextmenüs, 634
Markup-Anzeige, 627
Markup-Anzeige in, 624
neue XML-Datei öffnen, 1429
SPS-Tabellen, 642
Symbolleisten-Schaltflächen, 624
Tabellen (SPS und XML), 642
Text formatieren in, 624
Übersicht über die GUI, 622
Verwendung von XML-Tabellen, 644
Verwendung wichtiger Features, 637
wechseln zur, 1503
XML-Dokument drucken, 620
XML-Dokument öffnen in, 609
XML-Tabellen, 644
XML-Tabellensymbole, 648
Authentic-Ansicht-Vorlage, 609
AuthenticDataTransfer,
dropEffect, 1768
getData, 1769
ownDrag, 1769
type, 1769
AuthenticRange,
AppendRow, 1775
Application, 1775
CanPerformAction, 1776
CanPerformActionWith, 1776
Close, 1777
CollapsToBegin, 1777
CollapsToEnd, 1777
Copy, 1777
Cut, 1778
Delete, 1778
DeleteRow, 1778
DuplicateRow, 1779
ExpandTo, 1780
FirstTextPosition, 1780
FirstXMLData, 1781
FirstXMLDataOffset, 1782
GetElementAttributeNames, 1783
GetElementAttributeValue, 1784
GetElementHierarchy, 1784
GetEntityNames, 1785
Goto, 1786
GotoNext, 1786
GotoNextCursorPosition, 1787
GotoPrevious, 1787
GotoPreviousCursorPosition, 1788
HasElementAttribute, 1788
InsertEntity, 1789
InsertRow, 1789
IsCopyEnabled, 1790
IsCutEnabled, 1790
IsDeleteEnabled, 1790
IsEmpty, 1791
IsEqual, 1791
IsFirstRow, 1791
IsInDynamicTable, 1791
IsLastRow, 1792
IsPasteEnabled, 1792

© 2016-2022 Altova GmbH

AuthenticRange,
IsTextStateApplied, 1793
LastTextPosition, 1793
LastXMLData, 1794
LastXMLDataOffset, 1794
MoveBegin, 1796
MoveEnd, 1796
MoveRowDown, 1797
MoveRowUp, 1796
Parent, 1797
Paste, 1797
PerformAction, 1798
Select, 1799
SelectNext, 1799
SelectPrevious, 1800
SetElementAttributeValue, 1801
SetFromRange, 1802
Text, 1802
AuthenticView, 1821
Application, 1813
AsXMLString, 1813
DocumentBegin, 1815
DocumentEnd, 1815
Event, 1816
Goto, 1817
IsRedoEnabled, 1818
IsUndoEnabled, 1818
MarkupVisibility, 1819
OnBeforeCopy, 1804
OnBeforeCut, 1805
OnBeforeDelete, 1805
OnBeforeDrop, 1805
OnBeforePaste, 1807
OnDragOver, 1807
OnKeyboardEvent, 1809
OnMouseEvent, 1810
OnSelectionChanged, 1810
Parent, 1819
Print, 1819
Redo, 1820
Selection, 1820
Undo, 1821
WholeDocument, 1822
XMLDataRoot, 1822
Auto-Komplettierung,
Text-Ansicht Ein/Aus, 1609
Autokomplettierung in SQL-Scripts, 1449
Automatische Validierung, 1606
Automatische Validierung ausschalten, 1606
Automatisches Backup, 142
Automatisches Backup (Einstellungen), 1603
Avro,
Dateitypen, 779
Datenstrukturen in Binärdatei, 786
Datenstrukturen in JSON, 785
Übersicht, 779
und RaptorXML, 779
Avro-Ansicht, 786
Avro-Binärdateien, 786
Avro-Schema, 676
Beschreibung, 782
Terminologie, 782

B

Backups, 142
Balkendiagrammfunktionen, 389
Base Type,
modifizieren, 243
Bearbeiten,
Makro Schaltfläche, 1598
Menü, 1297
Bearbeiten in der Textansicht, 151
Bearbeitet mit XMLSpy, 1603
Bearbeitungsansichten, 140
Beenden,
Debuggersitzung, 1423
Beendungsmodus, 1603
Befehl, 1593
hinzufügen zu Symbolleiste/Menü, 1584
Kontextmenü, 1593
löschen aus Menü, 1593
Menü zurücksetzen, 1593
Befehle,
Auflistung in Tastaturzuordnung, 1653
Befehle der Tabellenansicht (JSON-Grid-Ansicht), 1369
Befehle der Tabellenansicht (XML-Grid-Ansicht), 1350
Bei Bearbeitung validieren (Befehl), 1359
Beispieldateien,
Tutorial, 36
Beispiele,
Pfad installierter Dateien, 138
Beispielwerte von simpleTypes, 235
Benutzerdefiniertes Wörterbuch, 1558

© 2016-2022 Altova GmbH

Benutzer-DSN,
einrichten, 989
Benutzerhandbuch, 28
Benutzeroberfläche,
mittels Plug-in konfigurieren, 1691
Benutzeroberfläche - Beschreibung, 115
Benutzerreferenz, 1275
Beschreibung der Benutzeroberfläche, 115
Bestellen der Altova-Software, 1654
Betriebssysteme für Altova-Produkte, 2270
Beziehungen in Taxonomien, 865, 866
Beziehungen in Taxonomies, 869
Bibliothek, 1270
Big-endian, 1608
Bild in der Grid-Ansicht kopieren, 1300
Bilder in der Grid-Ansicht, 322
Bilder in der JSON-Grid-Ansicht, 715
Breakpoint,
Dialogfeld, 1426
Breakpoint Aus - XSLT Debugger, 1425
Breakpoint Ein - XSLT Debugger, 1425
Breakpoints,
im XSLT/XQuery Debugger, 557
Verwendung in SOAP Debugger, 814
Breakpoints (Dialogfeld), 543
Breakpoints im XPath/XQuery Debugger, 591
Browser, 1617
Ansicht, 1503
Browser-Ansicht,
Inhalt aktualisieren, 1508
Laden einer Seite abbrechen, 1508
Schriftgröße, 1508
separate Fenster, 1508
zurück und vorwärts bewegen, 1508
Browser-Fenster,
im Fenster "Datenbankabfrage", 1445
BSON in MongoDB, 754
BSON-Schemabearbeitung, 754
build.xml,
zip64mode aktivieren in, 1253

C

C#,
Code, 1156
Codegenerierungseinstellungen, 1254
Integration von XMLSpy, 2081
Referenz zu generierten Klassen, 1221
C++,
Code, 1156
Codegenerierungseinstellungen, 1254
Referenz zu generierten Klassen, 1205
Call Stack (XPath/XQuery), 591
Call Stack-Fenster,
in XSLT/XQuery Debugger, 553
Callgraph-Profilerstellung, 567
CDATA-Abschnitte,
in der Authentic-Ansicht einfügen, 637
Child hinzufügen (Befehl),
in Grid-Ansicht, 1349
Class,
JAVA, 2034
ClassPath statement, 2034
Code,
integrierte Typen, 1269
SPL, 1256
Code Generator, 1156, 1385
CodeGeneratorDlg,
Application, 1823
CPPSettings_DOMType, 1824
CPPSettings_LibraryType, 1825
CPPSettings_UseMFC, 1825
CSharpSettings_ProjectType, 1826
OutputPath, 1826
OutputPathDialogAction, 1826
OutputResultDialogAction, 1827
Parent, 1827
ProgrammingLanguage, 1827
PropertySheetDialogAction, 1828
TemplateFileName, 1828
Codepage, 1626
COM API,
im Skript-Editor, 1673
COM-API,
Dokumentation, 1706
ComplexType,
Definition erweitern, 63
ComplexTypes,
anonymous, 182
global, 182
in Schema-Definitionen, 63
named, 182
Conditional Type Assignments, 205
Conditional Type-Alternativen, 205

© 2016-2022 Altova GmbH

Content Model,
Attribute ein- und ausblenden, 73
Diagramm speichern, 1389
einfaches Content Model erstellen, 51
Content Model-Ansicht, 48, 190
bearbeiten in, 199
Beschreibung der Benutzeroberfläche, 192
Conditional Types zuweisen, 205
Diagrammobjekte, 192
Kompositoren und Komponenten, 192
konfigurieren, 1395
Content Models,
von Schemakomponenten, 190
Copyright-Informationen, 2273
CoreCatalog.xml, 464
CR&LF, 1603
CSS, 667
Autokomplettierung, 670
Dokumentübersicht, 670
Eigenschaften, 670
Info-Fenster, 670
Syntaxfärbung, 670
CSS-Info-Fenster, 670
CSV-Datei,
als XML importieren, 1470
CustomCatalog, 1353
CustomCatalog.xml, 464

D

Darstellung, 1617
DatabaseConnection,
ADOConnection, 1829
AsAttributes, 1830
CreateMissingTables, 1830
CreateNew, 1830
DatabaseKind, 1831
ExcludeKeys, 1831
File, 1831
IncludeEmptyElements, 1832
NumberDateTimeFormat, 1833
ODBCConnection, 1833
SQLSelect, 1834
TextFieldLen, 1834
DatabaseSpy,
3-D Diagramme, 400
Balkendiagrammfunktionen, 389
Diagramm x-Achse, 395
Diagramm y-Achse, 397
Diagramm z-Achse, 399
Diagrammfarben, 393
Diagrammgrößen, 400
Diagrammhintergrund, 386
Diagrammraster, 395, 397, 399
Diagrammschriften, 402
Diagrammtitel, 386
Einzelwertdiagrammfunktionen, 389
Flächendiagrammfunktionen, 389
Kerzendiagrammfunktionen, 389
Kreisdiagrammfunktionen, 389
Legende, 386
Liniendiagrammfunktionen, 389
Optionen für Diagrammschriften, 402
Datei, 1603
als Mail senden, 1292
Druckoptionen, 1293
Kodierung, 1285
neue erstellen, 1276
Öffnen, 1280
Optionen beim Öffnen, 1603
Register, 1603
schließen, 1286
speichern, 1286
Standard-Encoding, 1608
Datei (Menü), 1276
Dateiarten, 1606
Datei-DSN,
einrichten, 989
Dateien,
Vergleichsoptionen, 1575
Zu Versionskontrolle hinzufügen, 1327
zuletzt verwendete, 1296
zwei vergleichen, 1567
Dateien speichern,
Kodierung, 1285
Dateien vergleichen, 1567
Optionen, 1575
Dateierweiterungen,
anpassen, 1353
für XQuery-Dateien, 517
Dateipfade,
in XML-Dokument einfügen, 293
Dateivergleiche, 1111, 1112
Datenbank,

© 2016-2022 Altova GmbH

Datenbank,
Datensätze bearbeiten, 1454
DB auf Basis von Schema erzeugen, 1484
Oracle XML DB, 1465
Struktur als XML-Schema importieren, 1478
XML-Daten exportieren, 1490
Datenbank abfragen (Befehl), 1440
Datenbank/Tabellen-Ansicht,
Verwendung, 99
Datenbankabfrage,
Abfrage erstellen, 1453
Browser-Fenster im Fenster Datenbankabfrage, 1445
Ergebnisse, 1454
Ergebnisse bearbeiten, 1454
Meldungsregister, 1454
Verbindung zur DB für Abfrage herstellen, 1442
Datenbankabfrage (Fenster), 1440
Datenbanken,
in Authentic-Ansicht bearbeiten, 1429
siehe auch DB, 650
und globale Ressourcen, 1077
Unterstützung in XMLSpy, 1059
Datenbanken in XMLSpy, 967
Datenbanktreiber,
Übersicht, 972
Datenbankverbindung,
Assistenten starten, 970
aus globalen Ressourcen wiederverwenden, 1002
Beispiele einrichten, 1003
einrichten, 969
Datensätze bearbeiten, 1454
Datumswahl,
in der Authentic-Ansicht verwenden, 657
Datumswerte,
manuell ändern, 658
DB, 650, 651
Abfragen erstellen, 651
Abfragen in der Authentic-Ansicht, 650
Anzeige in der Authentic-Ansicht filtern, 651
Bearbeiten in der Authentic-Ansicht, 650
in der Authentic-Ansicht bearbeiten, 655
Navigieren in Tabellen in der Authentic-Ansicht, 650
Parameters in DB-Abfragen, 651
DB XML,
XML Schemas zuweisen für IBM DB2, 1460
XML-Schemas verwalten, für IBM DB2, 1457
db2-fn:sqlquery, 535
db2-fn:xmlcolumn, 535
Deaktivieren,
Tracepoint - XSLT Debugger, 1425
Debuggen von XPath/XQuery-Ausdrücken, 591
Debugger,
aktuelle Execution Nodes einblenden, 1424
aussteigen, 1424
Breakpoint Ein/Aus, 1425
Breakpoint einfügen/löschen, 1424
Breakpoints/Tracepoints (Dialogfeld), 1426
Debug-Fenster, 1427
einsteigen, 1423
Einstellungen, 1427
für SOAP, 802
Sitzung beenden, 1423
SOAP Debugger, 1529
Tracepoint aktivieren/deaktivieren, 1425
Tracepoint einfügen/entfernen, 1425
überspringen, 1424
XSLT Debugger erneut starten, 1423
XSLT Debugger starten, 1422
XSLT Debugger stoppen, 1423
Debugger -Fenster,
Anordnung, 555
Debug-Punkte im XPath/XQuery Debugger, 591
Default Open Content Models, 209
defaultOpenContent, 182
Definieren,
Diagrammschriften, 402
Diagrammtitel, 386
Rasterlinien, 395, 397, 399
Schriftarten in Diagrammen, 402
x-Achseneinstellungen, 395
y-Achseneinstellungen, 397
z-Achseneinstellungen, 399
Definition,
3-D Einstellungen, 400
Balkendiagrammfunktionen, 389
Diagrammgrößen, 400
Einzelwertdiagrammfunktionen, 389
Flächendiagrammfunktionen, 389
Kerzendiagrammfunktionen, 389
Kreisdiagrammfunktionen, 389
Liniendiagrammfunktionen, 389
Definitionsübersichtsraster, 726
Deinieren,
Diagrammfarben, 393
Farben von Diagrammen, 393
Delete,

© 2016-2022 Altova GmbH

Delete,
Application.URLDelete, 1764
Details-Eingabehilfe, 51
Diagramm anzeigen, 1398
Diagrammdatentabelle,
Anleitung zur Erstellung, 353
Diagramme,
3-D Einstellungen, 400
Abstände, 400
Auswahl für die X-Achse, 361
Auswahl für die Y-Achse, 366
Balkendiagrammfunktionen, 389
Beispiel (einfach), 404
Beispiel (Kerzendiagramm), 413
Beispiel (komplex), 406
Darstellung, 376
Daten, 370
Diagrammeinstellungen, 373
Einzelwertdiagrammfunktionen, 389
Entfernen von Legenden, 386
exportieren, 403
Farbbereich, 393
Farbe von Datenreihen, 393
Farben definieren, 393
Farbschema, 393
Flächendiagrammfunktionen, 389
Größen, 400
Hintergrundfarben, 386
Hinzufügen von Legenden, 386
im XSLT/XQuery Profiler, 579
Kerzendiagrammfunktionen, 389
Kreisdiagrammfunktionen, 389
Liniendiagrammfunktionen, 389
mehrere Diagrammregister, 350
neu laden, 350
Overlays, 372
Quell-XPath, 357
Rasterlinien, 395, 397, 399
Schriftarten, 402
Skalenstriche, 400
Überschrift, 386
Übersicht, 350
x-Achse, 395
y-Achse, 397
z-Achse, 399
Diagramme (Fenster), 130
Diagramme in der Grid-Ansicht, 323
Diagramme in der JSON-Grid-Ansicht, 716
Diagrammfunktionen,
Beispiel, 2233
Diagrammdatenstruktur, 2228
Liste, 2223
Dialogs,
Application, 1835
CodeGeneratorDlg, 1835
DTDSchemaGeneratorDlg, 1837
FileSelectionDlg, 1836
GenerateSampleXMLDlg, 1837
Parent, 1836
SchemaDocumentationDlg, 1836
DocEditEvent (obsolete),
altKey (obsolete), 1982
altLeft (obsolete), 1983
button (obsolete), 1984
cancelBubble (obsolete), 1985
clientX (obsolete), 1986
clientY (obsolete), 1986
ctrlKey (obsolete), 1987
ctrlLeft (obsolete), 1988
dataTransfer (obsolete), 1989
fromElement (obsolete), 1990
keyCode (obsolete), 1990
propertyName (obsolete), 1991
repeat (obsolete), 1991
returnValue (obsolete), 1991
shiftKey (obsolete), 1991
shiftLeft (obsolete), 1992
srcElement (obsolete), 1993
type (obsolete), 1994
DocEditView (obsolete),
ApplyTextState (obsolete), 1998
CurrentSelection (obsolete), 1999
EditClear (obsolete), 1999
EditCopy (obsolete), 2000
EditCut (obsolete), 2000
EditPaste (obsolete), 2001
EditRedo (obsolete), 2001
EditSelectAll (obsolete), 2001
EditUndo (obsolete), 2002
event (obsolete), 2002
GetAllowedElements (obsolete), 2003
GetNextVisible (obsolete), 2005
GetPreviousVisible (obsolete), 2005
IsEditClearEnabled (obsolete), 2006
IsEditCopyEnabled (obsolete), 2006
IsEditCutEnabled (obsolete), 2007

© 2016-2022 Altova GmbH

DocEditView (obsolete),
IsEditPasteEnabled (obsolete), 2008
IsEditRedoEnabled (obsolete), 2008
IsEditUndoEnabled (obsolete), 2009
IsRowAppendEnabled (obsolete), 2009
IsRowDeleteEnabled (obsolete), 2010
IsRowDuplicateEnabled (obsolete), 2010
IsRowInsertEnabled (obsolete), 2011
IsRowMoveDOwnEnabled (obsolete), 2011
IsRowMoveUpEnabled (obsolete), 2012
IsTextStateApplied (obsolete), 2012
IsTextStateEnabled (obsolete), 2013
LoadXML (obsolete), 2013
RowAppend (obsolete), 2014
RowDelete (obsolete), 2015
RowDuplicate (obsolete), 2015
RowInsert (obsolete), 2016
RowMoveDown (obsolete), 2016
RowMoveUp (obsolete), 2017
SaveXML (obsolete), 2017
SelectionMoveTabOrder (obsolete), 2018
SelectionSet (obsolete), 2018
XMLRoot (obsolete), 2019
Document, 1851
Application, 1843
AssignDTD, 1843
AssignSchema, 1844
AssignXSL, 1844
AssignXSLFO, 1844
AuthenticView, 1845
Close, 1846
ConvertDTDOrSchema, 1846
CreateChild, 1848
CreateSchemaDiagram, 1848
CurrentViewMode, 1849
DataRoot, 1849
DocEditView, 1849
Encoding, 1850
EndChanges, 1850
ExecuteXQuery, 1851
ExportToDatabase, 1851
ExportToText, 1852
FullName, 1854
GenerateDTDOrSchema, 1854, 1855
GenerateProgramCode, 1855
GenerateSampleXML, 1855
GenerateSchemaDocumentation, 1856
GetExportElementList, 1858
GetPathName, 1858
GridView, 1859
IsModified, 1859
IsValid, 1859
IsWellFormed, 1862
Name, 1862
OnBeforeCloseDocument, 1841
OnBeforeSaveDocument, 1840
OnBeforeValidate, 1842
OnCloseDocument, 1842
OnViewActivation, 1843
Path, 1863
RootElement, 1863
Save, 1863
SaveAs, 1864
Saved, 1864
SaveInString, 1864
SaveToURL, 1865
SetActiveDocument, 1865
SetEncoding, 1865
SetExternalIsValid, 1866
SetPathName, 1866
StartChanges, 1867
SwitchViewMode, 1867
Title, 1868
TransformXSL, 1868
TransformXSLFO, 1868
UpdateViews, 1869
UpdateXMLData, 1869
XQuery, 1851
Documents,
Count, 1871
Item, 1871
NewAuthenticFile, 1871
NewFile, 1871
NewFileFromText, 1872
OpenAuthenticFile, 1872
OpenFile, 1872
OpenURL, 1873
OpenURLDialog, 1873
Dokument,
Oracle XML DB durchsuchen, 1468
Rechtschreibprüfung, 1558
Dokumentation,
von WSDL-Dateien, 1517
von XBRL-Taxonomien, 1541
von XML-Schema, 1389
Dokumente im Hauptfenster, 116

© 2016-2022 Altova GmbH

Dokumentebene,
Beispiele für die Integration von XMLSpy, 2081
Dokumentieren,
Schema, 78
Drag & Drop,
Elemente hinzufügen, 56
Drag-and-Drop in der JSON Grid-Ansicht, 702
Drag-and-Drop in der XML Grid-Ansicht, 309
Drucken,
von der Authentic-Ansicht, 620
Druckereinrichtung, 1295
Druckoptionen, 1293
DTD,
Code generieren von, 1385
Entities inkludieren, 1372
Entwurf anhand von XML-Datei erzeugen, 1381
Gehe von XML-Dokument zu Definition, 1373
Gehe von XML-Dokument zu DTD, 1373
generieren anhand von XML-Schema (Enterprise und Professional Edition), 451
in UML konvertieren, 1379
Konvertieren in XML-Schema, 1376
Menübefehle zu, 1371
von XML-Dokument generieren, 1374
zu XML-Dokument zuweisen, 1371
DTD/Schema,
Menü, 1371
DTDs, 432, 1603, 1606
in der Grid-Ansicht bearbeiten (Enterprise und Professional Edition), 448
in Textansicht bearbeiten, 448
in XML-Schemas konvertieren (Enterprise und Professional Edition), 448
XML-Dokumentation generieren anhand von, 448
DTDs und Kataloge, 463
DTDSchemaGeneratorDlg,
Application, 1874
AttributeTypeDefinition, 1875
DTDSchemaFormat, 1875
FrequentElements, 1875
GlobalAttributes, 1876
MaxEnumLength, 1876
MergeAllEqualNamed, 1876
OnlyStringEnums, 1876
OutputPath, 1877
OutputPathDialogAction, 1877
Parent, 1877
ResolveEntities, 1877
TypeDetection, 1878
ValueList, 1878
Durchsuchen,
Oracle XML DB, 1468
Dynamische (SPS) Tabellen in der Authentic-Ansicht,
Verwendung, 642
Dynamische Tabellen,
bearbeiten, 624

E

Eclipse Plattform,
XMLSpy Debugger-Perspektive, 1155
Eclipse-Plattform,
und das XMLSpy-Integrationspaket, 1148
und XMLSPY, 1147
XMLSpy-Einstiegspunkte, 1153
XMLSpy-Perspektive in, 1150
EDGAR validieren, 1554
EDGAR-Validierung am Server, 1554
Ein- und auskommentieren,
in XML-Dokumenten in der Textansicht, 293
Einfügen,
als Text, 637
als XML, 637
Breakpoint - XSLT Debugger, 1424
Tracepoint - XSLT debugger, 1425
Zeile (in Authentic-Ansicht), 1437
Einfügen (Befehl), 1298
Einfügen als,
Text, 634
XML, 634
Einfügen nach/vor (Befehl),
in Grid-Ansicht, 1349
Eingabehilfe,
Details, 51
Eingabehilfe "Details", 233
Eingabehilfe "Komponenten", 229, 235
Eingabehilfen, 121
aktualisieren, 1359
Anzeige, 1650
für XQuery, 518
in Grid-Ansicht, 97
Eingabehilfen (Textansicht, Authentic-Ansicht), 337
Eingabehilfen aktualisieren (Befehl), 1359
Eingabehilfen in der Schema-Ansicht, 229

© 2016-2022 Altova GmbH

Eingabehilfen in der Textansicht, 165
Eingabetaste,
drücken, 663
Einrücklinien, 1506
Einrückung,
in der Textansicht, 1306
Einrückung in der Textansicht, 145
Einsteigen,
XSLT Debugger, 1423
Einstellungen, 135, 1602
3-D Diagramme, 400
Balkendiagrammfunktionen, 389
Diagramm x-Achse, 395
Diagramm y-Achse, 397
Diagramm z-Achse, 399
Diagrammfarben, 393
Diagrammgrößen, 400
Diagrammhintergrund, 386
Diagrammraster, 395, 397, 399
Diagrammschriften, 402
Diagrammtitel, 386
Einzelwertdiagrammfunktionen, 389
Flächendiagrammfunktionen, 389
Kerzendiagrammfunktionen, 389
Kreisdiagrammfunktionen, 389
Legende, 386
Liniendiagrammfunktionen, 389
Skripting, 1645
Übersicht, 46
XSLT Debugger, 1427
Einstellungen für den Dateivergleich, 1575
Einstellungen für Textansicht (Dialogfeld), 1506
Einzelwertdiagrammfunktionen, 389
Elemente, 59
Inhalt einschränken, 59
optional machen, 59
Element-Eingabehilfe,
in der Authentic-Ansicht, 630
ElementFormDefault,
Einstellungen in der Schema Design-Ansicht, 1386
ElementList,
Count, 1879
Item, 1879
RemoveElement, 1879
ElementListItem,
ElementKind, 1879
FieldCount, 1880
Name, 1880
RecordCount, 1880
Elementtyp,
in XML-Dokument festlegen, 85
E-Mail,
Dateien senden, 1292
Encoding,
Standard, 1608
Endbenutzer-Lizenzvereinbarung, 2273, 2275
Enhanced Grid-Ansicht, 1502
siehe Grid-Ansicht, 88
Entfernen,
Tracepoint - XSLT debugger, 1425
Entities,
in der Authentic-Ansicht definieren, 637, 659
in der Authentic-Ansicht einfügen, 637
in der Authentic-Ansicht Entities einfügen, 619
in XML Schema-basierten XML-Dokumenten, 291
Entities-Eingabehilfe,
in der Authentic-Ansicht, 630
Enumerationen,
für Attribute definieren, 73
in XMLSpyControl, 2135
Enumerations,
SPYAttributeTypeDefinition, 2020
SPYAuthenticActions, 2020
SPYAuthenticDocumentPosition, 2021
SpyAuthenticElementActions, 2021
SPYAuthenticElementKind, 2021
SPYAuthenticMarkupVisibility, 2022
SPYDatabaseKind, 2022
SPYDialogAction, 2023
SPYDOMType, 2023
SPYDTDSchemaFormat, 2023
SPYEncodingByteOrder, 2023
SPYExportNamespace, 2024
SPYFrequentElements, 2024
SPYKeyEvent, 2025
SPYLibType, 2025
SPYLoading, 2025
SPYMouseEvent, 2026
SPYNumberDateTimeFormat, 2026
SPYProgrammingLanguage, 2027
SPYProjectItemTypes, 2027
SPYProjectType, 2027
SPYSampleXMLGenerationOptimization, 2028
SPYSampleXMLGenerationSchemaOrDTDAssignment, 2029
SPYSchemaDefKind, 2029

© 2016-2022 Altova GmbH

Enumerations,
SPYSchemaDocumentationFormat, 2030
SPYTextDelimiters, 2030
SPYTextEnclosing, 2031
SPYTypeDetection, 2031
SPYURLTapes, 2031
SPYViewModes, 2032
SPYVirtualKeyMask, 2033
SPYXMLDataKind, 2033
Enumerations von simpleTypes, 235
EPUB-Dateien, 963
Erneut starten,
XSLT Debugger, 1423
Ersetzen, 127
mittels Regular Expressions, 157
Text, 1307
Text in Dokument, 1310
Text in mehreren Dateien, 1312
Erstellen,
Neues XML Dokument, 84
Erstellungsmodus, 598
Erweitern,
alles, 1504
Erweiterte Schemavalidierung, 456
Erweiterte Validierung,
in SchemaAgent, 1405
Erweiterungsfunktionen für XSLT und XQuery, 2236
Erweiterungsfunktionen in .NET für XSLT und XQuery,
siehe .NET Erweiterungsfunktionen, 2246
Erweiterungsfunktionen in Java für XSLT und XQuery,
siehe Java-Erweiterungsfunktionen, 2237
Erweiterungsfunktionen in MSXSL Scripts, 2253
Erzeugen,
DB auf Basis von Schema, 1484
Evaluierungs-Keycode,
für Ihre Altova-Software, 1654
Evaluierungszeitraum,
für Altova-Software-Produkte, 2273
von Altova Software-Produkten, 2273
Event, 1747, 1748, 1749, 1804, 1805, 1807, 1809, 1810, 1840, 1841, 1842, 1843, 1902, 1903, 1904
Excel 2007, 284, 955
Explorer, 1606
ExportSettings,
CreateKeys, 1881
ElementList, 1881
EntitiesToText, 1881
ExportAllElements, 1881
FromAttributes, 1882
FromSingleSubElements, 1882
FromTextValues, 1882
IndependentPrimaryKey, 1882
Namespace, 1882
SubLevelLimit, 1883
Extern geparste Entitäten, 1606
Externe Applikationen,
Dateien öffnen in, 1588
Externer JSON-Inhalt,
in die JSON-Grid-Ansicht kopieren, 702
mit Drag-and-Drop in die JSON Grid-Ansicht ziehen, 702
Externer XML-Inhalt,
in die XML-Grid-Ansicht kopieren, 309
mit Drag-and-Drop in die XML Grid-Ansicht ziehen, 309
Externer XSL Prozessor, 1633
Extras,
Menü, 1557
siehe auch externe Applikationen, 1588

F

Facets von simpleTypes, 235
Farbe, 1626
Favoriten, 1652
Fenster,
anordnen, 1650
Anzeige ein/auschalten, 1650
Anzeige verwalten, 115
automatisch ausblenden, 115
frei schwebend, angedockt, als Register, 115
Kacheln anzeigen, 1650
kaskadierende, 1650
Fenster "Meldungen",
Anzeige, 1650
Fenster "Trace", 560
Fenster Ablaufverfolgung,
im XSLT/XQuery Debugger, 555
Fenster automatisch ausblenden, 115
FileSelectionDlg,
Application, 1883
DialogAction, 1884
FullName, 1884
Parent, 1884
Filter in der JSON-Grid-Ansicht, 709
Einstellungen, 721

© 2016-2022 Altova GmbH

Filter in der XML-Grid-Ansicht, 317
Einstellungen, 329
Finden,
siehe Suchen, 1310
Firebird,
über ODBC verbinden, 1005
Verbindung über JDBC, 1003
Flächendiagrammfunktionen, 389
FOP,
Schriftarten, 347
Formatieren in der Textansicht, 145
Formeln in der JSON-Grid-Ansicht, 709, 710
Einstellungen, 721
Formeln in der XML-Grid-Ansicht, 319
Einstellungen, 329
Formulas,
anhand über die Tabellenlayoutvorschau erstellen, 935
Frei schwebende Fenster, 115
Freigeben,
aus Versionskontrolle, 1328
Funktionstasten, 1589

G

Gehe zu Datei, 1506
Gehe zu Zeile/Zeichen, 1505
Gemischte Markup-Symbole in (in Authentic)-Ansicht, 1436
Generate Sample XML, 2020, 2028, 2029
GenerateSampleXMLDlg,
Application, 1897
FillWithSampleData, 1899
NonMandatoryAttributes, 1899
NonMandatoryElements, 1900
Parent, 1900
RepeatCount, 1901
TakeFirstChoice, 1901
Generieren,
Code von Schema, 1156
DB-Struktur auf Basis von Schema, 1484
Geteilte Ansicht in der Textansicht, 166
Global, 2271
Globale,
Einstellungen, 1602
Globale Attribute, 182
Globale Attributgruppen, 182
Globale Elemente, 182
in XML-Schemas verwenden, 71
Globale Elemente anzeigen, 1398
Globale Kommentare,
Zeilenanzeige, 178
Globale Objekte,
in SPL, 1260
Globale Ressourcen, 1061
definieren, 1062, 1578
Konfigurationen ändern, 1078
Symbolleiste aktivieren, 1586
Typ Datei definieren, 1064
Typ Datenbank definieren, 1071
Typ Ordner definieren, 1069
verwenden, 1074, 1077, 1078
vom Typ Datei und Ordner verwenden, 1074
Globale Schemakomponenten,
suchen und umbenennen, 495
Grafikformate,
in der Authentic-Ansicht, 662
Grammatik, 1606
Graue Leiste, 1504
Grid-Ansicht, 97, 169, 1502, 1504
Dateneingaben, 88
Eingabehilfen verwenden, 97
Elemente und Attribute anhängen, 97
und Tabellenansicht, 99
Grid-Ansicht (JSON),
siehe JSON-Grid-Ansicht, 690
Grid-Ansicht (JSON) Einstellungen, 721
Grid-Ansicht (JSON) Tabellen,
als TSV oder JSON kopieren, 721
Grid-Ansicht (XML), 296
Anzeige filtern, 319
Bearben der Tabellenansicht, 312
Bilder in, 322
Diagramme erstellen in, 323
Dokumentanzeige, 298
Dokumentinhalt bearbeiten, 305
Dokumentstruktur bearbeiten, 302
Drag & Drop, 309
Eingabehilfen, 302
Formeln zur Berechnung einer Ausgabe, 317
Funktionalitäten, 298
Kontextmenü, 326
Metadatendatei, 317, 319
Nodes hinzufügen/löschen, 302
Node-Typen bearbeiten, 305

© 2016-2022 Altova GmbH

Grid-Ansicht (XML), 296
suchen und ersetzen, 305
Validierung, 305
XQuery-Ausdrücke in, 317, 319
Grid-Ansicht (XML) Einstellungen, 329
Grid-Ansicht (XML) Tabellen,
als TSV oder XML kopieren, 329
Grid-Ansicht von JSON-Dokumenten,
siehe JSON-Grid-Ansicht, 690
Grid-Schriftarten, 1626
GridView,
CurrentFocus, 1905
Deselect, 1905
IsVisible, 1905
OnBeforeDrag, 1902
OnBeforeDrop, 1903
OnBeforeStartEditing, 1903
OnEditingFinished, 1904
OnFocusChanged, 1904
Select, 1905
SetFocus, 1905
Große Markup-Symbole anzeigen, 624, 627
Große Markup-Symbole in (in Authentic)-Ansicht, 1436

H

Häkchen,
Rechtschreibprüfung, 1558
Hauptfenster, 116
Hilfe,
Inhaltsverzeichnis, 1652
Menü, 1652
Tastaturzuordnungen, 1653
Hilfesystem, 1652
Hintergrundinformationen, 2270
Hinzufügen,
Dateien zu Projekt, 1335
Ordner zu Projekt, 1337
HTML, 667
HTML-Ausgabe,
in der Authentic-Ansicht anhand der PXF-Datei generieren, 1438
HTML-Dokumente,
bearbeiten, 668
Info-Fenster, 668
HTML-Info-Fenster, 668
HTTP, 817
HTTP-Nachrichtenfenster,
einen Request importieren in, 823
Empfangen einer Response, 825
HTTP-Methoden, 818
Logs, 825
Requests senden, 818
und Accept Header, 825
und HTTP-Statuscodes, 825
und WADL, 823

I

IBM DB2,
Schemaverwaltung und -zuweisung, 1456
über JDBC verbinden, 1007
über ODBC verbinden, 1009
Verwalten von XML-Schemas, 1457
XML Schemas einer XML-Datei zuweisen, 1460
IBM DB2 für i,
über ODBC verbinden, 1016
Verbinden über JDBC, 1015
IBM Informix,
über JDBC verbinden, 1019
Identity Constraint,
in der Content Model-Ansicht ein- und ausblenden, 73
Identity Constraints in der Schema-Ansicht, 212, 220
Import,
Datenbankdaten basierend auf XML-Schema, 1483
Textdatei als XML, 1470
Import von Taxonomien, 1539
Importieren,
Datenbankstruktur als XML-Schema, 1478
MS Word-Dokument als XML, 1478
In Dateien suchen (Befehl), 1312
In Dateien suchen (Fenster), 127
In Element verpacken (Befehl),
in Grid-Ansicht, 1349
In Katalogen verwendete Umgebungsvariablen, 464
In OIM xBRL-CSV konvertieren, 1500
In OIM xBRL-JSON konvertieren, 1499
In OIM xBRL-XML konvertieren, 1499
In Schemas suchen, 482
Begriff ersetzen, 484
Begriff suchen, 484
Bereich definieren, 491

© 2016-2022 Altova GmbH

In Schemas suchen, 482
Ergebnisse, 494
Fenster, 494
globale Komponenten, 495
Suchbefehl ausführen, 949
Suche auf Komponenten einschränken, 485
Suche durch Eigenschaft und Eigenschaftswert einschränken, 488
Suchen- und Ersetzen-Befehl ausführen, 492
Umbenennen von globalen Komponenten, 495
In Schemas suchen (Fenster), 129
In XBRL suchen (Fenster), 129
Info-Fenster, 120
Anzeige, 1650
für CSS-Dokumente, 670
für HTML-Dokumente, 668
Info-Fenster in XSLT/XQuery Debugger, 555
Info-Fenster, XSLT-Register,
Beschreibung, 508
siehe auch XSL-Übersicht, 508
und Erstellen von ZIP-Ordnern, 508
und Projekte, 508
und XSLT-Dokumente, 508
Informationsfenster,
anordnen, 555
Inhaltsverzeichnis, 1652
Inline XBRL,
Verarbeitung, 1553
Whitespace-Behandlung, 1641
Inline XBRL in XMLSpy, 876
Installation,
Pfad von Beispieldateien, 138
Instanzen,
Anzahl, 51
Integrating,
XMLSpy in Applikationen, 2073
Integrationspaket für Eclipse,
installieren, 1148
Intelligentes Editieren, 1609
Internet, 1657
Internet-Verwendung,
in Altova-Produkten, 2271
ISO/IEC 10646, 2271

J

Java, 2034, 2084
API, 2034
Ausnahmen im generierten Code vermeiden, 1253
ClassPath, 2034
Code, 1156
Referenz zu generierten Klassen, 1237
Java Erweiterungsfunktionen,
Datentypkonvertierungen, Java in Xpath/XQuery, 2246
Instanzmethoden, Instanzfelder, 2244
Konstruktoren, 2243
statische Methoden, statische Felder, 2243
Java settings, 1638
Java virtual machine,
path setting, 1638
Java-Erweiterungsfunktionen,
benutzerdefinierte JAR-Dateien, 2242
benutzerdefinierte Klassendateien, 2239
Datentyp-Konvertierungen, XPath/XQuery in Java, 2245
für XSLT und XQuery, 2237
Übersicht, 2237
JDBC,
als Datenverbindungsschnittstelle, 969
mit Teradata verbinden, 1052
Verbindung einrichten (Windows), 992
JRE,
für das XMLSpy-Integrationspaket für Eclipse, 1148
JScript,
Skripterstellung mit XMLSpy, 1662
JSON, 667
JSON-Instanz von/in XML-Instanz konvertieren, 1493
JSON-Schema von/in XML-Schema konvertieren, 1497
und XPath, 769
und XPath/XQuery-Ausgabefenster, 769
und XQuery, 769
XQuery-Ausdrücke für, 772
JSON (Menü), 1367
JSON Lines, 684
JSON Schema, 676
Arrays, 748
atomare Typen, 750
Beschreibung, 682
einfache Typen, 750
nicht definierte Eigenschaften, 741

© 2016-2022 Altova GmbH

JSON Schema, 676
numerische Definitionen, 750
Objekte, 737, 741
Objekteigenschaften, 737
primitive Typen, 750
String-Definitionen, 750
Terminologie, 682
JSON-Daten,
Arrays, 679
Beispiel, 679
Objekte, 679
Typen, 679
JSON-Dokumente,
in der Grid-Ansicht bearbeiten, 690
in Textansicht bearbeiten, 685
in XMLSpy öffnen, 676
neue erstellen, 676
validieren, 765
von und in XML konvertieren, 685, 778
XQuery-Ausdrücke für, 601
JSON-Grid-Ansicht, 690
alle Filter und Formeln erneut auswerten, 1370
Ansicht filtern, 709
Autokomplettierung, 699
Befehlsschaltflächen innerhalb von Zellen, 697
Befehlsschaltflächen innerhalb von Zellen in der, 699
Bildanzeige in, 715
Diagramme erstellen in, 716
Dokumentanzeige in, 693
Dokumentinhalt bearbeiten in, 699
Dokumentstruktur bearbeiten in der, 697
Drag & Drop, 702
Formeln zur Berechnung einer Ausgabe, 709, 710
Komponente in Array verpacken, 1368
Komponente in Objekt verpacken, 1368
Komponenten als Child hinzufügen, 1368
Komponenten anhängen, 1368
Komponenten einfügen, 1368
Kontextmenübefehle, 718
Metadatendatei, 709, 710
Spalten in Tabelle sortieren, 1369
Symbole der Tabellenansicht, 704
Symbole für verschiedene Typen, 693
Tabellenansicht, 704
Tabellenansicht (Befehl), 1369
Typ (Befehl), 1367
XQuery-Ausdrücke in, 709, 710
Zeilen und Spalten von Tabelle tauschen, 1369
JSON-Metadatendatei für Filter und Formeln, 709
JSON-Schema, 684
Alle von, 758
anhand einer JSON-Instanz generieren, 774
Bedingungen, 760
beliebig, 752
Beliebiges von, 758
Eines von, 758
globale Definitionen hinzufügen, 729
if-then-else, 760
mehrere, 752
Nicht, 758
Objekte und Abhängigkeiten, 744
ohne Constraints, 752
Operatoren, 758
Typ-Selektoren (beliebig, mehrere), 752
Typ-Selektoren (ohne Constraings, unzulässig), 752
unzulässig, 752
JSON-Schema-Ansicht, 726
$ref, 733
Design-Ansicht, 735
Eingabehilfe "Constraints", 730
Eingabehilfe "Details", 730
Eingabehilfen, 730
erweiterte Referenzen auf JSON-Schemas, 733
externe Schemas hinzufügen, 730
globale und lokale Definitionen, 733
konfigurieren, 761
JSON-Schema-Version, 727
JSON-Text aus externen Quellen,
schnell zum Dokument hinzufügen, 767
JSON-Transformations mit XSLT/XQuery, 769

K

Kapitel, 1652
Katalog,
Oasis XML, 1353
Kataloganpassung, 466
Kataloge, 463
Kataloge in XMLSpy, 464
Kataloge und intelligente Bearbeitung, 466
Kataloge und Umgebungsvariablen, 467
Kerzendiagrammfunktionen, 389
Keycodes,
für Ihre Altova-Software, 1654

© 2016-2022 Altova GmbH

Klappleiste, 1506
Klappleiste in der Textansicht, 148
Kleine Markup-Symbole anzeigen, 624, 627
Kleine Markup-Symbole in (in Authentic)-Ansicht, 1436
Kodierung,
von Dateien, 1285
Kommentare, 178
Kommentare in der Schema-Ansicht, 182
Kommentare in SQL-Scripts, 1449
Kommunikationsvorgang,
SOAP Debugger, 804
Komponenten-Definitionen,
wiederverwenden, 63
Kompositor,
für Sequences, 51
in der Schema-Ansicht, 190
Kompositoren in der Schema-Ansicht, 192
Konfigurationen,
einer globalen Ressource, 1062, 1579
Konfigurationen in globalen Ressourcen, 1078
Konfigurieren,
XMLSPY-Benutzeroberfläche, 1691
Kontextfenster,
im XSLT/XQuery Debugger, 551
Kontextmenü,
anpassen, 1598
Befehle, 1593
Kontextmenüs,
in der Authentic-Ansicht, 634
Konvertieren,
Datenbankschema in XML-Schema, 1478
MS Word-Daten in XML, 1478
Oracle XML DB, 1465
Schema in DB-Struktur, 1484
Textdatei in XML, 1470
Kopieren,
Zeile (in Authentic-Ansicht), 1437
Kopieren (Befehl), 1298
Kreisdiagramme, 366
Kreisdiagrammfunktionen, 389
Kürzel, 1589
anzeigen in Tooltip, 1598
zuweisen/löschen, 1589

L

Leere Elemente, 1606
Leere Zeilen,
in XML-Dokumenten in der Textansicht, 293
Lesezeichen, 1652
einfügen und entfernen, 1313
navigieren, 1313
Lesezeichen in der Textansicht, 148
Lesezeichen in SQL-Scripts, 1449
Lesezeichenleiste, 1506
Liniendiagrammfunktionen, 389
Linkbases,
referenzieren, 1539
Linkbases in Taxonomien, 859
Linkroles in XBRL, 1535
Linkroles in XBRL-Taxonomien, 865
Links,
in der Authentic-Ansicht verwenden, 637
Little-endian, 1608
Lizenz, 2275
für Ihre Altova-Software, 1654
Informationen, 2273
Lizenzüberwachung,
in Altova-Produkten, 2274
loading, 1873
Löschen, 1584
Befehl aus Kontextmenü, 1593
Befehl aus Symbolleiste, 1584
Breakpoint - XSLT Debugger, 1424
Kürzel zu Befehl, 1589
Symbol aus Symbolleiste, 1584
Symbolleiste, 1586
Zeile (in Authentic-Ansicht), 1437
Löschen (Befehl), 1298

M

MainCatalog, 1353
Makro,
Hinzufügen zu Menü/Symbolleiste, 1595
Schaltfläche bearbeiten, 1598
Makros,

© 2016-2022 Altova GmbH

Makros,
aktivieren, 1675, 1687
ausführen, 1688
entwickeln, 1662, 1668
Makros anzeigen,
Skripting, 1567
MapForce, 1381
MariaDB,
über ODBC verbinden, 1021
Markup,
in Authentic-Ansicht, 624
Markup ausblenden, 624
Markup-Code (in Authentic)-Ansicht,
ausblenden, 1436
kleine/große/gemischte einblenden, 1436
Markup-Symbole,
in der Authentic-Ansicht, 627
Markup-Symbole (in Authentic)-Ansicht ausblenden, 1436
Markup-Symbole ausblenden, 627
Maximale Zellenbreite, 1617
Meldungen (Fenster), 122
Meldungsfenster,
in XSLT/XQuery Debugger, 553
Menü, 1593
anpassen, 1593
Ansicht, 1501
Bearbeiten, 1297
Befehl hinzufügen/löschen, 1584
Befehle löschen, 1593
Extras, 1557
Hilfe, 1652
JSON, 1367
Konvertieren, 1470
Makro hinzufügen, 1595
Projekt, 1316
SOAP, 1522
Standard/XMLSPY, 1593
WSDL, 1509
Menü "Fenster", 1650
Menü "Konvertieren", 1470
Menü "XBRL", 1533
Menüleiste, 133
Metadatendatei, 317, 319, 709
Methodennamen im generierten Code,
reservieren, 1253
Microsoft Access,
über ADO verbinden, 975, 1023
Microsoft Office 2007, 284, 955
Microsoft SQL Server,
über ADO verbinden, 1025
über ODBC verbinden, 1028
Microsoft® SharePoint® Server, 1340
MIME, 1606
Mixed Markup-Symbole anzeigen, 624, 627
Model Groups, 182
MongoDB und BSON, 754
MS SQL Server,
Schema Extensions, 1401, 1402
Schema-Einstellungen, 1402
MSXML, 1633
Code generieren für, 1254
msxsl:Script, 2253
Multi-user, 1603
MySQL,
über ODBC verbinden, 1034

N

Named Schema-Beziehungen,
MS SQL Server Schema-Einstellungen, 1402
Named Templates, 505
Named templates aufrufen, 505
Namepsaces,
in WSDL-Dokumenten, 789
Namespace Präfix,
in Grid-Ansicht einfügen, 1360
Namespaces,
Einstellungen in der Schema Design-Ansicht, 1386
in Schemas, 50
in XBRL-Taxonomien, 856
Namespaces in XBRL, 1537
Navigationsverlauf, 251
Navigieren,
Kürzel in Schema, 76
Nebeneinander, 1617
Neu laden, 1603
geänderte Dateien, 1285
Neue Datei,
erstellen, 1276
Neue Funktionen, 29
Neues XML-Dokument,
erstellen, 84
Nicht gespeicherte Änderungen, 1603

© 2016-2022 Altova GmbH

Nicht-XML Dateien, 1606
Node,
aktuellen Execution Node einblenden, 1424
Notations in der Schema-Ansicht, 182

O

OASIS,
XML-Katalog, 1353
Objektsuche,
im Fenster "Datenbankabfrage", 1445
ODBC,
als Datenverbindungsschnittstelle, 969
mit MariaDB verbinden, 1021
mit Teradate verbinden, 1053
Verbindung einrichten, 989
ODBC-Treiber,
Verfügbarkeit überprüfen, 989
Offene Content Models, 209
Öffentlicher Schlüssel von Zertifikaten, 425
Office Open XML, 284, 955
Öffnen,
Datei, 1280
OIM, 953, 1499, 1500
OLE DB,
als Datenverbindungsschnittstelle, 969
OOXML, 284
siehe Office Open XML, 955
Open Office XML,
Beispieldateien, 959
in der Archivansicht-Ansicht bearbeiten, 957
in der Archivansicht-Ansicht erstellen, 957
openContent, 182
OpenJDK,
als Java Virtual Machine, 992
Optimale Breite, 1505, 1617
Optionale Elemente,
erstellen, 59
Optionen,
3-D Diagramme, 400
Balkendiagrammfunktionen, 389
Diagramm x-Achse, 395
Diagramm y-Achse, 397
Diagramm z-Achse, 399
Diagrammfarben, 393
Diagrammgrößen, 400
Diagrammhintergrund, 386
Diagrammraster, 395, 397, 399
Diagrammschriften, 402
Diagrammtitel, 386
Einzelwertdiagrammfunktionen, 389
Flächendiagrammfunktionen, 389
Kerzendiagrammfunktionen, 389
Kreisdiagrammfunktionen, 389
Legende, 386
Liniendiagrammfunktionen, 389
Optionen beim Öffnen,
Datei, 1603
Oracle,
Schema Extensions, 1399
Oracle XML Datenbank, 1465
Oracle XML DB,
Datenbank durchsuchen, 1468
manage XML Schemas, 1465
Oracle-Datenbank,
über JDBC verbinden, 1036
über ODBC verbinden, 1038
Overrides, 182

P

Parameter,
in DB-Abfragen, 651
Übergabe an Stylesheet über Schnittstelle, 1414
Parent, 1862
Parser,
in Altova-Produkte integrierter, 2270
XSLT, 1633
Patterns von simpleTypes, 235
PDF,
Transformieren in PDF in XMLSpy, 501
PDF-Ausgabe,
in der Authentic-Ansicht anhand der PXF-Datei generieren, 1438
PDF-Schriftarten, 347
Pending Update List (PUL), 529
Plattformen für Altova-Produkte, 2270
Plug-In, 1597
ATL-Beispieldateien, 1694
Konfiguration der Benutzeroberfläche, 1691
Registrierung, 1690
XMLSPY, 1690

© 2016-2022 Altova GmbH

Position, 1505
Zeichen, 1505
Zeile, 1505
PostgreSQL,
nativ verbinden, 997
über ODBC verbinden, 1043
PowerPoint 2007, 284, 955
Pretty-Print,
in der Text-Ansicht, 1306
Private Key von Zertifikaten, 425
Privater Schlüssel von Zertifikaten, 425
Processing Instructions in der Schema-Ansicht, 182
Profiler, 567, 1420
Profilerstellung, 567
Callgraph, 567
Trefferquote, 567
Programmeinstellungen, 1602
Progress OpenEdge-Datenbank,
über ODBC verbinden, 1047
verbinden über JDBC, 1045
Projekt,
aktive Dateien hinzufügen, 1336
Dateien zu Projekt hinzufügen, 1335
Eigenschaften, 1344
externen Webordner hinzufügen, 1340
Hinzufügen aktiver und verwandter Dateien, 1336
Hinzufügen zu Versionskontrolle, 1327
Menü, 1316
Neu laden, 1319
öffnen, 1319
Ordner hinzufügen, 1337
schließen, 1319
Speichern, 1320
URL zu Projekt hinzufügen, 1336
zuletzt verwendete, 1347
Projekte,
Batchverarbeitung mit Projekten, 1086
benennen, 1081
Eigenschaften, 1081
erstellen und bearbeiten, 1081
externe Ordner hinzufügen, 1337
globale Ressourcen hinzufügen, 1336
neue erstellen, 1319
Pfad installierter Dateien, 138
speichern, 1081
Übersicht, 1080, 1316
verwenden, 1086
Vorteile, 1086
Projekte in XMLSpy,
erstellen, 110
Vorteile, 110
Projektfenster, 118
Projekt-Fenster,
Anzeige, 1650
Projektverwaltung in XMLSpy, 110
PUBLIC,
Identifier - Katalog, 1353
Public Key von Zertifikaten, 425
PUL in XQuery Update, 529
PXF-Datei,
Ausgabedokumente von der Authentic-Ansicht aus generieren, 1438

R

Rechtliches, 2273
Rechtschreiboptionen, 1562
Rechtschreibprüfung, 1558
benutzerdefiniertes Wörterbuch, 1558
Redefines, 182
Reduzieren,
unselektierte, 1504
Referenz für Programmierer, 1660
Referenzieren von JSON-Schemas, 733
Regeln,
für Schemavalidierung (siehe Schemaregeln), 456
Regionen in SQL-Scripts, 1449
Registrierung,
Plug-In, 1690
Registrierung der Altova-Software, 1654
Registry,
Einstellungen, 1602
Regular Expression,
in Suchstring, 1307
Regular Expressions, 1312
suchen und ersetzen, 157
Request,
SOAP Request senden, 1524
von WSDL senden, 797
RichEdit, 1437
RootCatalog.xml, 464
RTF-Ausgabe,
in der Authentic-Ansicht anhand der PXF-Datei generieren, 1438

© 2016-2022 Altova GmbH

Rückgängig, 1298

S

save, 1865
schema, 1758
Code Generator, 1156
DB auf Basis von Schema erzeugen, 1484
Design-Ansicht, 1502
Dokumentation, 78
Dokumentationsschriftart, 1628
einer DB-XML zuweisen, 1460
Einstellungen, 1603
Einstellungen für Oracle, 1400
in UML konvertieren, 1379
siehe auch XML-Schema, 1371
siehe XML-Schema, 48
Verwaltung und Zuweisung in IBM DB2-Datenbanken, 1456
WSDL Schema öffnen, 796
Schema Design,
Menü, 1386
Schema Design-Ansicht,
Diagramm anzeigen, 1398
Globale Elemente anzeigen, 1398
Zoom-Funktion, 1398
Schema View, 171
SchemaAgent, 479
arbeiten mit, 474, 475, 479
erweiterte Validierung, 1405
Schemas anzeigen, 1404
Schemas von XMLSpy aus öffnen, 479
Verbindung trennen, 1404
Verbindung zu Server herstellen von XMLSpy, 1403
SchemaAgent in XMLSpy, 470
SchemaAgent Server,
Verbindung herstellen, 471
SchemaAgent-Validierung, 479
Schema-Ansicht,
Eingabehilfe "Details", 233
Eingabehilfe "Komponenten", 229, 235
Eingabehilfen, 229
konfigurieren, 57
vorwärts und rückwärts navigieren, 251
Schema-Ansicht, suchen in,
siehe In Schemas suchen, 482
Schemabearbeitung,
Annotationen, 190
Content Model, 190
Kommentare, 190
Processing Instructions, 190
SchemaDocumentationDlg,
AllDetails, 1907
Application, 1907
IncludeAll, 1909
IncludeAttributeGroups, 1909
IncludeComplexTypes, 1909
IncludeGlobalElements, 1910
IncludeGroups, 1910
IncludeIndex, 1911
IncludeLocalElements, 1911
IncludeRedefines, 1912
IncludeSimpleTypes, 1912
OptionsDialogAction, 1913
OutputFile, 1913
OutputFileDialogAction, 1913
OutputFormat, 1914
Parent, 1914
ShowAnnotations, 1914
ShowAttributes, 1915
ShowChildren, 1915
ShowConstraints, 1916
ShowDiagram, 1915
ShowEnumerations, 1916
ShowNamespace, 1916
ShowPatterns, 1917
ShowProgressBar, 1917
ShowProperties, 1917
ShowResult, 1917
ShowSingleFacets, 1918
ShowSourceCode, 1918
ShowType, 1918
ShowUsedBy, 1919
Schema-Manager,
CLI-Befehl Help, 442
CLI-Befehl Info, 442
CLI-Befehl Initialize, 443
CLI-Befehl Install, 443
CLI-Befehl List, 444
CLI-Befehl Reset, 445
CLI-Befehl Uninstall, 445
CLI-Befehl Update, 446
CLI-Befehl Upgrade, 447
CLI-Übersicht, 441

© 2016-2022 Altova GmbH

Schema-Manager,
Patch für Schema installieren, 439
Schema deinstallieren, 440
Schema installieren, 439
Schemas nach Status auflisten in, 437
starten, 435
Status von Schemas, 437
Übersicht, 433
Upgrade für Schema installieren, 439
zurücksetzen, 440
Schemamodule, 452, 1405, 1407
schemanativetype, 1257
Schemaregeln, 456
definieren, 458
Regeln zu einem Schema hinzufügen, 456
Schemas,
für IBM DB2 verwalten, 1457
im Speicher, 1385
über Kataloge finden, 466
Schemas und Kataloge, 463
Schemas, suchen in,
siehe In Schemas suchen, 482
Schemaschriftarten, 1628
Schema-Übersicht, 48
Bearbeiten in der Schema-Übersicht, 178
Komponenten sortieren, 178
Symbole, 178
und Content Model-Ansicht, 177, 178
und globale Kommentare, 178
und Zeilenanzeige von globalen Kommentaren, 178
Schließen,
Projekt, 1319
Schriftart, 1628
Schema, 1628
Schema-Dokumentation, 1628
Schriftarten in der Textansicht, 145
Scripts in XSLT/XQuery,
siehe Erweiterungsfunktionen, 2236
Sekundärschlüssel,
in SQLite deaktivieren, 1001
Sequence-Kompositor,
verwenden, 51
SharePoint® Server, 1340
Shortcuts, 286
Sich wiederholende Elemente, 1609
Signatur,
siehe XML-Signatur, 1360
Signaturen,
siehe XML-Signaturen, 417
SimpleType,
Beispielwerte von, 235
Enumerations von, 235
Facets definieren, 235
Patterns von, 235
simpleType Ableitungen, 233
SimpleTypes,
anonymous, 182
global, 182
in Schema-Definitionen, 63
named, 182
Skript-Editor,
Übersicht, 1662, 1664
Skripting,
Makros anzeigen, 1567
Zu Skripting-Umgebung wechseln, 1566
Skripts, 1645
Skriptsprache, 1645
Smart Fix für XML-Schemas, 239
Smart Restrictions, 244
SOAP, 788, 801
Debugger Optionen, 1531
Debugger Sitzung, 1529
Menü, 1522
Neuen Request erstellen, 1522
Proxyserver anhalten, 1531
Proxyserver starten, 1530
Request an Server senden, 1524
Request von WSDL senden, 797
Request-Einstellungen, 1525
Requests, 1530, 1531
Response, 1531
SOAP Debugger, 802
Breakpoints setzen, 810, 814
SOAP Kommunikationsvorgang, 804
SOAP-Validierung, 801
Software-Produktlizenz, 2275
Sortieren in Tabellen (XML-Grid-Ansicht), 1350
Speicher,
Speichern von Schemainformationen, 1385
Speichern,
Projekt, 1320
SPL, 1256
Codeblöcke, 1256
Dateien verwenden, 1262
foreach, 1265
globale Objekte, 1260

© 2016-2022 Altova GmbH

SPL, 1256
Variablen, 1259
Splashscreen, 1617
SPP Datei-Speicherpfade, 1316
SPS,
neuer XML-Datei zuweisen, 1276
SPS-Datei,
zu XML-Datei zuweisen, 1430
SPS-Tabellen,
dynamischen Tabellen bearbeiten, 624
SPS-Tabellen in der Authentic-Ansicht,
Verwendung, 642
SpyProject,
CloseProject, 1920
ProjectFile, 1920
RootItems, 1920
SaveProject, 1921
SaveProjectAs, 1921
SpyProjectItem,
ChildItems, 1921
FileExtensions, 1922
ItemType, 1922
Name, 1922
Open, 1922
ParentItem, 1922
Path, 1923
ValidateWith, 1923
XMLForXSLTransformation, 1923
XSLForXMLTransformation, 1923
XSLTransformationFileExtension, 1923
XSLTransformationFolder, 1923
SpyProjectItems,
AddFile, 1924
AddFolder, 1924
AddURL, 1924
Count, 1925
Item, 1925
RemoveItem, 1925
SQL Editor,
Abfrage erstellen, 1453
Beschreibung, 1449
im Fenster Datenbankabfrage, 1449
SQL Server,
über ADO verbinden, 975
über ADO.NET verbinden, 982
über JDBC verbinden, 992
XML-Schemas verwalten, 1462
SQLite,
Sekundärschlüssel deaktivieren, 1001
Verbindung einrichten (Windows), 1001
Standard,
Encoding, 1608
Standardansicht,
Einstellungen im Hauptfenster, 1606
Standard-Editor, 1606
Standardwert,
Menü, 1593
Start,
XSLT Debugger, 1422
Startgruppe,
hinzufügen (Kontextmenü), 1598
Statische (SPS) Tabellen in der Authentic-Ansicht,
Verwendung, 642
Statusleiste, 133
Stil, 1626
Stoppen,
XSLT Debugger, 1423
Strukturierter Text, 1609
Stylesheet PI, 1420, 1421
StyleVision, 1381
und XBRL, 1546
zur Bearbeitung eines StyleVision Power Stylesheet, 1430
StyleVision Power Stylesheet,
in StyleVision bearbeiten, 1430
zu XML-Datei zuweisen, 1430
StyleVision Power Stylesheet zu XML-Datei zuweisen, 1430
Suchen,
mittels Regular Expressions, 157
Text in Dokument, 1307
und Text in Dokument ersetzen, 1310
Suchen in Schemas,
siehe In Schemas suchen, 482
Suchen in XBRL, 947
Ergebnisse, 951
Fenster, 951
Suchbegriff, 947
Support Center, 1657
Support-Optionen, 138
Sybase,
über JDBC verbinden, 1050
Symbol,
Hinzufügen zu Symbolleiste/Menü, 1584
Symbole,
große anzeigen, 1598
Symbolleiste, 133, 1586

© 2016-2022 Altova GmbH

Symbolleiste, 133, 1586
Befehle anpassen, 1584
Ein/Aus, 1586
große Symbole anzeigen, 1598
Makro hinzufügen, 1595
Neue erstellen, 1586
Symbolleiste & Menübefehle zurücksetzen, 1586
Syntaxfärbung, 1606, 1617
für XQuery, 519
Syntaxüberprüfung von JSON-Dokumenten, 685, 690
System-DSN,
einrichten, 989

T

Tabelle,
automatisch erstellen, 1606
Tabellen,
dynamische (SPS)-Tabellen (SPS) bearbeiten, 624
in der Authentic-Ansicht, 613
Tabellen in der Authentic-Ansicht,
Arbeiten mit (statischen und dynamischen) SPS Tabellen, 642
Arbeiten mit XML-Tabellen, 644
Symbole zum Bearbeiten von XML-Tabellen, 648
Verwendung, 642
Tabellen in der JSON-Grid-Ansicht,
als TSV oder JSON kopieren, 721
Tabellen in der XML-Grid-Ansicht,
als TSV oder XML kopieren, 329
Tabellenansicht, 1609
Verwendung, 99
Tabellenansicht der JSON-Grid-Ansicht,
bearbeiten in, 704
in Arbeitsblätter exportieren/aus Arbeitsblättern importieren, 704
Tabellenansicht der XML-Grid-Ansicht,
bearbeiten in, 312
in Arbeitsblätter exportieren/aus Arbeitsblättern importieren, 312
Tabellenlayoutvorschau, 933
Formulas erstellen, 935
Tabellenparameter, 929
Tabulatorgröße,
einstellen, 1506
und Pretty-Print, 1506
Tabulatorzeichen, 1603
Target Namespaces in XBRL, 1538
Tastaturzuordnungen, 1653
Tatstaturkürzel, 1589
Taxonomie-Manager,
CLI-Befehl Help, 840
CLI-Befehl Info, 840
CLI-Befehl Initialize, 841
CLI-Befehl Install, 841
CLI-Befehl List, 842
CLI-Befehl Reset, 843
CLI-Befehl Uninstall, 844
CLI-Befehl Update, 845
CLI-Befehl Upgrade, 845
CLI-Übersicht, 839
Installieren einer Taxonomie, 837
Patch auf Taxonomie anwenden, 837
starten, 834
Status von Taxonomien, 836
Taxonomie deinstallieren, 838
Taxonomien nach Status auflisten, 836
Übersicht, 831
Upgrade für eine Taxonomie installieren, 837
zurücksetzen, 838
Taxonomien,
Beziehungen, 865
Beziehungen in, 866, 869
Dateien in, 848, 859
Elemente hinzufügen, 861
importieren, 852, 1539
Namespaces, 856, 1537
neue erstellen, 851
Schritte beim Erstellen, 847
Target Namespace von, 856
Target Namespaces, 1538
und der neue Taxonomieassistent, 851
und Linkbases, 859
und Linkroles, 865
Taxonomien in XBRL, 847, 848
Technische Informationen, 2270
Technischer Support, 1657
Templates-Fenster,
in XSLT/XQuery Debugger, 554
Teradata,
über JDBC verbinden, 1052
über ODBC verbinden, 1053
terminate, 1761
Text,
in der Authentic-Ansicht bearbeiten, 637

© 2016-2022 Altova GmbH

Text,
in der Authentic-Ansicht formatieren, 637
in Dokument suchen, 1307
Pretty-Print-Anzeige, 1306
suchen und ersetzen, 1310
Text aus der Grid-Ansicht als tabulatorgetrennten Text kopieren, 1300
Text der Grid-Ansicht kopieren, 1299
Textansicht, 144
Autokomplettierung ein/aus, 1609
bearbeiten in, 89
Eingabehilfen, 165
Einrückung, 145, 148
Formatierung von Text, 145
geteilte Ansicht, 166
intelligente Bearbeitungsfunktionen, 151
Klappleiste, 148
Lesezeichen, 148
Schemaschriftarten, 1628
Schriftarteigenschaften, 145
spezielle Bearbeitungsfunktionen für XML-Dokumente, 293
und Kommentare in XML-Dokumenten, 293
und leere Zeilen in XML-Dokumenten, 293
Zeilennummerierung, 148
Zeilenumbruch, 145
Text-Ansicht, 1501
Textdatei,
als XML importieren, 1470
Export von XML-Daten, 1487
TextImportExportSettings,
DestinationFolder, 1926
EnclosingCharacter, 1926
Encoding, 1926
EncodingByteOrder, 1927
FieldDelimiter, 1927
FileExtension, 1927
HeaderRow, 1927
ImportFile, 1927
Themen,
Ansicht im Inhaltsverzeichnis, 1652
Tooltip, 1598
anzeigen, 1598
Kürzel anzeigen, 1598
Tracepoint,
Dialogfeld, 1426
Tracepoint aktivieren - XSLT Debugger, 1425
Tracepoints,
in XSLT/XQuery Debugger verwenden, 560
Tracepoints im XPath/XQuery Debugger, 591
Transformation,
siehe XSLT-Transformation, 1412
Trefferquote-Profilerstellung, 567
TSV,
Tabellen der JSON-Grid-Ansicht kopieren als, 721
Tabellen der XML-Grid-Ansicht kopieren als, 329
Tutorial, 36
Beispieldateien, 36
WSDL, 789
Ziele, 36
Tutorials,
Pfad installierter Dateien, 138
Typ,
Erweiterung im XML-Dokument, 85
Typen,
integrierte, 1269

U

Übernehmen, 1602
Übersicht über den Katalogmechanismus, 463
Überspringen,
XSLT Debugger, 1424
Überwachungsausdrücke im XPath/XQuery Debugger, 591
UCS-2, 1608, 2271
UCS-4, 2271
Umgebungsvariablen, 467
ANT_OPS, 1253
UML,
Schemas konvertieren, 1379
Unicode,
Konsortium, 2271
Standard, 2271
Unnamed Element Relationships,
MS SQL Server Schema-Einstellungen, 1402
Unselektierte, 1504
URL, 1764, 1765, 1865, 1873
per E-Mail senden, 1292
UTF-16, 1608
UTF-8, 2271

© 2016-2022 Altova GmbH

V

Validator,
in Altova-Produkten, 2270
Validieren,
WSDL-Datei, 795
XML-Dokumente, 93
Validieren von XML-Dokumenten, 291
Validierung, 135, 1353
DTD zu XML-Dokument zuweisen, 1371
durch Schemaregeln erweitern, 456
mittels Schemaregeln erweitern, 456
von verknüpften Schemas mit Hilfe von SchemaAgent, 479
WSDL-Dateien, 1358
XML-Schema zu XML-Dokument zuweisen, 1372
Validierung von XML-Schemas, 239
Validierungseinstellungen, 1603
Validierungsmeldungen, 122
Variablen,
in SPL, 1259
Variablen im XPath/XQuery Debugger, 591
Variablenfenster,
in XSLT/XQuery Debugger, 552
VBScript,
Skripterstellung mit XMLSpy, 1662
Verbindung mit SchemaAgent Server trennen, 1404
Verbindung zu SchemaAgent Server herstellen, 471
Verbindung zu SchemaAgent Server von XMLSpy, 1403
Vergleiche,
von Dateien, 1112
von Dateien und Verzeichnissen, 1111
von Verzeichnissen, 1113
Vergrößern/Verkleinern,
in Schema Design-Ansicht, 1398
Verschieben-Befehle,
in Grid-Ansicht, 1350
Versionskontrolle, 1645
aktivieren /deaktivieren, 1322
aktuellste Version holen, 1322
auschecken, 1324
Auschecken rückgängig, 1327
ausgliedern, 1328
Dateien abrufen, 1323
Eigenschaften, 1333
Hinzufügen in Versionskontrolle, 1327
installieren eines Versionskontroll-Plug-in, 1115
Projekt öffnen, 1321
Status aktualisieren, 1334
Unterschiede anzeigen, 1331
unterstützte Anbieter, 1320
Verlauf anzeigen, 1330
Versionskontrollsystem wechseln, 1334
Versionskontrollmanager, 1334
Vertrauenswürdige Pfade für Authentic Skripts, 1438
Vertrieb,
von Altova Software-Produkten, 2273
von Altova-Software-Produkten, 2273
Verzeichnisse,
mit Application.URLMakeDirectory erstellen, 1765
zwei vergleichen, 1572
Verzeichnisse vergleichen, 1572
Verzeichnisvergleiche, 1111, 1113
Visual Studio,
Code generieren für, 1254
die XMLSpy ActiveX Controls zur Toolbox hinzufügen, 2075
Visual Studio .Net,
und XMLSPY, 1141
und XMLSpy-Unterschiede, 1143
und XSLT Debugger, 1146
Vorlagen,
Ordner, 36
von XML-Dokumenten in Authentic-Ansicht, 1429
Vorlagendateien,
für neue Dokumente, 1276
Vorwärts,
in der Schema-Ansicht, 251
VS .NET,
und XMLSpy Integrationspaket, 1142

W

WADL,
Verwendung für HTTP- Requests, 823
Web Server, 1657
Webservice,
Verbindung herstellen, 796
Wechseln,
Zu Skript-Umgebung, 1566
Whitespace,
entfernen, 1306

© 2016-2022 Altova GmbH

Whitespace in Inline XBRL, 1641
Whitespace-Behandlung, 341
Whitespace-Markierungen, 1506
Whitespaces entfernen, 1306
Windows,
Unterstützung für Altova-Produkte, 2270
Wohlgeformtheit von XML-Dokumenten, 291
Wohlgeformtheitsprüfung, 1351
bei XML-Dokument, 93
Wohlgefortmheitsprüfung von JSON-Dokumenten, 685, 690
Word 2007, 284, 955
Word 2007+-Ausgabe,
in der Authentic-Ansicht anhand der PXF-Datei generieren, 1438
Word-Dokument,
als XML importieren, 1478
Wörterbuch, 1558
benutzerdefiniertes hinzufügen, 1558
bestehendes ändern, 1558
Rechschreibprüfung, 1558
Wrapper-Klassen,
im generierten Code, 1254
WSDL, 788, 796
2.0-Komponenten, 1512
Binding in 1.1, 1511
Binding in 2.0, 1514
Dokumentation erstellen, 798
Einstellungen, 1521
Erstellen von Bindings, 792
Erstellen von Messages, 790
Erstellen von Operationen, 790
Erstellen von Parametern, 790
Erstellen von PortTypes, 790
Erzeugen von Dokumentation, 1517
Interface in 2.0, 1513
Messages in 1.1, 1510
Namespaces, 789
neues Dokument erstellen, 789
Operations in 1.1, 1510
Ports erstellen, 794
PortType in 1.1, 1511
Schema öffnen, 796
Service in 1.1, 1512
Service in 2.0, 1515
Services erstellen, 794
SOAP Debugger, 802
SOAP Request senden, 797
Tutorial, 789
Types, 1516
validieren, 795
Verbindung zu einem Webservice herstellen, 796
Verwendung für HTTP- Requests, 823
von 1.1 in 2.0 konvertieren, 1521
Web service, 1522
WSDL Design-Ansicht,
Beschreibung, 253
Bindings, 254
Dateien anzeigen, 253
Funktionalitäten, 253
Hauptfenster, 254
PortTypes, 254
Services, 254
WSDL Menü, 1509
WSDL-Ansicht,
Details-Eingabehilfe, 258
Eingabehilfen, 258
importieren in ein WSDL-Dokument, 258
Übersicht-Eingabehilfe, 258
WSDL-Dateien,
Erweiterungselemente, 1358
Validierung, 1358
WSDL-Schriftarten, 1631

X

XBRL, 829
arcroles, 1533
Dokumentation generieren, 1541
Linkroles, 1535
Namespaces, 1537
Target Namespaces, 1538
und MapForce, 1546
und StyleVision, 1546
Validierung, 954
XULE, 938
xBRL OIM-Formate, 953
XBRL-Ansicht, 266
Eingabehilfen, 274
Einstellungen, 1545
Register Berechnung im Hauptfenster, 271
Register Darstellung im Hauptfenster, 271
Register Definitionen im Hauptfenster, 271
Register Elemente im Hauptfenster, 266

© 2016-2022 Altova GmbH

XBRL-Ansicht, suchen in,
siehe Suchen in XBRL, 947
xBRL-CSV (OIM), 1500
xBRL-JSON (OIM), 1499
XBRL-Schriftarten, 1632
XBRL-Tabellendefinitionseditor, 904
XBRL-Taxonomien, 847
siehe auch Taxonomien, 848
XBRL-Validierungsoptionen, 1639
xBRL-XML (OIM), 1499
Xerces,
Code generieren für, 1254
XInclude,
in Grid-Ansicht einfügen, 1301
in Textansicht einfügen, 1301
in XML-Dokument einfügen, 293
XML, 83, 2271
Dokument, 83
Menü, 1348
Neues Dokument, 84
Oasis-Katalog, 1353
Rechtschreibprüfung, 1558
Symbole, 83
XML Diff,
Dateien vergleichen, 1575
Dateivergleich, 1567
Verzeichnisse vergleichen, 1572
XML Import,
Basierend auf Schema, 1483
XML Parser,
Info, 2270
XML Schema,
MS SQL Server Schema-Einstellungen, 1402
xml:base, 249
xml:id, 249
xml:lang, 249
xml:space, 249
XMLData,
AppendChild, 1971
EraseAllChildren, 1972
EraseCurrentChild, 1973
GetChild, 1974
GetChildKind, 1975
GetCurrentChild, 1975
GetFirstChild, 1975
GetNextChild, 1976
HasChildren, 1978
HasChildrenKind, 1978
InsertChild, 1978
IsSameNode, 1979
Kind, 1980
MayHaveChildren, 1980
Name, 1980
Parent, 1980
TextValue, 1981
XML-Datei,
von DTD oder XML-Schema generieren, 1381
XML-Datei Global Resources, 1062
XML-Daten,
in Datenbank exportieren, 1490
in Textdatei exportieren, 1487
XML-Daten in Datenbank exportieren, 1490
XML-Daten in Textdateien exportieren, 1487
XML-Datenbank,
neue Datenzeile in die Authentic-Ansicht laden, 1431
XML-DB,
neue XML-Datenzeile laden, 650
XML-Dokument,
anhand von DTD generieren, 448
Bearbeiten in der Authentic-Ansicht, 335
generieren anhand von XML-Schema (Enterprise und Professional Edition), 451
in der Authentic-Ansicht öffnen, 609
in Textansicht bearbeiten, 89
Oracle XML DB durchsuchen, 1468
zu XSLT Stylesheet zuweisen, 1421
XML-Dokumente, 288
automatische Validierung, 289
Automatisieren von XQuery-Ausführungen, 345
Automatisieren von XSLT-Transformationen, 345
Bearbeitungsfunktionen in der Textansicht, 293
Dateipfade einfügen, 293
in der Grid-Ansicht bearbeiten, 296
Kodierung, 430
öffnen, 289
Schemas generieren, 430
Schemas zuweisen (einschl. DTDs), 291
speichern, 289
Standardansichten, 289
Suchen und Ersetzen in XML-Dokumenten, 430
Text importieren und exportieren, 430
transformieren mit XSLT, 345
Überprüfung der Gültigkeit (Validierung), 93
und Kommentieren in der Textansicht, 293
und leere Zeilen in der Textansicht, 293
und XPath-Ausdruck eines Node, 293

© 2016-2022 Altova GmbH

XML-Dokumente, 288
und XQuery, 345
validieren, 291
Wohlgeformtheit prüfen, 291
XInclude einfügen, 293
XPath-Ausdrücke auswerten, 430
XML-Grid-Ansicht,
siehe Grid-Ansicht, 296
XML-Konformität, 1606
XML-Metadatendatei für Filter, 317
XML-Metadatendatei für Formeln, 319
XML-Schema, 48, 1371, 1386
<alternative>-Element, 205
Ansicht konfigurieren, 57
Code generieren von, 1385
Content Model-Ansicht konfigurieren, 1395
Content Model-Diagramm, 1389
Dokumentation generieren, 1389
einer DB-XML zuweisen, 1460
einfaches Schema erstellen, 48
Einstellungen in der Schema Design-Ansicht, 1386
Entwurf anhand von XML-Datei erzeugen, 1381
für IBM DB2 verwalten, 1457
Gehe von XML-Dokument zu Definition, 1373
Gehe von XML-Dokument zu Schema, 1373
generieren anhand von DTD (Enterprise und Professional Edition), 448
globale Komponenten, 182
in DTD konvertieren, 1376
Komponenten hinzufügen, 51
Menübefehle zu, 1371
MS SQL Server Extensions, 1401
MS SQL Server Schema-Einstellungen, 1402
Namespace-Einstellungen in der Schema Design-Ansicht, 1386
Namespaces definieren, 50
Navigation in der Design-Ansicht, 76
neue Datei erstellen, 48
Oracle Extensions, 1399
Oracle Schema-Einstellungen, 1400
siehe auch Schema, 1371
Smart Fix, 239
Tutorial, 48
Validierunge, 239
Verwaltung und Zuweisung in IBM DB2-Datenbanken, 1456
von XML-Dokument generieren, 1374
während der Bearbeitung des XML-Dokuments ändern, 103
zu XML-Dokument zuweisen, 1372
XML-Schema-Definitionen,
komplexe Definitionen, 63
XML-Schemas, 432
andere Schemas inkludieren, 452
in der Grid-Ansicht bearbeiten (Enterprise und Professional Edition), 451
in DTD konvertieren (Enterprise und Professional Edition), 451
in Module aufteilen, 1405
in Textansicht bearbeiten, 451
inkludierte Schemas auf eine Ebene bringen (nivellieren), 1407
inkludierte Schemas nivellieren, 452
plus DTDs, 291
und globale Ressourcen, 291
XML-Dokumentation generieren anhand von, 451
XML-Signatur, 1432
XML-Signaturen, 417, 661
erstellen, 419, 1360
überprüfen, 422, 1363
XMLSpy, 1275
Integration, 2073
Plug-In Registrierung, 1690
XMLSpy API,
Dokumentation, 1706
XMLSpy Debugger-Perspektive, 1155
XMLSpy Enterprise Edition,
Benutzerhandbuch, 28
XMLSpy in Eclipse, 1147
XMLSpy Integrationspaket, 1142
XMLSPY Plug-In, 1690
XMLSpy Plugin für VS .NET,
installieren, 1142
XMLSpyCommand,
in XMLSpyControl, 2114
XMLSpyCommands,
in XMLSpyControl, 2116
XMLSpyControl, 2117
Beispiele für die Integration auf Dokumentebene, 2081
Dokumentation, 2073
Integration mittels C#, 2081
Objektreferenz, 2113
XMLSpyControlDocument, 2125
XMLSpyControlPlaceHolder, 2132
XMLSpyDocumentEditor,
MarkUpView, 2014
XMLSpyLib, 1706

© 2016-2022 Altova GmbH

XMLSpyLib, 1706
Application, 1746
AuthenticDataTransfer, 1768
AuthenticRange, 1773
AuthenticSelection (obsolete), 1994
AuthenticView, 1803
CodeGeneratorDlg, 1822
DatabaseConnection, 1828
Dialogs, 1835
DocEditEvent (obsolete), 1981
DocEditView (obsolete), 1996
Document, 1838
Documents, 1870
DTDSchemaGeneratorDlg, 1874
ElementList, 1878
ElementListItem, 1879
ExportSettings, 1880
FileSelectionDlg, 1883
GenerateSampleXMLDlg, 1896
GridView, 1902
ProjectItem, 1921
SchemaDocumentationDlg, 1906
SpyProject, 1920
SpyProjectItems, 1924
TextImportExportSettings, 1925
XMLData, 1970
XMLSpy-Perspektive in Eclipse, 1150
XMLSpy-Plugin für Eclipse,
siehe Integrationspaket für Eclipse, 1148
XML-Tabellen in der Authentic-Ansicht,
Editiersymbole, 648
Verwendung, 644
XML-Text, 1609
XML-Text aus externen Quellen,
schnell zum Dokument hinzufügen, 343
XML-Text ein- und auskommentieren, 1314
XML-Vorlagendatei,
in Authentic-Ansicht, 609
XPath,
auswerten, 124
Generieren eines Node in einem XML-Dokument, 293
XPath (Fenster), 124
XPath 1.0,
in XPath Evaluator, 1351
XPath 2.0,
in XPath Evaluator, 1351
XPath auswerten, 124
XPath des ausgewählten Node im XML-Dokument,
in die Zwischenablage kopieren, 1301
XPath Evaluator,
Verwendung, 1351
XPath kopieren, 1301
XPath zum ausgewählten Node, 622
XPath/XQuery-Ausdrücke,
auswerten, 585
auswerten und debuggen, 582
Debuggen, 591
für JSON-Dokumente, 601
im Fenster "XPath/XQuery" erstellen, 598
im XPath/XQuery-Fenster, 604
XPath/XQuery-Fenster, 582, 583
XPaths,
für tracepoints definieren, 560
XPath-Watch-Fenster,
in XSLT/XQuery Debugger, 552
XPointer,
Generieren eines Node in einem XML-Dokument, 293
XPointer des ausgewählten Node im XML-Dokument,
in die Zwischenablage kopieren, 1301
XPointer kopieren, 1301
XQuery, 524
Ausführung, 524
Datei öffnen, 517
DB-Unterstützung, 535
Dokumentvalidierung, 524
Eingabehilfen, 518
Erweiterungsfunktionen, 2236
Funktionen für IBM DB2, 535
in Textansicht bearbeiten, 514
intelligente Editierfeatures, 520
Syntaxfärbung, 519
Variablen an XQuery-Dokument übergeben, 1414
zur Abfrage von XML-Datenbanken, 535
XQuery Debugger,
siehe XSLT/XQuery Debugger, 540
XQuery Profiler,
Ergebnisdiagramme, 579
XQuery Update, 525
PUL, 529
Vorschau, 529
XQuery Update Facility, 528
XQuery Update in XMLSpy, 528
XQuery Update-Optionen, 1636
XQuery-Ausdrücke für JSON, 772
XQuery-Ausdrücke in der JSON-Grid-Ansicht, 709
XQuery-Ausdrücke in der XML-Grid-Ansicht, 317, 319

© 2016-2022 Altova GmbH

XQuery-Ausführung, 525, 1417
XQuery-Dateien,
Dateierweiterungen in XMLSpy festlegen, 517
XQuery-Dokumente,
Ausführungsdauer analysieren, 567, 1420
XQuery-Optionen, 1636
XSD 1.0 und 1.1,
Bearbeitungsmodi, 173
XSD-Modus, 173
XSD-Validierung, 239
xsi:type,
Verwendung, 85
XSL,
siehe XSLT, 1422
XSL/Query Menü, 1409
xsl:call-template, 505
XSL:FO,
Transformation, 1412
und XSLT-Transformationen, 501
xsl:param, 505
xsl:with-param, 505
XSL-Geschwindigkeitsoptimierung, 512, 1412
XSL-Geschwindigkeitsoptimierungsoptionen, 1636
XSLT, 1505
Autokomplettierung in der Textansicht, 499
Dokumente, 499
Eingabehilfen, 499
Erweiterungsfunktionen, 2236
Funktionalität in XMLSpy, 498
in XMLSpy ändern, 107
Prozessor, 1633
Transformationen in XMLSpy, 501
und Batch-Transformationen, 501
Validierung, 499
XSLT Debugger,
aktuelle Execution Nodes einblenden, 1424
aussteigen, 1424
Breakpoint einfügen/löschen, 1424
Breakpoints Ein/Aus, 1425
Breakpoints/Tracepoints (Dialogfeld), 1426
Debug-Fenster, 1427
Debugger erneut starten, 1423
Debugger starten, 1422
Debugger stoppen, 1423
Debuggersitzung beenden, 1423
einsteigen, 1423
Einstellungen, 1427
in Visual Studio .NET, 1146
Tracepoint aktivieren/deaktivieren, 1425
Tracepoint einfügen/entfernen, 1425
überspringen, 1424
XSLT Profiler,
Ergebnisdiagramme, 579
XSLT Stylesheet,
öffnen, 1422
zu XML-Dokument zuweisen, 1420, 1421
XSLT Stylesheet für FO,
zu XML-Dokument zuweisen, 1421
XSLT Stylesheets,
Ausführung analysieren, 567, 1420
XSLT/XQuery debuggen,
verwendete Dateien, 540
XSLT/XQuery Debugger, 541
Beschreibung der Benutzeroberfläche, 541
Beschreibung der Funktionsweise, 541
Einstellungen, 546
Fenster "Call Stack", 553
Fenster "Info", 555
Fenster "Meldungen", 553
Fenster "Templates", 554
Fenster "Variablen", 552
Fenster "XPath-Watch", 552
Fenster Ablaufverfolgung, 555
Funktionen und Verwendung, 540
Informationsfenster, 550
keyboard shortcuts, 566
Kontextfenster, 551
Sitzung starten, 548
Symbolleisten-Schaltflächen, 543
Tracepoints Verwendung, 560
Verwendung von Breakpoints, 557
XSLT/XQuery Profiler, 567
XSLT-Dokumente,
mit XSL-Übersicht bearbeiten und verwalten, 504
und Info-Fenster, 508
XSLT-Dokumentstruktur, 125
XSLT-Funktionen,
im Fenster XSL-Übersicht, 505
XSLT-Parameter,
über Schnittstelle an Stylesheet übergeben, 1414
XSL-Transformation,
siehe XSLT, 105
XSLT-Transformation, 1411, 1412
in FO, 1412
in PDF, 1412
in XMLSpy, 106

© 2016-2022 Altova GmbH

XSLT-Transformation, 1411, 1412
Tutorial, 105
XSLT-Datei zuweisen, 105
XSLT-Vorlagen,
im Fenster XSL-Übersicht, 505
XSL-Übersicht, 504
XSL-Übersicht (Fenster), 125, 505
XULE (Fenster), 132
für die interaktive Abfrage von XBRL-Instanzen, 941
XULE ausführen, 1552
XULE für XBRL, 938
XULE-Ausführung,
Anleitung, 945
Optionen, 945
XULE-Dokumente,
Autokomplettierung, 939
Bearbeitungsunterstützung für, 939
und damit konforme Dateitypen, 939
und Taxonomie für die Autokomplettierung, 939
und XMLSpy-Projekte, 939
Validierung von, 939
XULE-Optionen, 1644
XULE-Verarbeitung, 1552

Z

Zeichen,
Position, 1505
Zeichensatz,
Kodierung, 1608
Zeile,
anhängen (in Authentic-Ansicht), 1437
einfügen (in Authentic-Ansicht), 1437
Gehe zu, 1505
kopieren (in Authentic-Ansicht), 1437
löschen, 1438
löschen (in Authentic-Ansicht), 1437
nach oben/nach unten verschieben, 1438
Zeile in Authentic-Ansicht löschen, 1438
Zeile in Authentic-Ansicht verschieben, 1438
Zeilenendezeichen, 1506
Zeilenlänge,
Zeilenumbruch in Textansicht, 1505
Zeilenleiste, 1506
Zeilennummerierung in der Textansicht, 148
Zeilenschaltung,
siehe Eingabetaste, 663
Zeilenumbruch, 1505
aktivieren/dekativieren, 1505
Ein/Aus, 1505
Zeilenumbruch in der Textansicht, 145
Zeilenumbrüche, 1603
Zertifikate, 425
Zertifikatspeicher, 425
zip64mode,
in der build.xml-Datei aktivieren, 1253
ZIP-Dateien, 955
in der Archivansicht-Ansicht bearbeiten, 961
in der Archivansicht-Ansicht erstellen, 961
siehe Archivansicht-Ansicht, 284
Zoomen in der Textansicht, 148
Zuletzt verwendete Dateien,
Liste, 1296
Zurück,
in der Schema-Ansicht, 251
Zurücksetzen,
Kürzel, 1589
Menübefehle, 1593
Symbolleiste & Menübefehle, 1586
Zuweisen,
Kürzel zu Befehl, 1589

© 2016-2022 Altova GmbH