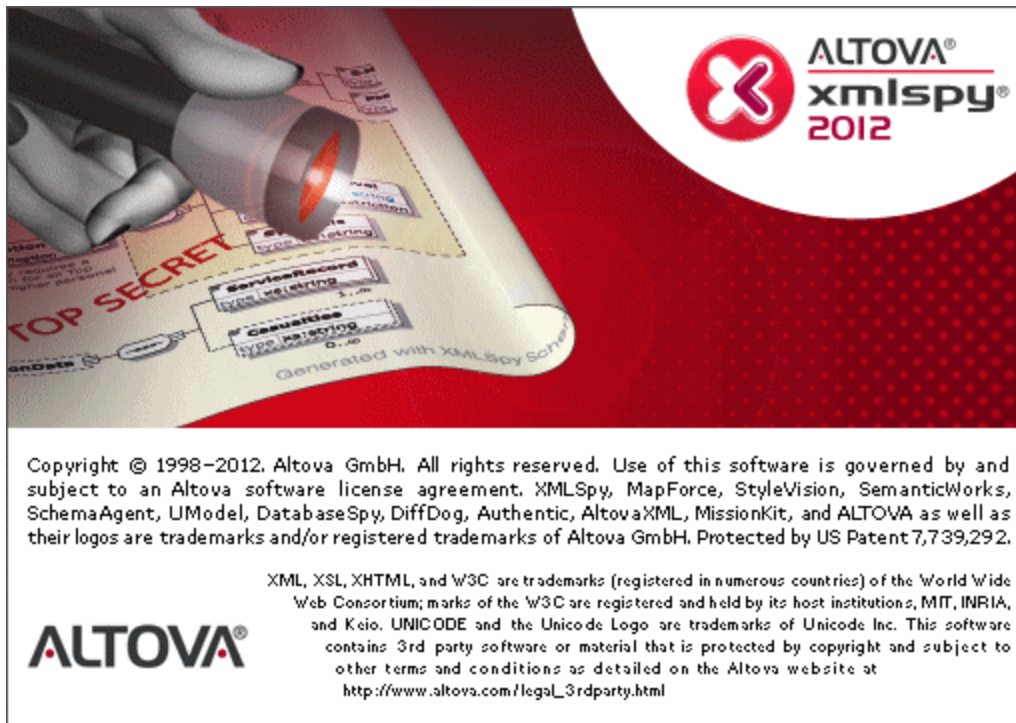


Benutzerhandbuch und Referenz für Programmierer



Altova XMLSpy 2012 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: 2012

© 2012 Altova GmbH

Inhaltsverzeichnis

Willkommen bei XMLSpy	3

Benutzerhandbuch	6

1	Neue Funktionen	8
2	Einführung	10
2.1	Die grafische Benutzeroberfläche	11
2.1.1	Hauptfenster	12
2.1.2	Projektfenster	13
2.1.3	Info-Fenster	15
2.1.4	Eingabehilfen	16
2.1.5	Ausgabefenster "Meldungen"	16
2.1.6	Ausgabefenster "XPath"	17
2.1.7	Ausgabefenster "XSL-Übersicht"	21
2.1.8	Ausgabefenster "In Dateien suchen"	22
2.1.9	Ausgabefenster "In Schemas suchen"	23
2.1.10	Menüleiste, Symbolleisten und Statusleiste	24
2.2	Die Applikationsumgebung	26
2.2.1	Einstellungen, Anpassung und Menüs	26
2.2.2	Tutorials, Projekte, Beispiele	28
2.2.3	XMLSpy Features, Hilfe und Altova-Produkte	29
3	XMLSpy Tutorial	30
3.1	XMLSpy-Benutzeroberfläche	31
3.2	Erstellen eines einfachen XML-Schemas	32
3.2.1	Anlegen einer neuen XML-Schema-Datei	32
3.2.2	Definieren von Namespaces	35
3.2.3	Definieren eines Content Model	36
3.2.4	Hinzufügen von Elementen mit Drag & Drop	41
3.2.5	Konfigurieren der Content Model-Ansicht	42

3.2.6	Fertigstellen des Schemas.....	44
3.3	XML-Schemas: komplexere Vorgänge	48
3.3.1	Arbeiten mit ComplexTypes und SimpleTypes.....	48
3.3.2	Referenzieren von globalen Elementen.....	55
3.3.3	Attribute und Attribut-Enumerationen.....	57
3.4	XML-Schemas: XMLSpy-Funktionen	60
3.4.1	Schemanavigation.....	60
3.4.2	Schema-Dokumentation.....	62
3.5	XML Dokumente	67
3.5.1	Erstellen eines neuen XML-Dokuments	67
3.5.2	Festlegen des Typs eines Elements.....	69
3.5.3	Eingabe von Daten in der Grid-Ansicht.....	71
3.5.4	Eingabe von Daten in der Textansicht.....	71
3.5.5	Validierung des Dokuments.....	76
3.5.6	Hinzufügen von Elementen und Attributen.....	80
3.5.7	Editieren in der Datenbank/Tabellen-Ansicht.....	82
3.5.8	Ändern des Schemas.....	86
3.6	XSLT-Transformationen	89
3.6.1	Zuweisen einer XSLT-Datei.....	89
3.6.2	Transformieren der XML-Datei.....	90
3.6.3	Ändern der XSL-Datei.....	91
3.7	Projektverwaltung	93
3.7.1	Vorteile von Projekten.....	93
3.7.2	Erstellen eines Projekts.....	93
3.8	Fertig!	95

4 Bearbeitungsansichten 96

4.1	Textansicht	97
4.1.1	Formatieren in der Textansicht.....	98
4.1.2	Anzeige des Dokuments.....	100
4.1.3	Bearbeiten in der Textansicht.....	102
4.1.4	Eingabehilfen in der Textansicht.....	105
4.2	Grid-Ansicht	107
4.2.1	Bearbeiten in der Grid-Ansicht.....	108
4.2.2	Tabellen der Grid-Ansicht.....	109
4.2.3	Eingabehilfen in der Grid-Ansicht.....	113
4.3	Schema-Ansicht	116
4.3.1	Schema-Übersicht.....	117
4.3.2	Content Model-Ansicht.....	121
4.3.3	Modifizieren des Base Types.....	133
4.3.4	Eingabehilfen in der Schema-Ansicht.....	134
4.3.5	Identity Constraints.....	138
4.3.6	Smart Restrictions.....	143
4.3.7	xml:base, xml:id, xml:lang, xml:space.....	147

4.3.8	Zurück und Vorwärts: Navigieren von einer Position zu einer anderen	149
4.4	Authentic-Ansicht	151
4.4.1	Übersicht über die Benutzeroberfläche	151
4.4.2	Authentic-Ansicht Symbolleistenschaltflächen	152
4.4.3	Authentic-Ansicht Hauptfenster	155
4.4.4	Authentic-Ansicht Eingabehilfen	157
4.4.5	Authentic-Ansicht Kontextmenüs	161
4.5	Browser-Ansicht	164
5	XML	165
5.1	Erstellen, Öffnen und Speichern von XML-Dokumenten	166
5.2	Zuweisen von Schemas und Validierung	168
5.3	Bearbeiten von XML in der Textansicht	170
5.4	Bearbeiten von XML in der Grid-Ansicht	172
5.5	Bearbeiten von XML in der Authentic-Ansicht	175
5.6	Eingabehilfen für XML-Dokumente	177
5.7	Verarbeitung mit XSLT und XQuery	179
5.8	Diagramme	181
5.8.1	Diagrammbeispiel: Einfach	184
5.9	Zusätzliche Funktionen	187
6	DTDs und XML Schemas	189
6.1	DTDs	190
6.2	XML-Schemas	192
6.3	Schemamodule	193
6.4	Kataloge in XMLSpy	197
6.5	Arbeiten mit SchemaAgent	201
6.5.1	Herstellen einer Verbindung mit SchemaAgent Server	202
6.5.2	Öffnen von Schemas aus dem Suchpfad	204
6.5.3	Verwenden von IIR-Beziehungen	205
6.5.4	Anzeigen von Schemas in SchemaAgent	209
6.5.5	SchemaAgent Validierung	210
6.6	In Schemas suchen	212
6.6.1	Suchbegriff	213
6.6.2	Komponenten	215
6.6.3	Eigenschaften	217
6.6.4	Bereich	220
6.6.5	Die Befehle "Suchen" und "Ersetzen"	221
6.6.6	Ergebnisse und Informationen	222
6.6.7	Suchen und Umbenennen von globalen Komponenten	223

7	XSLT und XQuery	225
7.1	XSLT	226
7.1.1	XSLT-Dokumente	226
7.1.2	XSLT-Verarbeitung	228
7.1.3	XSL-Übersicht	230
	– Das Fenster "XSL-Übersicht"	231
	– Das Fenster "Info"	234
7.2	XQuery	237
7.2.1	XQuery-Dokumente	238
7.2.2	XQuery Eingabehilfen	239
7.2.3	XQuery Syntaxfärbung	239
7.2.4	Intelligentes Editieren von XQuery	240
7.2.5	Validierung und Ausführung von XQuery-Dokumenten	242
7.2.6	XQuery und XML-Datenbanken	244
7.3	XSLT und XQuery Debugger	248
7.3.1	Funktionsweise und Benutzeroberfläche	249
7.3.2	Befehle und Symbolleistenflächen	251
7.3.3	XSLT/XQuery Einstellungen	254
7.3.4	Starten einer Debugger-Sitzung	255
7.3.5	Informationsfenster	257
	– Kontext	258
	– Variablen	259
	– XPath-Watch	259
	– Call Stack	260
	– Meldungen	260
	– Templates	261
	– Info	261
	– Trace	262
	– Anordnen der Informationsfenster	262
7.3.6	Breakpoints	263
7.3.7	Tracepoints	266
8	Authentic	272
8.1	Tutorial Authentic-Ansicht	274
8.1.1	Öffnen eines XML-Dokuments in der Authentic-Ansicht	275
8.1.2	Die Benutzeroberfläche der Authentic-Ansicht	276
8.1.3	Node-Operationen	279
8.1.4	Eingabe von Daten in der Authentic-Ansicht	282
8.1.5	Eingabe von Attributwerten	284
8.1.6	Hinzufügen von Entities	285
8.1.7	Drucken des Dokuments	286

8.2	Bearbeitung in der Authentic-Ansicht	288
8.2.1	Grundlegendes zur Bearbeitung	288
8.2.2	Tabellen in der Authentic-Ansicht	293
	– SPS-Tabellen	293
	– CALS/HTML-Tabellen	294
	– Schaltflächen zum Bearbeiten von CALS/HTML-Tabellen	298
8.2.3	Bearbeiten einer DB	300
	– Navigieren in einer DB-Tabelle	301
	– DB-Abfragen	301
	– Ändern einer DB-Tabelle	306
8.2.4	Arbeiten mit Datumsangaben	307
	– Datumswahl	308
	– Texteintrag	308
8.2.5	Definieren von Entities	309
8.2.6	XML-Signaturen	311
8.2.7	Grafiken in der Authentic-Ansicht	312
8.2.8	Verwendung von Tasten in der Authentic-Ansicht	313
8.3	Authentic Skripterstellung	314
9	HTML, CSS, JSON	316
9.1	HTML	317
9.2	CSS	319
9.3	JSON	324
10	Office Open XML, ZIP, EPUB	327
10.1	EPUB-Dateien	329
11	Datenbanken	333
11.1	Herstellen einer Verbindung zu einer Datenquelle	335
11.1.1	Verbindungsassistent	336
11.1.2	Bestehende Verbindungen+	337
11.1.3	ADO-Verbindungen	338
11.1.4	ODBC-Verbindungen	342
11.1.5	JDBC-Verbindungen	345
11.1.6	Globale Ressourcen	348
11.2	Unterstützte Datenbanken	350
12	Globale Altova-Ressourcen	351
12.1	Definieren globaler Ressourcen	352
12.1.1	Dateien	354

12.1.2	Ordner	356
12.1.3	Datenbanken.....	357
12.1.4	Kopieren von Konfigurationen.....	359
12.2	Verwendung globaler Ressourcen	360
12.2.1	Zuweisen von Dateien und Ordnern.....	360
12.2.2	Zuweisen von Datenbanken.....	363
12.2.3	Wechseln von Konfigurationen.....	364
13	Projekte	366
13.1	Erstellen und Bearbeiten von Projekten	367
13.2	Verwenden von Projekten	371
14	Datei-/Verzeichnisvergleiche	373
14.1	Dateivergleiche	374
14.2	Verzeichnisvergleiche	375
15	Versionskontrolle	376
15.1	Unterstützte Versionskontrollsysteme	378
15.2	Installation von Versionskontrollsystemen	383
15.3	Versionskontrolle und Vergleich mit Altova DiffDog	390
16	XMLSpy in MS Visual Studio	395
16.1	Installieren des XMLSpy Plugin	396
16.2	Unterschiede zur XMLSpy-Standalone-Version	398
16.3	Die XMLSpy Debugger in Visual Studio	400
16.4	Bekannte Probleme	401
17	XMLSpy in Eclipse	402
17.1	Installation des XMLSpy Plugin für Eclipse	403
17.2	XMLSpy-Einstiegspunkte in Eclipse	407
17.3	Die XMLSpy Debugger-Perspektiven	410
18	Benutzerreferenz	411
18.1	Menü "Datei"	412
18.1.1	Neu	412
18.1.2	Öffnen	416
18.1.3	Neu laden.....	421

18.1.4	Kodierung.....	421
18.1.5	Schließen, Alle schließen, Alle inaktiven schließen.....	422
18.1.6	Speichern, Speichern unter, Alles speichern.....	422
18.1.7	Als Mail senden.....	427
18.1.8	Drucken.....	428
18.1.9	Druckvorschau, Druckereinrichtung.....	429
18.1.10	Zuletzt verwendete Dateien, Beenden.....	430
18.2	Menü "Bearbeiten"	431
18.2.1	Rückgängig, Wiederherstellen.....	432
18.2.2	Ausschneiden, Kopieren, Einfügen, Löschen.....	432
18.2.3	Als XML-Text kopieren.....	432
18.2.4	Als strukturierten Text kopieren.....	433
18.2.5	XPath kopieren.....	435
18.2.6	XPointer kopieren.....	435
18.2.7	Einfügen.....	436
18.2.8	Pretty-Print.....	439
18.2.9	Alles markieren.....	440
18.2.10	Suchen, Weitersuchen.....	440
18.2.11	Ersetzen.....	444
18.2.12	In Dateien suchen.....	445
18.2.13	Lesezeichen-Befehle.....	447
18.2.14	Ein-/Auskommentieren.....	448
18.3	Menü "Projekt"	449
18.3.1	Neues Projekt.....	451
18.3.2	Projekt öffnen.....	452
18.3.3	Projekt neu laden.....	452
18.3.4	Projekt schließen.....	452
18.3.5	Projekt speichern, Projekt speichern unter.....	452
18.3.6	Versionskontrolle.....	452
	– Aus Versionskontrolle öffnen.....	453
	– Versionskontrolle aktivieren.....	454
	– Aktuellste Version holen.....	455
	– Abrufen.....	455
	– Ordner abrufen.....	456
	– Auschecken.....	457
	– Einchecken.....	458
	– Auschecken rückgängig.....	459
	– Zu Versionskontrolle hinzufügen.....	460
	– Von Versionskontrolle ausgliedern.....	461
	– Aus Versionskontrolle freigeben.....	461
	– Verlauf anzeigen.....	462
	– Unterschiede anzeigen.....	463
	– Eigenschaften anzeigen.....	465
	– Status aktualisieren.....	465
	– Versionskontrollmanager.....	465

– Versionskontrolle wechseln	465
18.3.7 Dateien zu Projekt hinzufügen.....	466
18.3.8 Globale Ressource zu Projekt hinzufügen.....	466
18.3.9 URL zu Projekt hinzufügen.....	466
18.3.10 Aktive Datei zu Projekt hinzufügen.....	467
18.3.11 Aktive und verwandte Dateien zu Projekt hinzufügen.....	467
18.3.12 Projektordner zu Projekt hinzufügen.....	467
18.3.13 Externes Verzeichnis zum Projekt hinzufügen.....	467
18.3.14 Externes Webverzeichnis zum Projekt hinzufügen.....	471
18.3.15 Skript-Einstellungen.....	474
18.3.16 Projekteigenschaften.....	475
18.3.17 Zuletzt verwendete Projekte.....	477
18.4 Menü "XML"	478
18.4.1 Einfügen.....	479
– Attribut einfügen.....	480
– Element einfügen.....	480
– Text einfügen.....	480
– CDATA einfügen.....	481
– Comment einfügen.....	481
– XML einfügen.....	481
– Processing Instruction einfügen.....	481
– XInclude einfügen.....	481
– DOCTYPE einfügen.....	484
– ExternalID einfügen.....	484
– Element einfügen.....	485
– ATTLIST einfügen.....	485
– ENTITY einfügen.....	485
– NOTATION einfügen.....	485
– Externe kodierte Datei einfügen.....	485
18.4.2 Anhängen.....	486
– Attribut anhängen.....	487
– Element anhängen.....	487
– Text anhängen.....	487
– CDATA anhängen.....	488
– Comment anhängen.....	488
– XML anhängen.....	488
– Processing Instruction anhängen.....	488
– XInclude anhängen.....	488
– DOCTYPE anhängen.....	491
– ExternalID anhängen.....	491
– Element anhängen.....	492
– ATTLIST anhängen.....	492
– Entity anhängen.....	492
– NOTATION anhängen.....	492
– Externe kodierte Datei anhängen.....	492

18.4.3	Child hinzufügen.....	493
	– Child-Attribut hinzufügen.....	494
	– Child-Element hinzufügen.....	494
	– Child-Text hinzufügen.....	494
	– Child CDATA hinzufügen.....	494
	– Child Comment hinzufügen.....	494
	– Child XML hinzufügen.....	494
	– Child Processing Instruction hinzufügen.....	495
	– Child XInclude anhängen.....	495
	– Child DOCTYPE hinzufügen.....	497
	– Child ExternalID hinzufügen.....	497
	– Child Element hinzufügen.....	497
	– Child ATTLIST hinzufügen.....	498
	– Child ENTITY hinzufügen.....	498
	– Child NOTATION hinzufügen.....	498
	– Externe kodierte Child-Datei hinzufügen.....	498
18.4.4	Konvertieren in.....	499
	– In Attribut konvertieren.....	499
	– In Element konvertieren.....	499
	– In Text konvertieren.....	500
	– In CDATA konvertieren.....	500
	– In Comment konvertieren.....	500
	– In XML konvertieren.....	500
	– In Processing Instruction konvertieren.....	500
	– In DOCTYPE konvertieren.....	500
	– In ExternalID konvertieren.....	500
	– In Element konvertieren.....	500
	– In ATTLIST konvertieren.....	500
	– In ENTITY konvertieren.....	501
	– In NOTATION konvertieren.....	501
18.4.5	Tabelle.....	501
	– Als Tabelle anzeigen.....	501
	– Zeile einfügen.....	502
	– Zeile anhängen.....	502
	– Aufsteigend sortieren.....	503
	– Absteigend sortieren.....	503
18.4.6	Höher stufen.....	503
18.4.7	Tiefer stufen.....	504
18.4.8	In Element einbinden.....	504
18.4.9	XPath auswerten.....	504
18.4.10	Wohlgeformtheit prüfen.....	505
18.4.11	XML validieren.....	506
18.4.12	Eingabehilfen aktualisieren.....	509
18.4.13	Namespace Präfix.....	509
18.5	Menü "DTD/Schema"	510

18.5.1	DTD zuweisen.....	510
18.5.2	Schema zuweisen.....	511
18.5.3	Weitere DTD einfügen.....	511
18.5.4	Gehe zu DTD.....	511
18.5.5	Gehe zu Schema.....	511
18.5.6	Gehe zu Definition.....	512
18.5.7	DTD/Schema generieren.....	512
18.5.8	DTD/Schema konvertieren.....	514
18.5.9	Im UML konvertieren.....	517
18.5.10	XML anhand von DB, Excel, EDI in MapForce generieren.....	518
18.5.11	HTML/PDF/Word-Ausgabe mit StyleVision erstellen.....	518
18.5.12	XML-Beispieldatei erzeugen.....	518
18.5.13	Cache-Speicher leeren.....	520
18.6	Menü "Schema-Design"	521
18.6.1	Schema-Einstellungen.....	521
18.6.2	Diagramm speichern.....	523
18.6.3	Schema-Dokumentation generieren.....	523
	– Dokumentationsoptionen.....	524
	– Benutzerdefiniertes Design.....	526
18.6.4	Ansicht konfigurieren.....	528
18.6.5	Vergrößern/Verkleinern.....	531
18.6.6	Globale Elemente anzeigen.....	531
18.6.7	Diagramm anzeigen.....	532
18.6.8	Schemaerweiterungen für Datenbanken.....	532
	– Oracle Schema Extensions aktivieren.....	532
	– Oracle Schema-Einstellungen.....	533
	– Microsoft SQL Server Schema Extensions aktivieren.....	533
	– Named Schema Relationships.....	534
	– Unnamed Element Relationships.....	535
18.6.9	Verbindung zu SchemaAgent Server herstellen.....	535
18.6.10	Verbindung mit SchemaAgent Server trennen.....	536
18.6.11	In SchemaAgent anzeigen.....	536
18.6.12	SchemaAgent-Validierung.....	537
18.6.13	Schemamodul erstellen.....	537
18.6.14	Schema nivellieren.....	538
18.7	Menü "XSL/XQuery"	540
18.7.1	XSL-Transformation.....	541
18.7.2	XSL-FO-Transformation.....	542
18.7.3	XSL-Parameter / XQuery-Variablen.....	544
18.7.4	XQuery-Ausführung.....	547
18.7.5	XSL zuweisen.....	547
18.7.6	XSL-FO zuweisen.....	548
18.7.7	XML-Beispieldatei zuweisen.....	548
18.7.8	Gehe zu XSL.....	548
18.7.9	Debugger starten.....	549

18.7.10	Debugger anhalten.....	549
18.7.11	Debugger neu starten.....	549
18.7.12	Debugger-Sitzung beenden.....	549
18.7.13	Einsteigen.....	549
18.7.14	Aussteigen.....	550
18.7.15	Überspringen.....	550
18.7.16	Aktuellen Execution Node einblenden.....	550
18.7.17	Breakpoint einfügen/löschen.....	550
18.7.18	Tracepoint einfügen/löschen.....	550
18.7.19	Breakpoint aktivieren/deaktivieren.....	551
18.7.20	Tracepoint aktivieren/deaktivieren.....	551
18.7.21	Breakpoints/Tracepoints.....	551
18.7.22	Debug-Fenster.....	552
18.7.23	XSLT/XQuery-Einstellungen.....	552
18.8	Menü "Authentic".....	554
18.8.1	Neues Dokument.....	554
18.8.2	Datenbankdaten bearbeiten.....	555
18.8.3	StyleVision Stylesheet zuweisen.....	556
18.8.4	Neue Zeile mit XML-Daten für Bearbeitung auswählen.....	557
18.8.5	XML-Signatur.....	558
18.8.6	XML Entities definieren.....	559
18.8.7	Markup-Code anzeigen.....	561
18.8.8	RichEdit.....	561
18.8.9	Zeile anhängen/einfügen/kopieren/löschen.....	562
18.8.10	Zeile nach oben/unten.....	562
18.8.11	Vertrauenswürdige Pfade.....	563
18.9	Menü "DB".....	564
18.9.1	Herstellen einer Verbindung zu einer Datenquelle.....	564
	– Verbindungsassistent.....	566
	– Bestehende Verbindungen.....	568
	– ADO-Verbindungen.....	569
	– ODBC-Verbindungen.....	573
	– JDBC-Verbindungen.....	576
	– Globale Ressourcen.....	579
18.9.2	Datenbank abfragen.....	580
	– Datenquellen.....	582
	– Browser-Fenster: Anzeigen der DB-Objekte.....	584
	– Abfragefenster: Beschreibung und Funktionen.....	587
	– Abfragefenster: Arbeiten mit Abfragen.....	591
	– Ergebnisse und Meldungen.....	592
18.9.3	IBM DB2.....	595
	– XML-Schemas verwalten.....	595
	– XML-Schema zuweisen.....	598
18.9.4	SQL Server.....	600
	– XML-Schemas verwalten.....	600

18.9.5	Oracle XML DB.....	604
	– Oracle XML-Dokumente durchsuchen.....	604
18.10	Menü "Konvertieren"	606
18.10.1	Textdatei importieren.....	606
18.10.2	Datenbankdaten importieren.....	609
18.10.3	MS Word Dokument Importieren.....	614
18.10.4	XML-Schema von DB-Struktur erzeugen.....	614
18.10.5	DB-Import auf Basis von XML-Schema.....	618
18.10.6	DB-Struktur von XML-Schema erzeugen.....	619
18.10.7	Export in Textdateien.....	621
18.10.8	Export in Datenbank.....	624
18.10.9	XML von/in JSON konvertieren.....	626
18.11	Menü "Ansicht"	628
18.11.1	Text-Ansicht.....	628
18.11.2	Grid Ansicht.....	628
18.11.3	Schema-Ansicht.....	629
18.11.4	Authentic-Ansicht.....	629
18.11.5	Browser-Ansicht.....	629
18.11.6	Erweitern.....	629
18.11.7	Reduzieren.....	630
18.11.8	Alles erweitern.....	630
18.11.9	Unselektierte reduzieren.....	630
18.11.10	Optimale Breite.....	630
18.11.11	Zeilenumbruch.....	630
18.11.12	Gehe zu Zeile/Zeichen.....	631
18.11.13	Gehe zu Datei.....	631
18.11.14	Einstellungen für Textansicht.....	631
18.12	Menü "Browser"	633
18.12.1	Zurück	633
18.12.2	Vorwärts.....	633
18.12.3	Abbrechen.....	633
18.12.4	Aktualisieren.....	633
18.12.5	Schriftgröße.....	633
18.12.6	Neues Fenster.....	633
18.13	Menü "Extras"	635
18.13.1	Rechtschreibung.....	635
18.13.2	Rechtschreibung - Optionen.....	639
18.13.3	Skript-Editor.....	643
18.13.4	Makros.....	644
18.13.5	Vergleiche.....	644
	– Geöffnete Datei vergleichen mit.....	644
	– Verzeichnisse vergleichen.....	646
	– Vergleichsoptionen.....	649
18.13.6	Globale Ressourcen.....	651
18.13.7	Aktive Konfiguration.....	652

18.13.8	Anpassen	653
–	Befehle	653
–	Symbolleisten	654
–	Tastatur	655
–	Menü	660
–	Makros	662
–	Plug-Ins	663
–	Optionen	664
–	Kontextmenü 'Anpassen'	664
18.13.9	Symbolleisten und Fenster zurücksetzen	666
18.13.10	Optionen	667
–	Datei	667
–	Dateiarten	668
–	Bearbeiten	670
–	Ansicht	671
–	Grid-Schriftarten	673
–	Schemaschriftarten	673
–	Textschriftarten	674
–	Farben	675
–	Kodierung	677
–	XSL	677
–	Skripting-Umgebung	679
–	Versionskontrolle	680
18.14	Menü "Fenster"	682
18.14.1	Überlappend	682
18.14.2	Horizontal anordnen	682
18.14.3	Vertikal anordnen	682
18.14.4	Projekt-Fenster	682
18.14.5	Info-Fenster	683
18.14.6	Eingabehilfen	683
18.14.7	Ausgabefenster	683
18.14.8	Projekt- und Eingabehilfen	683
18.14.9	Alle ein/aus	683
18.14.10	Liste aller offenen Fenster	684
18.15	Menü "Hilfe"	685
18.15.1	Inhaltsverzeichnis	685
18.15.2	Index	685
18.15.3	Suche	686
18.15.4	Tastaturzuordnungen	686
18.15.5	Aktivierung, Bestellformular, Registrieren, Updates	687
18.15.6	Support Center, Fragen und Antworten, Komponenten downloaden	688
18.15.7	Internet	688
18.15.8	Über	688
18.16	Befehlszeile	689

1	Skripterstellung	694
1.1	Übersicht	696
1.1.1	Skripting-Projekte in XMLSpy.....	697
1.1.2	Die Benutzeroberfläche des Skript-Editors.....	698
1.1.3	Komponenten eines Skripting-Projekts.....	702
1.2	Erstellen eines Skripting-Projekts	704
1.3	Globale Deklarationen	706
1.4	Formulare	708
1.4.1	Erstellung eines neuen Formulars.....	708
1.4.2	Formulardesign und Formularobjekte.....	709
1.4.3	Formular-Events.....	711
1.5	Events	714
1.6	Makros	717
1.6.1	Erstellen und Bearbeiten eines Makros.....	717
1.6.2	Ausführen eines Makros.....	719
1.6.3	Debuggen eines Makros	721
1.7	Anmerkungen zur Programmierung	723
1.7.1	Vordefinierte Befehle.....	724
	– Verwendung von Formularen und Befehle.....	731
1.8	Migration zum Skript-Editor 2010 und höheren Versionen	732
2	Plugins	735
2.1	Registrierung von IDE Plug-Ins	736
2.2	ActiveX Controls	737
2.3	Konfigurations-XML	738
2.4	ATL-Beispieldateien	741
2.4.1	Schnittstellenbeschreibung (IDL).....	741
2.4.2	Klassendefinition.....	743
2.4.3	Implementierung.....	743
2.5	IXMLSpyPlugIn	746
2.5.1	OnCommand	746
2.5.2	OnUpdateCommand.....	747
2.5.3	OnEvent.....	747
2.5.4	GetUIModifications.....	750
2.5.5	GetDescription.....	750

3	Applikations-API	752
3.1	Übersicht	754
3.1.1	Objektmodell.....	754
3.1.2	Programmiersprachen.....	755
	– JScript.....	757
	<i>Applikation starten</i>	757
	<i>Einfacher Dokumentaufruf</i>	758
	<i>Iteration</i>	759
	<i>Fehlerbehandlung</i>	761
	<i>Events</i>	762
	<i>Import und Export von Daten</i>	762
	<i>Import aus einer Datenbank</i>	765
	<i>Export in eine Datenbank</i>	767
	<i>Import aus einer Textdatei</i>	769
	<i>Export in eine Textdatei</i>	771
	<i>Beispiel: Bubble-Sortierung dynamischer Tabellen</i>	772
	– VBScript.....	774
	<i>Events</i>	774
	<i>Beispiel: Verwendung von Events</i>	774
	– C#	776
	<i>Hinzufügen einer Referenz zur Application API</i>	780
	<i>Starten und Beenden der Applikation</i>	781
	<i>Öffnen von Dokumenten</i>	782
	<i>Iterieren durch offene Dokumente</i>	783
	<i>Fehler und COM-Ausgabeparameter</i>	784
	<i>Events</i>	785
	– Java	785
	<i>Java-Beispielprojekt</i>	787
	<i>Starten und Beenden der Applikation</i>	790
	<i>Einfacher Dokumentzugriff</i>	791
	<i>Iterationen</i>	791
	<i>Verwendung von Out-Parametern</i>	792
	<i>Event Handler</i>	792
3.1.3	DOM und XML Data	793
3.1.4	Nicht mehr verwendete Zeilenoperationen in der Authentic-Ansicht	795
3.2	Schnittstellen	797
3.2.1	Application.....	797
	– Events.....	799
	<i>OnBeforeOpenDocument</i>	799
	<i>OnBeforeOpenProject</i>	799
	<i>OnDocumentOpened</i>	800
	<i>OnProjectOpened</i>	800

– ActiveDocument.....	801
– AddMacroMenuItem.....	801
– Application.....	801
– ClearMacroMenu.....	802
– CreateXMLSchemaFromDBStructure.....	802
– CurrentProject.....	802
– Dialogs.....	803
– Documents.....	803
– Edition.....	803
– FindInFiles.....	803
– GetDatabaseImportElementList.....	804
– GetDatabaseSettings.....	804
– GetDatabaseTables.....	805
– GetExportSettings.....	805
– GetTextImportElementList.....	806
– GetTextImportExportSettings.....	806
– ImportFromDatabase.....	807
– ImportFromSchema.....	808
– ImportFromText.....	808
– ImportFromWord.....	809
– IsAPISupported.....	809
– MajorVersion.....	810
– MinorVersion.....	810
– NewProject.....	810
– OpenProject.....	810
– Parent.....	811
– Quit.....	811
– ReloadSettings.....	812
– RunMacro.....	812
– ScriptingEnvironment.....	812
– ServicePackVersion.....	812
– ShowApplication.....	813
– ShowFindInFiles.....	813
– ShowForm.....	813
– Status.....	814
– URLDelete.....	814
– URLMakeDirectory.....	814
– Visible.....	815
– WarningNumber.....	815
– WarningText.....	815
3.2.2 AuthenticContextMenu.....	815
– CountItems.....	815
– DeleteItem.....	816
– GetItemText.....	816
– InsertItem.....	816

	– SetItemText.....	816
3.2.3	AuthenticDataTransfer	817
	– dropEffect.....	817
	– getData.....	818
	– ownDrag.....	818
	– type	818
3.2.4	AuthenticEventContext.....	818
	– EvaluateXPath.....	819
	– GetEventContextType.....	819
	– GetNormalizedTextValue.....	819
	– GetVariableValue.....	820
	– GetXMLNode.....	820
	– IsAvailable.....	820
	– SetVariableValue.....	820
3.2.5	AuthenticRange.....	821
	– AppendRow	823
	– Application.....	823
	– CanPerformAction.....	823
	– CanPerformActionWith.....	824
	– Clone.....	824
	– CollapsToBegin.....	825
	– CollapsToEnd.....	825
	– Copy.....	825
	– Cut	825
	– Delete.....	826
	– DeleteRow.....	826
	– DuplicateRow.....	827
	– EvaluateXPath.....	827
	– ExpandTo.....	828
	– FirstTextPosition.....	828
	– FirstXMLData.....	829
	– FirstXMLDataOffset.....	830
	– GetElementAttributeNames	831
	– GetElementAttributeValue.....	831
	– GetElementHierarchy.....	831
	– GetEntityNames.....	832
	– GetVariableValue.....	833
	– Goto	833
	– GotoNext.....	833
	– GotoNextCursorPosition.....	834
	– GotoPrevious.....	834
	– GotoPreviousCursorPosition.....	835
	– HasElementAttribute.....	835
	– InsertEntity.....	836
	– InsertRow.....	836

– IsCopyEnabled.....	837
– IsCutEnabled.....	837
– IsDeleteEnabled.....	837
– IsEmpty.....	838
– IsEqual.....	838
– IsFirstRow.....	838
– IsInDynamicTable.....	838
– IsLastRow.....	839
– IsPasteEnabled.....	839
– IsSelected.....	839
– IsTextStateApplied.....	839
– LastTextPosition.....	840
– LastXMLData.....	841
– LastXMLDataOffset.....	841
– MoveBegin.....	842
– MoveEnd.....	843
– MoveRowDown.....	843
– MoveRowUp.....	843
– Parent.....	844
– Paste.....	844
– PerformAction.....	844
– Select.....	845
– SelectNext.....	846
– SelectPrevious.....	846
– SetElementAttribute Value.....	847
– SetFromRange.....	848
– SetVariableValue.....	848
– Text.....	849
3.2.6 AuthenticView.....	849
– Events.....	850
<i>OnBeforeCopy</i>	850
<i>OnBeforeCut</i>	851
<i>OnBeforeDelete</i>	851
<i>OnBeforeDrop</i>	852
<i>OnBeforePaste</i>	853
<i>OnBeforeSave</i>	853
<i>OnDragOver</i>	853
<i>OnKeyboardEvent</i>	855
<i>OnLoad</i>	855
<i>OnMouseEvent</i>	856
<i>OnSelectionChanged</i>	856
<i>OnToolbarButtonClicked</i>	857
<i>OnToolbarButtonExecuted</i>	858
<i>OnUserAddedXMLNode</i>	859
– Application.....	859

– AsXMLString.....	859
– ContextMenu.....	860
– CreateXMLNode.....	860
– DisableAttributeEntryHelper.....	860
– DisableElementEntryHelper.....	860
– DisableEntityEntryHelper.....	861
– DocumentBegin.....	861
– DocumentEnd.....	861
– DoNotPerformStandardAction.....	861
– EvaluateXPath.....	862
– Event.....	862
– EventContext.....	862
– GetToolBarButtonState.....	862
– Goto.....	863
– IsRedoEnabled.....	864
– IsUndoEnabled.....	864
– MarkupVisibility.....	864
– Parent.....	864
– Print.....	865
– Redo.....	865
– Selection.....	865
– SetToolBarButtonState.....	866
– Undo.....	866
– UpdateXMLInstanceEntities.....	866
– WholeDocument.....	867
– XMLDataRoot.....	867
3.2.7 CodeGeneratorDlg.....	868
– Application.....	868
– CompatibilityMode.....	869
– CPPSettings_DOMType.....	869
– CPPSettings_GenerateVC6ProjectFile.....	869
– CPPSettings_GenerateGCCMakefile.....	869
– CPPSettings_GenerateVSProjectFile.....	870
– CPPSettings_LibraryType.....	870
– CPPSettings_UseMFC.....	870
– CSharpSettings_ProjectType.....	871
– OutputPath.....	871
– OutputPathDialogAction.....	871
– OutputResultDialogAction.....	871
– Parent.....	872
– ProgrammingLanguage.....	872
– PropertySheetDialogAction.....	872
– TemplateFileName.....	873
3.2.8 DatabaseConnection.....	873
– ADOConnection.....	874

– AsAttributes	874
– CommentIncluded	874
– CreateMissingTables	875
– CreateNew	875
– DatabaseKind	875
– DatabaseSchema	875
– ExcludeKeys	876
– File	876
– ForeignKeys	876
– ImportColumnsType	876
– IncludeEmptyElements	877
– NullReplacement	877
– NumberDateTimeFormat	877
– ODBCConnection	877
– PrimaryKeys	878
– SchemaExtensionType	878
– SchemaFormat	878
– SQLSelect	879
– TextFieldLen	879
– UniqueKeys	879
3.2.9 Dialogs	879
– Application	880
– CodeGeneratorDlg	880
– FileSelectionDlg	880
– Parent	881
– SchemaDocumentationDlg	881
– GenerateSampleXMLDlg	881
– DTDSchemaGeneratorDlg	881
– FindInFilesDlg	882
– WSDLDocumentationDlg	882
– WSDL20DocumentationDlg	882
– XBRLDocumentationDlg	883
3.2.10 Document	883
– Events	885
<i>OnBeforeSaveDocument</i>	885
<i>OnBeforeCloseDocument</i>	885
<i>OnBeforeValidate</i>	886
<i>OnCloseDocument</i>	887
<i>OnViewActivation</i>	887
– Application	887
– AssignDTD	888
– AssignSchema	888
– AssignXSL	888
– AssignXSLFO	889
– AsXMLString	889

- AuthenticView	889
- Close	890
- ConvertDTDOOrSchema	890
- ConvertDTDOOrSchemaEx	891
- ConvertToWSDL20	891
- CreateChild	892
- CreateDBStructureFromXMLSchema	892
- CreateSchemaDiagram	892
- CurrentViewMode	893
- DataRoot	893
- DocEditView	893
- Encoding	894
- EndChanges	894
- ExecuteXQuery	894
- ExportToDatabase	895
- ExportToText	895
- FullName	896
- GenerateDTDOOrSchema	897
- GenerateDTDOOrSchemaEx	897
- GenerateProgramCode	898
- GenerateSampleXML	898
- GenerateSchemaDocumentation	898
- GenerateWSDL20Documentation	899
- GenerateWSDLDocumentation	899
- GenerateXBRLDocumentation	900
- GetDBStructureList	900
- GetExportElementList	900
- GetPathName (nicht mehr verwendet)	901
- GridView	901
- IsModified	902
- IsValid	902
- IsWellFormed	903
- Name	904
- Parent	904
- Path	904
- RootElement	904
- Save	905
- SaveAs	905
- Saved	905
- SaveInString	906
- SaveToURL	906
- SetActiveDocument	906
- Document.SetEncoding (nicht mehr verwendet)	907
- SetExternalIsValid	908
- Document.SetPathName (nicht mehr verwendet)	908

	– StartChanges	908
	– Suggestions	909
	– SwitchViewMode	909
	– TextView	909
	– Title	909
	– TransformXSL	910
	– TransformXSLEx	910
	– TransformXSLFO	910
	– TreatXBRLInconsistencies AsErrors	910
	– UpdateViews	911
	– UpdateXMLData	911
3.2.11	Documents	911
	– Count	912
	– Item	912
	– NewAuthenticFile	912
	– NewFile	913
	– NewFileFromText	913
	– OpenAuthenticFile	914
	– OpenFile	914
	– OpenURL	914
	– OpenURLDialog	915
3.2.12	DTDSchemaGeneratorDlg	916
	– Application	916
	– AttributeTypeDefinition	916
	– DTDSchemaFormat	917
	– FrequentElements	917
	– GlobalAttributes	917
	– MaxEnumLength	917
	– MergeAllEqualNamed	917
	– OnlyStringEnums	918
	– OutputPath	918
	– OutputPathDialogAction	918
	– Parent	918
	– ResolveEntities	919
	– TypeDetection	919
	– ValueList	919
3.2.13	ElementList	919
	– Count	920
	– Item	920
	– RemoveElement	920
3.2.14	ElementListItem	920
	– ElementKind	921
	– FieldCount	921
	– Name	921
	– RecordCount	921

3.2.15	ExportSettings.....	921
	– CreateKeys.....	922
	– ElementList.....	922
	– EntitiesToText.....	922
	– ExportAllElements.....	923
	– ExportCompleteXML.....	923
	– FromAttributes.....	923
	– FromSingleSubElements.....	923
	– FromTextValues.....	923
	– IndependentPrimaryKey.....	924
	– Namespace.....	924
	– StartFromElement.....	924
	– SubLevelLimit.....	924
3.2.16	FileSelectionDlg.....	924
	– Application.....	925
	– DialogAction.....	925
	– FullName.....	925
	– Parent.....	926
3.2.17	FindInFilesDlg.....	926
	– AdvancedXMLSearch.....	926
	– Application.....	927
	– DoReplace.....	927
	– FileExtension.....	927
	– Find.....	927
	– IncludeSubfolders.....	928
	– MatchCase.....	928
	– MatchWholeWord.....	928
	– Parent.....	928
	– RegularExpression.....	928
	– Replace.....	929
	– ReplaceOnDisk.....	929
	– SearchInProjectFilesDoExternal.....	929
	– SearchLocation.....	929
	– ShowResult.....	929
	– StartFolder.....	930
	– XMLAttributeContents.....	930
	– XMLAttributeNames.....	930
	– XMLCDATA.....	930
	– XMLComments.....	930
	– XMLElementContents.....	931
	– XMLElementNames.....	931
	– XMLPI.....	931
	– XMLRest.....	931
3.2.18	FindInFilesResult.....	932
	– Application.....	932

	– Count.....	932
	– Document.....	932
	– Item	932
	– Parent.....	933
	– Path	933
3.2.19	FindInFilesResultMatch.....	933
	– Application.....	933
	– Length.....	934
	– Line	934
	– LineText.....	934
	– Parent.....	934
	– Position.....	934
	– Replaced.....	935
3.2.20	FindInFilesResults.....	935
	– Application.....	935
	– Count.....	935
	– Item	936
	– Parent.....	936
3.2.21	GenerateSampleXMLDlg.....	936
	– Application.....	936
	– Parent.....	937
	– NonMandatoryAttributes	937
	– NonMandatoryElements - obsolete.....	937
	– TakeFirstChoice - obsolete.....	937
	– RepeatCount.....	937
	– FillWithSampleData - obsolete.....	938
	– FillElementsWithSampleData	938
	– FillAttributesWithSampleData.....	938
	– ContentOfNillableElementsIsNonMandatory	938
	– TryToUseNonAbstractTypes.....	939
	– Optimization.....	939
	– SchemaOrDTDAssignment.....	939
	– LocalNameOfRootElement.....	939
	– NamespaceURIOfRootElement.....	939
	– OptionsDialogAction	940
3.2.22	GridView.....	940
	– Events.....	940
	<i>OnBeforeDrag</i>	940
	<i>OnBeforeDrop</i>	941
	<i>OnBeforeStartEditing</i>	941
	<i>OnEditingFinished</i>	942
	<i>OnFocusChanged</i>	942
	– CurrentFocus.....	943
	– Deselect.....	943
	– IsVisible.....	943

	– Select.....	943
	– SetFocus.....	943
3.2.23	SchemaDocumentationDlg.....	944
	– AllDetails.....	945
	– Application.....	945
	– CreateDiagramsFolder.....	945
	– DiagramFormat.....	946
	– EmbedCSSInHTML.....	946
	– EmbedDiagrams.....	946
	– GenerateRelativeLinks.....	947
	– IncludeAll.....	947
	– IncludeAttributeGroups.....	947
	– IncludeComplexTypes.....	947
	– IncludeGlobalAttributes.....	948
	– IncludeGlobalElements.....	948
	– IncludeGroups.....	948
	– IncludeIndex.....	949
	– IncludeLocalAttributes.....	949
	– IncludeLocalElements.....	949
	– IncludeRedefines.....	949
	– IncludeReferencedSchemas.....	950
	– IncludeSimpleTypes.....	950
	– MultipleOutputFiles.....	950
	– OptionsDialogAction.....	951
	– OutputFile.....	951
	– OutputFileDialogAction.....	951
	– OutputFormat.....	952
	– Parent.....	952
	– ShowAnnotations.....	952
	– ShowAttributes.....	952
	– ShowChildren.....	953
	– ShowDiagram.....	953
	– ShowEnumerations.....	953
	– ShowIdentityConstraints.....	954
	– ShowNamespace.....	954
	– ShowPatterns.....	954
	– ShowProgressBar.....	955
	– ShowProperties.....	955
	– ShowResult.....	955
	– ShowSingleFacets.....	955
	– ShowSourceCode.....	956
	– ShowType.....	956
	– ShowUsedBy.....	956
	– SPSFile.....	957
	– UseFixedDesign.....	957

3.2.24	SpyProject.....	957
	– CloseProject.....	957
	– ProjectFile.....	958
	– RootItems.....	958
	– SaveProject.....	958
	– SaveProjectAs.....	958
3.2.25	SpyProjectItem.....	959
	– ChildItems.....	959
	– FileExtensions.....	959
	– ItemType.....	959
	– Name.....	960
	– Open.....	960
	– ParentItem.....	960
	– Path.....	960
	– ValidateWith.....	960
	– XMLForXSLTransformation.....	960
	– XSLForXMLTransformation.....	961
	– XSLTransformationFileExtension.....	961
	– XSLTransformationFolder.....	961
3.2.26	SpyProjectItems.....	961
	– AddFile.....	961
	– AddFolder.....	962
	– AddURL.....	962
	– Count.....	962
	– Item.....	963
	– RemoveItem.....	963
3.2.27	TextImportExportSettings.....	963
	– CommentIncluded.....	963
	– DestinationFolder.....	964
	– EnclosingCharacter.....	964
	– Encoding.....	964
	– EncodingByteOrder.....	964
	– FieldDelimiter.....	964
	– FileExtension.....	965
	– HeaderRow.....	965
	– ImportFile.....	965
	– RemoveDelimiter.....	965
	– RemoveNewline.....	965
3.2.28	TextView.....	966
	– Events.....	966
	<i>OnBeforeShowSuggestions</i>	966
	<i>OnChar</i>	967
	– Application.....	967
	– GetRangeText.....	968
	– GoToLineChar.....	968

	– Length.....	968
	– LineCount.....	968
	– LineFromPosition.....	968
	– LineLength.....	969
	– MoveCaret.....	969
	– Parent.....	969
	– PositionFromLine.....	969
	– ReplaceText.....	969
	– SelectionEnd.....	970
	– SelectionStart.....	970
	– SelectText.....	970
	– SelText.....	970
	– Text.....	970
3.2.29	WSDLDocumentationDlg.....	971
	– UseFixedDesign.....	972
	– SPSFile.....	972
3.2.30	XBRLDocumentationDlg.....	972
	– SPSFile.....	973
	– UseFixedDesign.....	974
3.2.31	XMLData.....	974
	– AppendChild.....	975
	– CountChildren.....	976
	– CountChildrenKind.....	976
	– EraseAllChildren.....	976
	– EraseChild.....	977
	– EraseCurrentChild.....	977
	– GetChild.....	978
	– GetChildAttribute.....	978
	– GetChildElement.....	978
	– GetChildKind.....	978
	– GetCurrentChild.....	979
	– GetFirstChild.....	979
	– GetNamespacePrefixForURI.....	980
	– GetNextChild.....	980
	– GetTextValueXMLDecoded.....	981
	– HasChildren.....	981
	– HasChildrenKind.....	981
	– InsertChild.....	982
	– InsertChildAfter.....	982
	– InsertChildBefore.....	982
	– IsSameNode.....	983
	– Kind.....	983
	– MayHaveChildren.....	983
	– Name.....	984
	– Parent.....	984

	– SetValueXMLEncoded.....	984
	– TextValue.....	984
3.3	Schnittstellen (nicht mehr verwendet)	986
3.3.1	AuthenticEvent (obsolete).....	986
	– altKey (nicht mehr verwendet).....	987
	– altLeft (nicht mehr verwendet).....	988
	– button (nicht mehr verwendet).....	989
	– cancelButton (nicht mehr verwendet).....	990
	– clientX (nicht mehr verwendet).....	991
	– clientY (nicht mehr verwendet).....	992
	– ctrlKey (nicht mehr verwendet).....	993
	– ctrlLeft (nicht mehr verwendet).....	994
	– dataTransfer (nicht mehr verwendet).....	995
	– fromElement (nicht mehr verwendet).....	996
	– keyCode (nicht mehr verwendet).....	996
	– propertyName (nicht mehr verwendet).....	997
	– repeat (nicht mehr verwendet).....	997
	– returnValue (nicht mehr verwendet).....	997
	– shiftKey (nicht mehr verwendet).....	998
	– shiftLeft (nicht mehr verwendet).....	999
	– srcElement (nicht mehr verwendet).....	1000
	– type (nicht mehr verwendet).....	1001
3.3.2	AuthenticSelection (nicht mehr verwendet).....	1002
	– End (nicht mehr verwendet).....	1003
	– EndTextPosition (nicht mehr verwendet).....	1003
	– Start (nicht mehr verwendet).....	1004
	– StartTextPosition (nicht mehr verwendet).....	1004
3.3.3	OldAuthenticView (nicht mehr verwendet).....	1005
	– ApplyTextState (nicht mehr verwendet).....	1007
	– CurrentSelection (nicht mehr verwendet).....	1008
	– EditClear (nicht mehr verwendet).....	1008
	– EditCopy(nicht mehr verwendet).....	1009
	– EditCut (nicht mehr verwendet).....	1009
	– EditPaste (nicht mehr verwendet).....	1010
	– EditRedo (nicht mehr verwendet).....	1010
	– EditSelectAll (nicht mehr verwendet).....	1011
	– EditUndo (nicht mehr verwendet).....	1011
	– event (nicht mehr verwendet).....	1012
	– GetAllowedElements (nicht mehr verwendet).....	1012
	– GetNextVisible (nicht mehr verwendet).....	1015
	– GetPreviousVisible (nicht mehr verwendet).....	1016
	– IsEditClearEnabled (nicht mehr verwendet).....	1017
	– IsEditCopyEnabled (nicht mehr verwendet).....	1017
	– IsEditCutEnabled (nicht mehr verwendet).....	1018
	– IsEditPasteEnabled (nicht mehr verwendet).....	1018

– IsEditRedoEnabled (nicht mehr verwendet).....	1019
– IsEditUndoEnabled (nicht mehr verwendet).....	1019
– IsRowAppendEnabled (nicht mehr verwendet).....	1020
– IsRowDeleteEnabled (nicht mehr verwendet).....	1020
– IsRowDuplicateEnabled (nicht mehr verwendet).....	1021
– IsRowInsertEnabled (nicht mehr verwendet).....	1021
– IsRowMoveDownEnabled (nicht mehr verwendet).....	1023
– IsRowMoveUpEnabled (nicht mehr verwendet).....	1023
– IsTextStateApplied (nicht mehr verwendet).....	1024
– IsTextStateEnabled (nicht mehr verwendet).....	1024
– LoadXML (nicht mehr verwendet).....	1025
– MarkupView (nicht mehr verwendet).....	1025
– RowAppend (nicht mehr verwendet).....	1027
– RowDelete (nicht mehr verwendet).....	1027
– RowDuplicate (nicht mehr verwendet).....	1028
– RowInsert (nicht mehr verwendet).....	1028
– RowMoveDown (nicht mehr verwendet).....	1029
– RowMoveUp (nicht mehr verwendet).....	1029
– SaveXML (nicht mehr verwendet).....	1030
– SelectionMoveTabOrder (nicht mehr verwendet).....	1031
– SelectionSet (nicht mehr verwendet).....	1032
– XMLRoot (nicht mehr verwendet).....	1033
3.4 Enumerationen	1035
3.4.1 ENUMApplicationStatus.....	1035
3.4.2 SPYAttributeTypeDefinition.....	1035
3.4.3 SPYAuthenticActions.....	1035
3.4.4 SPYAuthenticDocumentPosition.....	1035
3.4.5 SPYAuthenticElementActions.....	1036
3.4.6 SPYAuthenticElementKind	1036
3.4.7 SPYAuthenticMarkupVisibility.....	1036
3.4.8 SPYAuthenticToolBarButtonState.....	1036
3.4.9 SPYDatabaseKind.....	1037
3.4.10 SPYDialogAction.....	1037
3.4.11 SPYDOMType.....	1037
3.4.12 SPYDTDSchemaFormat	1037
3.4.13 SPYEncodingByteOrder.....	1038
3.4.14 SPYExportNamespace.....	1038
3.4.15 SPYFindInFilesSearchLocation.....	1038
3.4.16 SPYFrequentElements.....	1038
3.4.17 SPYImageKind.....	1038
3.4.18 SPYImportColumnsType	1038
3.4.19 SPYKeyEvent.....	1039
3.4.20 SPYKeyStatus.....	1039
3.4.21 SPYLibType.....	1039
3.4.22 SPYLoading.....	1039

3.4.23	SPYMouseEvent	1039
3.4.24	SPYNumberDateTimeFormat	1040
3.4.25	SPYProgrammingLanguage	1040
3.4.26	SPYProjectItemTypes	1041
3.4.27	SPYProjectType	1041
3.4.28	SPYSampleXMLGenerationOptimization	1041
3.4.29	SPYSampleXMLGenerationSchemaOrDTDAssignment	1041
3.4.30	SPYSchemaDefKind	1042
3.4.31	SPYSchemaDocumentationFormat	1042
3.4.32	SPYSchemaExtensionType	1042
3.4.33	SPYSchemaFormat	1043
3.4.34	SPYTextDelimiters	1043
3.4.35	SPYTextEnclosing	1043
3.4.36	SPYTypeDetection	1043
3.4.37	SPYURLTypes	1043
3.4.38	SPYViewModes	1044
3.4.39	SPYVirtualKeyMask	1044
3.4.40	SPYXMLDataKind	1045
3.5	Applikations-API für Java	1046
3.5.1	Beispielquellcode	1047
3.5.2	SpyApplication	1049
3.5.3	SpyCodeGeneratorDlg	1050
3.5.4	SpyDatabaseConnection	1050
3.5.5	SpyDialogs	1051
3.5.6	SpyDoc	1052
3.5.7	SpyDocuments	1053
3.5.8	SpyDTDSchemaGeneratorDlg	1053
3.5.9	SpyElementList	1054
3.5.10	SpyElementListItem	1054
3.5.11	SpyExportSettings	1054
3.5.12	SpyFileSelectionDlg	1055
3.5.13	SpyFindInFilesDlg	1055
3.5.14	SpyFindInFilesMatch	1056
3.5.15	SpyFindInFilesResult	1056
3.5.16	SpyFindInFilesResults	1056
3.5.17	SpyGenerateSampleXMLDlg	1056
3.5.18	SpyGridView	1057
3.5.19	SpyProject	1057
3.5.20	SpyProjectItem	1058
3.5.21	SpyProjectItems	1058
3.5.22	SpySchemaDocumentationDlg	1058
3.5.23	SpyTextImportExportSettings	1060
3.5.24	SpyTextView	1060
3.5.25	SpyWSDL20DocumentationDlg	1061
3.5.26	SpyWSDLDocumentationDlg	1062

3.5.27	SpyXBRLDocumentationDlg.....	1064
3.5.28	SpyXMLData.....	1066
3.5.29	Authentic.....	1066
	– SpyAuthenticRange.....	1066
	– SpyAuthenticView.....	1068
	– SpyDocEditSelection.....	1068
	– SpyDocEditView.....	1068
3.5.30	Vordefinierte Konstanten.....	1069
	– SPYApplicationStatus.....	1069
	– SPYAttributeTypeDefinition.....	1069
	– SPYAuthenticActions.....	1070
	– SPYAuthenticDocumentPosition.....	1070
	– SPYAuthenticElementKind.....	1070
	– SPYAuthenticMarkupVisibility.....	1071
	– SPYDatabaseKind.....	1071
	– SPYDialogAction.....	1071
	– SPYDOMType.....	1071
	– SPYDTDSchemaFormat.....	1072
	– SPYEncodingByteOrder.....	1072
	– SPYExportNamespace.....	1072
	– SPYFindInFilesSearchLocation.....	1072
	– SPYFrequentElements.....	1072
	– SPYImageKind.....	1073
	– SPYLibType.....	1073
	– SPYLoading.....	1073
	– SPYNumberDateTimeFormat.....	1073
	– SPYProgrammingLanguage.....	1073
	– SPYProjectItemTypes.....	1074
	– SPYProjectType.....	1074
	– SPYSampleXMLGenerationOptimization.....	1074
	– SPYSampleXMLGenerationSchemaOrDTDAssignment.....	1074
	– SPYSchemaDefKind.....	1075
	– SPYSchemaDocumentationFormat.....	1075
	– SPYSchemaExtensionType.....	1075
	– SPYSchemaFormat.....	1076
	– SPYTextDelimiters.....	1076
	– SPYTextEnclosing.....	1076
	– SPYTypeDetection.....	1076
	– SPYURLTypes.....	1077
	– SpyViewModes.....	1077
	– SPYWhitespaceComparison.....	1077
	– SPYXMLDataKind.....	1077

4.1	Integration auf Applikationsebene	1080
4.1.1	Beispiel: HTML	1080
	– Instantiieren des Control	1080
	– Hinzufügen einer Schaltfläche zum Öffnen eines Standarddokuments	1081
	– Hinzufügen von Schaltflächen für die Codegenerierung	1081
	– Verbindung zu benutzerdefinierten Events herstellen	1082
4.2	Integration auf Dokumentebene	1083
4.2.1	Verwendung von XMLSpyControl	1083
4.2.2	Verwendung von XMLSpyControlDocument	1084
4.2.3	Verwendung von XMLSpyControlPlaceHolder	1084
4.2.4	Abfragen von XMLSpy Befehlen	1084
4.2.5	Beispiele	1084
	– C#	1085
	<i>Einführung</i>	1085
	<i>Platzieren des XMLSpyControl</i>	1085
	<i>Hinzufügen des Document Control</i>	1085
	<i>Hinzufügen des Placeholder Control</i>	1086
	<i>Abrufen von Befehlsinformationen</i>	1086
	<i>Behandlung von Events</i>	1087
	<i>Testen des Beispiels</i>	1088
	– HTML	1089
	<i>Instantiieren des XMLSpyControl</i>	1089
	<i>Erstellen eines Bearbeitungsfensters</i>	1089
	<i>Erstellen eines Projektfensters</i>	1090
	<i>Erstellen eines Placeholder für Eingabehilfenfenster</i>	1090
	<i>Erstellen einer benutzerdefinierten Symbolleiste</i>	1090
	<i>Erstellen weiterer Schaltflächen</i>	1091
	<i>Erstellen eines Event Handler zum Aktualisieren des Schaltflächenstatus</i>	1093
4.3	Befehlstabelle für XMLSpy	1094
4.3.1	Menü "Datei"	1094
4.3.2	Menü "Bearbeiten"	1095
4.3.3	Menü "Projekt"	1095
4.3.4	Menü "XML"	1096
4.3.5	Menü "DTD/Schema"	1098
4.3.6	Menü "Schema-Design"	1098
4.3.7	Menü "XSL/XQuery"	1099
4.3.8	Menü "Authentic"	1101
4.3.9	Menü "Konvertieren"	1101
4.3.10	Menü "Ansicht"	1102
4.3.11	Menü "Browser"	1102
4.3.12	Menü "WSDL"	1103
4.3.13	Menü "SOAP"	1104
4.3.14	Menü "Extras"	1105
4.3.15	Menü "Fenster"	1105

4.3.16	Menü "Hilfe".....	1105
4.4	Aufrufen von XMLSpyAPI	1107
4.5	Objektreferenz	1108
4.5.1	XMLSpyCommand.....	1108
	– Accelerator.....	1108
	– ID	1108
	– IsSeparator.....	1109
	– Label.....	1109
	– StatusText.....	1109
	– SubCommands	1109
	– ToolTip.....	1109
4.5.2	XMLSpyCommands.....	1109
	– Count.....	1110
	– Item.....	1110
4.5.3	XMLSpyControl.....	1110
	– Eigenschaften.....	1110
	<i>Appearance</i>	1111
	<i>Application</i>	1111
	<i>BorderStyle</i>	1111
	<i>CommandsList</i>	1111
	<i>EnableUserPrompts</i>	1112
	<i>IntegrationLevel</i>	1112
	<i>MainMenu</i>	1112
	<i>Toolbars</i>	1112
	– Methoden	1112
	<i>Exec</i>	1113
	<i>Open</i>	1113
	<i>QueryStatus</i>	1113
	– Events.....	1113
	<i>OnCloseEditingWindow</i>	1114
	<i>OnContextChanged</i>	1114
	<i>OnDocumentOpened</i>	1114
	<i>OnFileChangedAlert</i>	1114
	<i>OnLicenseProblem</i>	1115
	<i>OnOpenedOrFocused</i>	1115
	<i>OnToolWindowUpdated</i>	1115
	<i>OnUpdateCmdUI</i>	1115
	<i>OnValidationWindowUpdated</i>	1116
4.5.4	XMLSpyControlDocument.....	1116
	– Eigenschaften.....	1117
	<i>Appearance</i>	1117
	<i>BorderStyle</i>	1117
	<i>Document</i>	1117
	<i>IsModified</i>	1117
	<i>Path</i>	1118

	<i>ReadOnly</i>	1118
–	Methoden	1118
	<i>Exec</i>	1118
	<i>New</i>	1119
	<i>Open</i>	1119
	<i>QueryStatus</i>	1119
	<i>Reload</i>	1119
	<i>Save</i>	1119
	<i>SaveAs</i>	1120
–	Events	1120
	<i>OnActivate</i>	1120
	<i>OnContextChanged</i>	1120
	<i>OnDocumentClosed</i>	1121
	<i>OnDocumentOpened</i>	1121
	<i>OnDocumentSaveAs</i>	1121
	<i>OnFileChangedAlert</i>	1121
	<i>OnModifiedFlagChanged</i>	1121
	<i>OnSetEditorTitle</i>	1122
4.5.5	XMLSpyControlPlaceholder	1122
–	Eigenschaften	1122
	<i>Label</i>	1122
	<i>PlaceholderWindowID</i>	1123
	<i>Project</i>	1123
–	Methoden	1123
	<i>OpenProject</i>	1123
	<i>CloseProject</i>	1123
–	Events	1124
	<i>OnModifiedFlagChanged</i>	1124
	<i>OnSetLabel</i>	1124
4.5.6	Enumerationen	1124
–	ICActiveXIntegrationLevel	1124
–	XMLSpyControlPlaceholderWindow	1124

Anhänge

1128

1 Prozessor-Informationen

1129

1.1	XSLT 1.0-Prozessor: Implementierungsinformationen	1130
1.2	XSLT 2.0-Prozessor: Implementierungsinformationen	1132
1.2.1	Allgemeine Informationen	1132
1.2.2	XSLT 2.0-Elemente und -Funktionen	1134

1.3	XQuery 1.0-Prozessor: Implementierungsinformationen	1135
1.4	XPath 2.0- und XQuery 1.0-Funktionen	1138
1.4.1	Allgemeine Informationen.....	1138
1.4.2	Unterstützung von Funktionen.....	1139
1.5	Erweiterungen	1143
1.5.1	Java-Erweiterungsfunktionen.....	1143
	– Benutzerdefinierte Klassendateien.....	1145
	– Benutzerdefinierte Jar-Dateien.....	1147
	– Java: Konstruktoren	1148
	– Java: Statische Methoden und statische Felder	1149
	– Java: Instanzmethoden und Instanzfelder.....	1149
	– Datentypen: XPath/XQuery in Java	1150
	– Datentypen: Java in XPath/XQuery	1151
1.5.2	.NET-Erweiterungsfunktionen.....	1151
	– .NET: Konstruktoren.....	1154
	– .NET: Statische Methoden und statische Felder.....	1154
	– .NET: Instanzmethoden und Instanzfelder.....	1155
	– Datentypen: XPath/XQuery in .NET.....	1156
	– Datentypen: .NET in XPath/XQuery.....	1157
1.5.3	MSXSL Scripts für XSLT.....	1157
1.5.4	Altova Erweiterungsfunktionen.....	1159
	– Allgemeine Funktionen.....	1160
2	Datentypen in XML-Schemas, die von DB generiert wurden	1164
2.1	MS Access	1165
2.2	MS SQL Server	1166
2.3	MySQL	1167
2.4	Oracle	1168
2.5	ODBC	1169
2.6	ADO	1170
2.7	Sybase	1171
3	Datentypen in DB, die von XML-Schema generiert wurden	1172
3.1	MS Access	1173
3.2	MS SQL Server	1175
3.3	MySQL	1177
3.4	Oracle	1179
4	Technische Daten	1181
4.1	OS- und Arbeitsspeicheranforderungen	1182
4.2	Altova XML Parser	1183

4.3	Altova XSLT- und XQuery-Prozessor	1184
4.4	Unicode-Unterstützung	1185
4.4.1	Windows XP	1185
4.4.2	Rechts nach Links-Schreibrichtung	1186
4.5	Internet-Verwendung	1187

5	Lizenzinformationen	1188

5.1	Electronic Software Distribution	1189
5.2	Software-Aktivierung und Lizenzüberwachung	1190
5.3	Rechte am geistigen Eigentum	1191
5.4	Altova Endbenutzer-Lizenzvereinbarung	1192

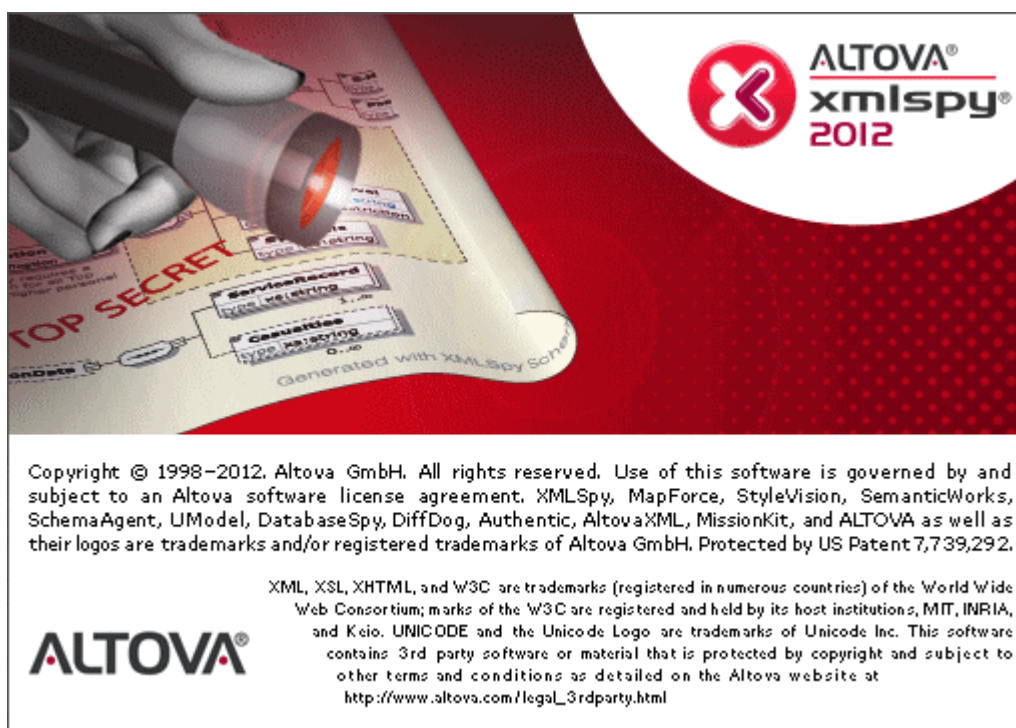
Index	1207

Altova XMLSpy 2012

Willkommen bei XMLSpy

Willkommen bei XMLSpy

Die [Altova XMLSpy® 2012 Professional Edition](#) ist die Industriestandard XML-Entwicklungsumgebung zum Erstellen, Editieren und Debuggen von Unternehmensapplikationen, die auf XML, XML-Schema, XSLT, XQuery, SOAP, WSDL und Webservice-Technologien basieren. Mit Hilfe dieser Anwendung können J2EE-, .NET- und Datenbank-Entwickler nun schneller und effizienter arbeiten. XMLSpy steht in einer 64-Bit und einer 32-Bit-Version zur Verfügung.



Diese Dokumentation ist in die folgenden Abschnitte gegliedert:

- [Benutzerhandbuch](#)
- [Referenz für Programmierer](#)
- [Anhänge](#)

Sie steht in folgenden Formaten zur Verfügung:

- in Form der integrierten Hilfe von XMLSpy ([Menü "Hilfe"](#) oder Taste **F1**)
- Auf der [Altova Website \(www.altova.com\)](#) steht Sie in den Formaten HTML und PDF zur Verfügung und kann in Form eines gedruckten Handbuchs bestellt werden.

Letzte Aktualisierung: 02/17/2012

Altova XMLSpy 2012

Benutzerhandbuch

Benutzerhandbuch

Dieser Teil der Dokumentation, das Benutzerhandbuch, enthält alle Informationen, die Sie benötigen, um sich mit [XMLSpy](#) und seinen verschiedenen Funktionalitäten vertraut zu machen. Das Benutzerhandbuch ist in vier allgemeine Teile gegliedert: (i) eine [Einführung](#); (ii) ein [Tutorial](#); (iii) eine Anleitung für die Arbeit mit [XMLSpy](#) und (iii) eine [Benutzerreferenz](#).

Am besten, Sie beginnen mit der Einführung, um einen Überblick über die Benutzeroberfläche und die wichtigsten Applikationseinstellungen zu erhalten. Wenn Sie noch keine XML-Kenntnisse haben, arbeiten Sie das [XMLSpy Tutorial](#) durch, um sich mit XMLSpy vertraut zu machen und zu lernen, wie man einfache XML-Dokumente erstellt und bearbeitet. Lesen Sie anschließend die Abschnitte [Bearbeitungsansichten](#) und die [Anleitung für die Arbeit mit XMLSpy](#). Diese enthalten die wichtigsten Informationen für Sie. Der Abschnitt [Benutzerreferenz](#) kann anschließend zu Referenzzwecken herangezogen werden.

Einführung

In der [Einführung](#) werden die Benutzeroberfläche und wichtige Einstellungen im Dialogfeld "Optionen" beschrieben.

Tutorials

Das [XMLSpy Tutorial](#) hilft Ihnen, sich mit XMLSpy und den wichtigsten Funktionen von XMLSpy vertraut zu machen.

Arbeiten mit

Im Teil "Arbeiten mit" werden die Funktionalitäten beschrieben, die in XMLSpy zur Entwicklung der verschiedensten XML-Ressourcen zur Verfügung stehen. Im ersten Abschnitt dieses Teils werden die [verschiedenen Bearbeitungsansichten](#) von XMLSpy beschrieben. Im Anschluss daran werden die Funktionalitäten erläutert, die zur Entwicklung spezifischer XML-Ressourcen zur Verfügung stehen. Abschließend enthält der Teil "Arbeiten mit" eine Reihe von Abschnitten über Funktionen, die in der gesamten Applikation zur Verfügung stehen. Im Folgenden sind die einzelnen Abschnitte dieses Teils aufgelistet:

- [Bearbeitungsansichten](#): eine Beschreibung der verschiedenen Bearbeitungsansichten in XMLSpy
- [XML](#): eine Beschreibung der verschiedenen Funktionen für das Arbeiten mit XML-Dokumenten in XMLSpy
- [DTDs und XML-Schemas](#): eine Anleitung, wie Schemas (DTDs und XML-Schemas) in XMLSpy bearbeitet und verwendet werden können.
- [XSLT und XQuery](#): Beschreibung der Funktionspalette für die XSLT- und XQuery-Erstellung
- [Authentic](#): Anleitung zur Verwendung des grafischen XML-Editors von Altova
- [HTML, CSS und JSON](#): eine Beschreibung der HTML- und CSS-Funktionen von XMLSpy
- SOAP und WSDL: Beschreibung der Mechanismen zum Arbeiten mit diesen Protokollen in XMLSpy
- [MS Office Open XML und ZIP-Dateien](#): Eine Beschreibung der Archiv-Ansicht, in der Dokumente, für die diese Standards verwendet werden, bearbeitet und verwaltet werden können
- [Datenbanken](#): eine Beschreibung der breiten Palette an XMLSpy-Funktionen für die Anbindung von XML an Datenbanken
- [Globale Ressourcen](#): eine Beschreibung dieser einzigartigen Altova-Funktionalitäten,

die bei der Arbeit mit dem Altova-Produktpaket zur effizienteren Entwicklung eingesetzt werden können

- [Projekte](#): Beschreibung der XMLSpy-Projektfunktionen zur Gliederung Ihrer Arbeit in Projekte zur besseren Effizienz und um zusätzliche Entwicklungsoptionen nutzen zu können
- [Datei-/Verzeichnisvergleiche](#): Beschreibung des integrierten Vergleichsprozessors von XMLSpy
- XMLSpy in Visual Studio: eine Beschreibung der Integration in diese Entwicklungsplattform
- XMLSpy in Eclipse: eine Beschreibung der Integration in diese Entwicklungsplattform
- Code Generator: eine Beschreibung der Funktionsweise des integrierten Code Generators, der Programmcode für Schemadefinitionen erzeugt

Benutzerreferenz

Der Teil [Benutzerreferenz](#) ist nach den Menüs in XMLSpy gegliedert und enthält eine Beschreibung der einzelnen Menübefehle in allen Einzelheiten.

Dateipfade in Windows XP, Windows Vista und Windows 7

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 in Windows XP entspricht in Windows Vista und Windows 7 dem Verzeichnis Dateien. Es befindet sich standardmäßig an den nachstehend angeführten Orten. Beispieldateien befinden sich im Normalfall in einem Unterverzeichnis des Verzeichnisses (Eigene) Dateien.

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien
Windows Vista, Windows 7	C:/Benutzer/<Benutzername>/Dateien

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

Windows XP	C:/Programme/Altova
Windows Vista, Windows 7	C:/Programme/Altova
32-Bit Paket auf 64-Bit Windows OS (XP, Vista, 7)	C:/Programme (x86)/Altova

1 Neue Funktionen

XMLSpy **Version 2012** enthält die folgenden neuen Funktionen. Weiter unten sind die neuen Funktionen vorhergehender Versionen aufgelistet.

Version 2012 Release 2

- 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 Archiv-Ansicht öffnen, die Dateien 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](#).
- Mit Hilfe eines Kontextmenübefehls können XML-Sonderzeichen mit [Escape versehen werden oder das Escape-Zeichen kann wieder entfernt werden](#).
- Fehlerbehebungen.

Version 2012

- [In Projekten suchen](#): Mit dieser Funktion lassen sich Projektdateien und -ordner im Projektfenster schnell finden.
- Eingabehilfen und intelligente Bearbeitungshilfen für [CSS 3.0](#). CSS 3.0 wird zusätzlich zu CSS 2.1 unterstützt.
- Unterstützung für [HTML 5.0](#)
- Unterstützung für [JDBC-Verbindungen in Datenbanken](#).
- Neue [Applikations-API-Schnittstelle für Java](#).

Version 2011 Release 3

- Vereinfachte [Rechtschreibprüfung](#)
- [Die Skripterstellung und der Skript-Editor](#) in XMLSpy unterstützen (zusätzlich zu den Versionen 2.0, 3.0 und 3.5) .NET Version 4.0.

Version 2011 Release 2

- In der Schema-Ansicht kann eine [Dokumentation des XML Schema-Dokuments mit Hilfe eines SPS generiert](#) werden, wobei das SPS im StyleVision Produkt von Altova angepasst werden kann. In früheren Versionen von XMLSpy stand für das Design der generierten Dokumentation ein Standard-Design zur Verfügung. Diese neue Funktion erlaubt den Benutzern die Struktur und die Formatierung des Designs selbst zu gestalten.
- Der Inhalt einer externen Datei, wie zum Beispiel einer Bilddatei, kann als Base-16 oder Base-64 Text kodiert und in Text-Ansicht und Grid-Ansicht integriert werden. Dies geschieht mit Hilfe des Befehls [Einfügen | Externe kodierte Datei](#) im Menü "Bearbeiten".
- Die Schema-Ansicht wurde mit folgenden Funktionen erweitert: (i) [Sortieren von globalen Komponenten, Attributen und Identity Constraints](#); Sortierung wurde auch in der [Content Model-Ansicht](#) eingeführt; (ii) [Suchen und Umbenennen von globalen Komponenten](#) in den Schema-Dateien und in verwandten Schema-Dateien; (iii) Wenn der [Base Type eines abgeleiteten Types modifiziert](#) wird, können Inhalt, Attribute, Facets und Beispielwerte, die innerhalb des abgeleiteten Types definiert sind, erhalten werden, wenn sie auf den neuen Base Type noch anwendbar sind.

Version 2011

- Erstellen von [Schemamodulen](#) aus den Komponenten einer einzigen Schemadatei in der Schema-Ansicht. Eine Schema-Datei kann daher jederzeit in einzelne Module aufgeteilt werden. Auch der umgekehrte Vorgang, nämlich die Integration von Komponenten aus inkludierten Schemamodulen in die aktive Datei wird unterstützt.

Version 2010 Release 3

- [Generieren von Beispielwerten in Beispieldateien](#), die anhand eines XML-Schemas generiert werden. (Nur Enterprise und Professional Edition.)
- [Integration in Microsoft Visual Studio 2010](#). Damit wird nun neben Version 2005 und 2008 auch die neueste Version von Visual Studio unterstützt. Visual Studio 2003 wird nicht mehr unterstützt. (Nur Enterprise und Professional Edition.)

Version 2010 Release 2

- Optimierungen beim Arbeiten mit großen Dateien in der Text- und der Grid-Ansicht
- Die Validierung anhand von XML-Schemas erfolgt in der Textansicht nun bis zu drei Mal so schnell
- Dank der [Unterstützung für SharePoint® Server](#) können Dateien nun direkt von der XMLSpy Benutzeroberfläche aus aus- und eingecheckt werden.
- Schnelle Verknüpfung von CSS- und HTML-Dateien über die Benutzeroberfläche und einfaches Öffnen in verschiedenen Browser-Applikationen und Editoren dank dem [CSS- und dem HTML.-Infofenster](#).
- 64-Bit-Versionen von XMLSpy verfügbar für die Enterprise und die Professional Edition.

2 Einführung

Diese Einführung enthält eine Beschreibung :

- [der Benutzeroberfläche der Applikation](#) und
- [der Applikationsumgebung](#).

Der [Abschnitt "Benutzeroberfläche"](#) 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.

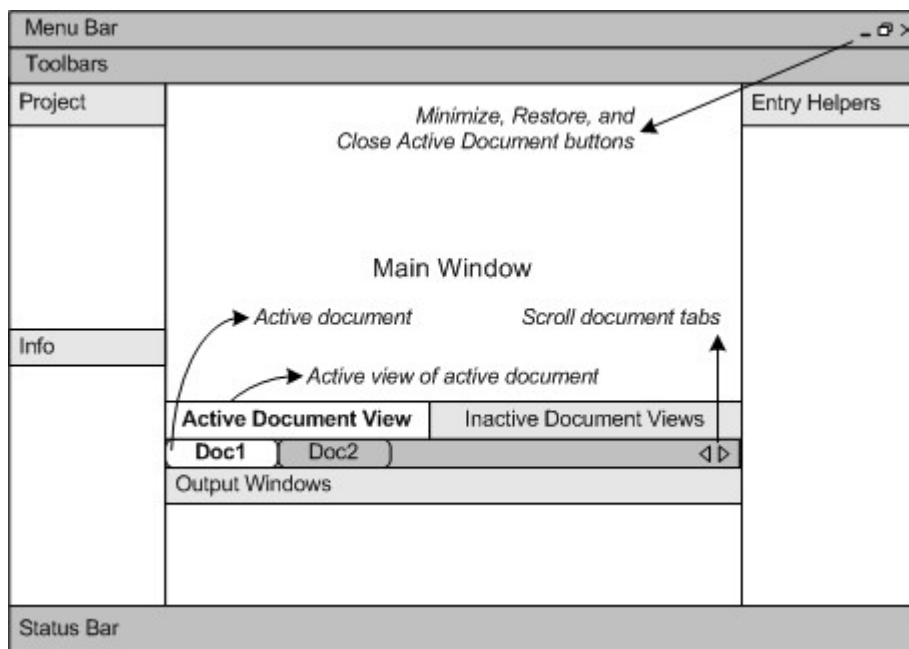
Im [Abschnitt "Applikationsumgebung"](#) werden die verschiedenen Einstellungen erläutert, mit denen festgelegt wird, wie die Dateien angezeigt und bearbeitet werden können. Außerdem wird darin erklärt, wie und wo Sie Ihre Applikation anpassen können. In diesem Abschnitt erfahren Sie, wo wichtige Beispiel- 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.

2.1 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:

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

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



Außerdem enthält die Benutzeroberfläche eine Menüleiste, eine Statusleiste sowie Symbolleisten, die alle in einem Unterabschnitt dieses Abschnitts 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, kann auch durch Rechtsklick auf die Titelleiste der Seitenleiste (oder die Titelleiste der mit Registern versehenen Seitenleistengruppe) und Auswahl des Befehls **Ausblenden** ausgeblendet werden.

Seitenleisten freischwebend anzeigen und andocken

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

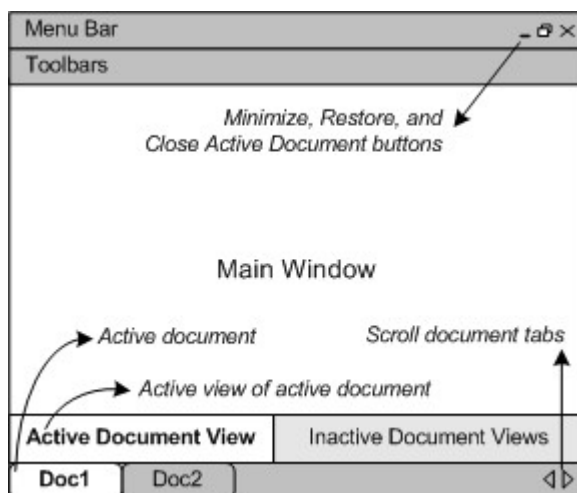
Automatisches Ausblenden der Seitenleisten

Mit der Funktion zum automatischen Ausblenden können Sie angedockte Seitenleisten als Schaltflächen am Rand des Applikationsfensters minimieren, damit Sie mehr Platz für das Hauptfenster und andere Seitenleisten haben. Wenn Sie über eine minimierten Seitenleiste scrollen, wird diese ausgerollt.

Um Seitenleisten automatisch auszublenden und wiederherzustellen, klicken Sie auf das Pin-Symbol in der Titelleiste des Seitenleistenfensters (oder rechtsklicken Sie auf die Titelleiste und wählen Sie den Befehl **Automatisch ausblenden**).

2.1.1 Hauptfenster

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



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](#) 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](#).
- 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](#) den entsprechenden Befehl auf.

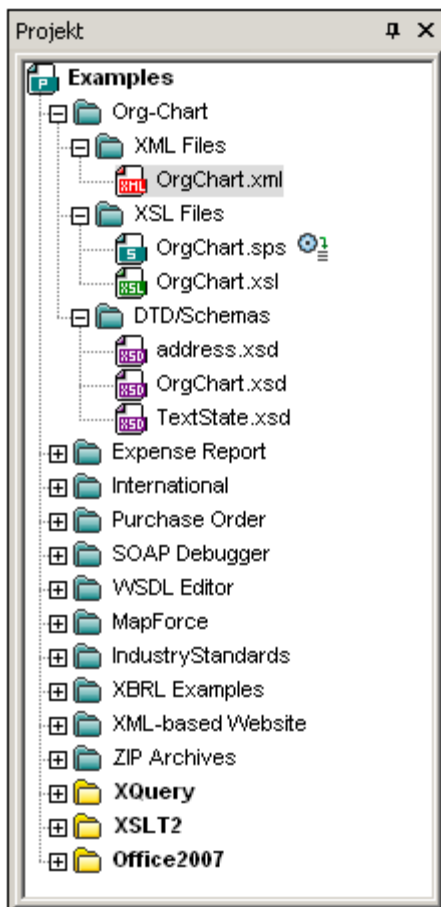
Verfügbar sind entweder Editier- oder Browser-Ansichten:

- [Textansicht](#): Eine Editieransicht mit Syntaxfärbung zur Bearbeitung einer Datei auf Source-Ebene
- [Grid-Ansicht](#): Zur strukturierten Bearbeitung. Das Dokument wird in Tabellenform dargestellt und kann grafisch bearbeitet werden. Diese Ansicht enthält auch eine eingebettete [Tabellen-Ansicht](#), in der sich wiederholende Elemente in Tabellenform dargestellt werden.
- [Schema-Ansicht](#): Zum Anzeigen und Bearbeiten von XML-Schemas.
- [Authentic-Ansicht](#): Zum Bearbeiten von XML-Dokumenten auf Basis von StyleVision Power Stylesheets
- [Browser-Ansicht](#): Eine integrierte Browser-Ansicht, die sowohl CSS- als auch XSL-Stylesheets unterstützt.

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

2.1.2 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` ist der Beispielordner `OrgChart` in Unterordner für XML-, XSL- und Schema-Dateien gegliedert.



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.

- 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.
- 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 Dateiartern enthalten. Alternativ dazu können Sie (im Dialogfeld "Eigenschaften" für diesen Ordner) für jeden Ordner Dateierweiterungen definieren, damit bestimmte Dateiartern 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.
- 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. Außerdem können Dateien und Ordner aus dem Windows Datei-Explorer in das Projektfenster 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.

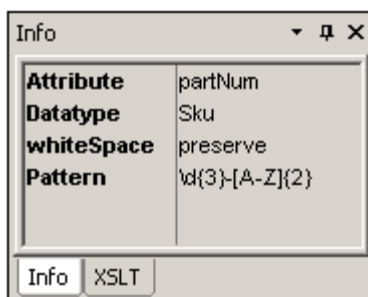
Eine nähere Beschreibung zu Projekten finden Sie im Abschnitt [Projekte](#).

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

2.1.3 Info-Fenster

Das Fenster "Info" (*Abbildung unten*) enthält detaillierte Informationen über das Element oder Attribut, in dem sich der Cursor gerade befindet.

- Wenn ein XSLT-Dokument aktiv ist, stehen auf den Register "XSLT" des Info-Fensters zusätzliche XSLT-spezifische Informationen und Befehle zur Verfügung (*siehe Register "XSLT" in der Abbildung unten*). Eine Erklärung zu den Informationen und Befehlen auf dem Register "XSLT" finden Sie im Abschnitt [XSLT und XQ | XSLT | XSL-Übersicht](#).



Das Info-Fenster enthält in der Textansicht, der Grid-Ansicht und der Authentic-Ansicht Informationen.

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

2.1.4 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. 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 Dokumente verfügbaren Eingabehilfen werden in 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 entsprechenden 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.

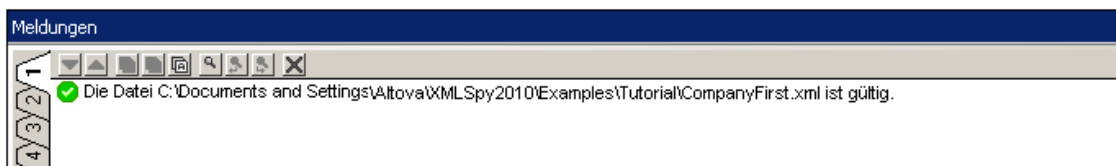
Eine Beschreibung der Eingabehilfen der einzelnen Ansichten finden Sie unter [Bearbeitungsansichten](#). Weitere Unterschiede im Zusammenhang mit Dokumentarten in einer Ansicht finden Sie in den Abschnitten über die einzelnen Dokumentarten, z.B. [XML-Eingabehilfen](#) und [XQuery-Eingabehilfen](#).

Hinweis:

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

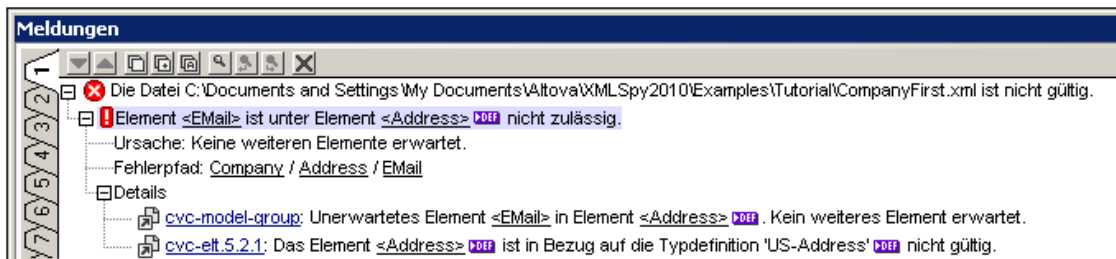
2.1.5 Ausgabefenster "Meldungen"

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 gültig ist, wird im Validierungsfenster eine entsprechende Meldung (*Abbildung unten*) angezeigt.



Andernfalls wird eine Meldung, die den Fehler beschreibt, angezeigt (*Abbildung unten*). Beachten Sie, 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.

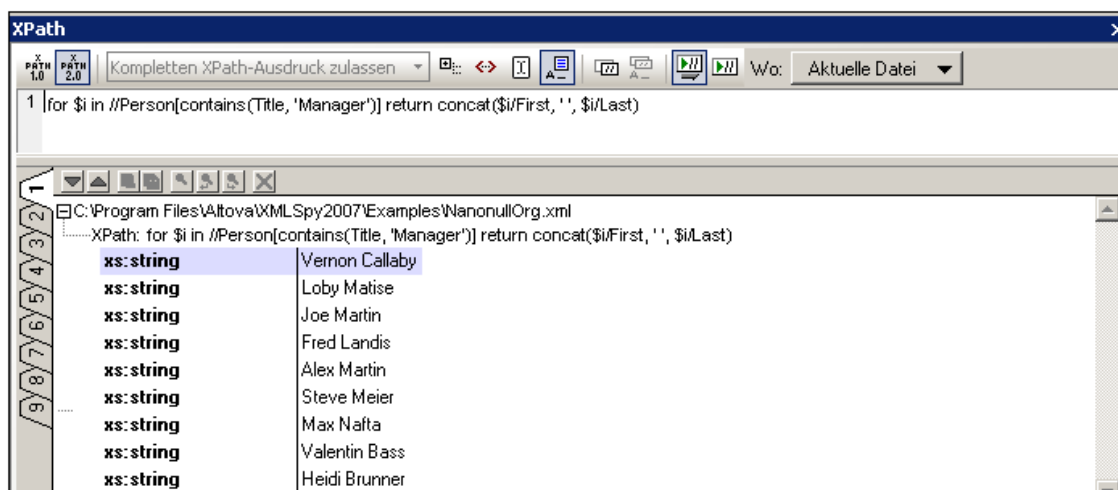


Das Validierungs-Ausgabefenster ist in allen Ansichten aktiv. Wenn Sie jedoch auf einen Link zum XML-Dokumentinhalt klicken, wird der entsprechende Node in der Text-Ansicht markiert. Beachten Sie jedoch: Wenn ein XML-Schema in der Schema-Ansicht validiert wird, ändert sich die Ansicht bei Klicken auf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

Anmerkung: 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** oder **F8**. Ungültige Dateien in einem Projekt werden geöffnet und im Hauptfenster aktiviert. Falls die Datei ungültig ist, wird eine entsprechende Fehlermeldung angezeigt.

2.1.6 Ausgabefenster "XPath"

Im Fenster "XPath" können Sie bis zu neun XPath-Ausdrücke auswerten, jeden davon im Zusammenhang mit dem gerade aktiven XML-Dokument. Außerdem können XPath-Ausdrücke in folgenden Dokumenten ausgewertet werden: (i) in allen gerade offenen XML-Dokumenten; (ii) in XML-Dateien des gerade aktiven XMLSpy Projekts; und (iii) in XML-Dateien eines ausgewählten Ordners. D.h. Sie können auf jedem der neun Ausgaberegister des XPath-Auswertungsfensters einen anderen XPath-Ausdruck eingeben und die einzelnen Ausdrücke anschließend an mehreren XML-Dateien auf einmal auswerten.



Eine großer Vorteil des Fensters "XPath" ist, dass Sie (i) den XPath-Ausdruck, (ii) das Ergebnis der Auswertung sowie (iii) das Ergebnis der Auswertung des XPath-Ausdrucks am aktiven

XML-Dokument in einer einzigen Ansicht sehen. Dadurch können Sie (i) im XML-Dokument navigieren, während Sie den XPath-Ausdruck und die Ergebnisse sehen und (ii) im XML-Dokument durch Klicken auf Einträge im Ergebnis navigieren, (iii) den XPath-Ausdruck bei eingeblendetem XML-Dokument ändern.

Schritte bei der Auswertung eines XPath-Ausdrucks

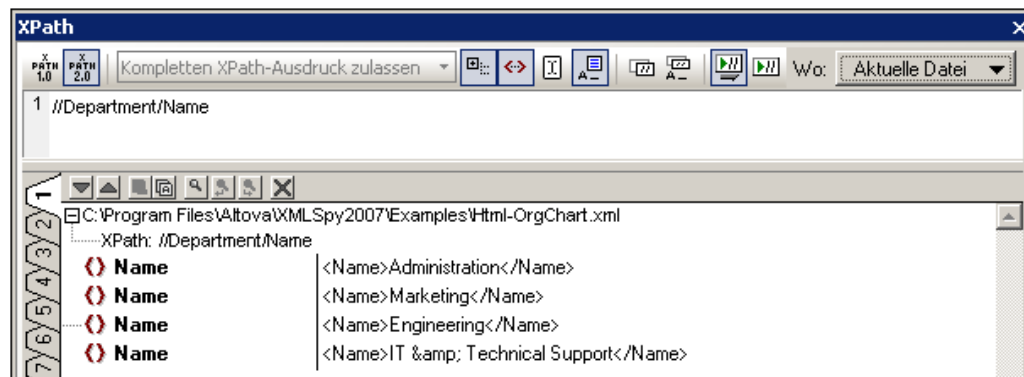
Im Folgenden sehen Sie die wichtigsten Schritte bei der Verwendung des Fensters "XPath".

1. Wählen Sie, je nachdem, wo der XPath-Ausdruck ausgewertet werden soll, einer der Optionen in der Liste der "Where"-Optionen (*Abbildung unten*): *Aktuelle Datei*; *Offene Dateien*; *(XMLSpy) Projekt* oder *Ordner*.



Bei Auswahl der Option *Aktuelle Datei* wird die gerade aktive Datei verwendet. Bei Auswahl von *Offene Dateien* wird der XPath-Ausdruck an allen derzeit in XMLSpy geöffneten Dateien ausgewertet. Projekt bezieht sich auf das derzeit aktive XMLSpy Projekt. Die externen Ordner in einem XMLSpy Projekt können durch Aktivierung des Kontrollkästchens *Externe Verzeichnisse überspringen* von der Auswertung ausgenommen werden. Mit Hilfe der Option *Ordner* können Sie nach dem gewünschten Ordner suchen. Der XPath-Ausdruck wird an den XML-Dateien in diesem Ordner ausgewertet.

2. Wählen Sie die gewünschte XPath-Version (1.0 oder 2.0), indem Sie auf das entsprechende Symbol in der Symbolleiste des Ausgabefensters klicken. (*siehe Abbildung unten*)
3. Wählen Sie die Art des XPath-Ausdrucks aus der Dropdown-Auswahlliste aus. Normalerweise benötigen Sie die Option *Vollständigen XPath* zulassen. Die Optionen *XPath-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](#) des W3C.
4. Aktivieren Sie das Symbol "XPath-Ausdruck bei der Eingabe auswerten" , wenn der XPath-Ausdruck schon während der Eingabe ausgewertet werden soll. Wenn dieses Symbol deaktiviert ist, wird der Ausdruck erst dann ausgewertet, wenn Sie auf das Symbol "XPath-Ausdruck auswerten"  klicken.
5. Aktivieren Sie das Symbol "Kopfzeile in der Ausgabe anzeigen" , um den Pfad zur XML-Datei und dem XPath-Ausdruck in der Ausgabe anzuzeigen (*siehe Abbildung unten*).
6. Wenn der XPath-Ausdruck Nodes zurückgibt - wie z.B. Elemente oder Attribute - können Sie auswählen, ob die einzelnen Nodes nach Node-Namen aufgelistet werden sollen oder ob auch der Inhalt des Node angezeigt werden soll. Klicken Sie dazu auf die Schaltfläche *Gesamtes Ergebnis anzeigen* . In der unten gezeigten Abbildung wird das Element und der Inhalt der Nodes angezeigt. (in der Spalte rechts).



7. Um einen XPath-Ausdruck relativ zu einer Auswahl im XML-Dokument zu definieren, aktivieren Sie das Symbol "Aktuelle Auswahl in Dokument als Ursprung für XPath-Ausdruck festlegen" .
8. Geben Sie den XPath-Ausdruck ein. Die Funktion zur intelligenten XPath-Bearbeitung zeigt daraufhin eine Liste von XPath-Funktionen, XPath-Achsen und Dokumentelemente und Attribute an, die zur Auswahl bereit stehen. Wenn der Ausdruck mehrere Zeilen lang sein ist, drücken Sie die **Eingabetaste**.
9. Um den Ausdruck auszuwerten, klicken Sie auf das Symbol "XPath-Ausdruck auswerten" (wenn die Option "XPath-Ausdruck bei der Eingabe auswerten" deaktiviert ist oder wenn ein neues XML-Dokument zum aktiven gemacht wird).

Funktionen des Ergebnisfensters

Das Ergebnisfenster hat einige sehr nützliche Features:

- 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 ein Node im Ergebnisfenster ausgewählt wird, wird bei Auswahl des Symbols "Kopiert den XPath-Ausdruck von der aktuellen Auswahl in das Bearbeiten-Feld"  (i) ein XPath-Ausdruck generiert, um den ausgewählten Node zu finden; (ii) der Ausdruck, der gerade im Feld "Bearbeiten" steht, durch den neu generierten XPath-Ausdruck überschrieben.
- Mit dem Schaltflächenbefehl "Kopiert den XPath-Ausdruck von der aktuellen Auswahl in das Bearbeiten-Feld"  wird der XPath-Ausdruck vom gerade offenen Ergebnisregister in das Bearbeitungsfeld für den XPath-Ausdruck kopiert. Dies ist nützlich, wenn (i) der Ausdruck im Bearbeiten-Feld ein anderer ist als der, der zum Generieren des Ergebnisses auf einem der Ergebnisregister verwendet wurde und (ii) Sie den auf diesem Ergebnisregister verwendeten Ausdruck nochmals auswerten möchten. In diesem Fall machen Sie das Ergebnisregister zum aktiven und kopieren den XPath-Ausdruck dieses Register in das Bearbeiten-Feld. Der XPath-Ausdruck kann dann erneut ausgewertet werden.

XPath-Ausdrücke im Bearbeiten-Feld

Beachten Sie bitte die folgenden allgemeinen Punkte zu XPath-Ausdrücken. XPath- 1.0 und XPath 2.0-spezifische Punkte werden weiter unten gesondert abgehandelt.

- Wenn Sie im Bearbeiten-Feld einen Ausdruck eingeben, wird er schwarz angezeigt, wenn die Syntax korrekt ist und rot, wenn die Syntax nicht korrekt ist. Beachten Sie, dass eine korrekte Syntax nicht bedeutet, dass der Ausdruck keinen Fehler enthält. Die Fehlermeldungen werden im Ergebnisfenster angezeigt.
- Ein Fehler kann dadurch entstehen, dass XPath-Evaluator Optionen falsch eingestellt

wurden. Stellen Sie daher die richtigen Optionen ein, v.a. die XPath-Version und den XPath-Ursprung. (absolut oder relativ zur aktuellen Auswahl im derzeit aktiven XML-Dokument)

- Nur Namespaces, die sich am Element, an dem der XPath seinen Ursprung hat (d.h. am Kontext-Node) im Geltungsbereich befinden, werden ausgewertet.

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-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 verwendet, 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-Namespaces 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-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 2.0 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.
- Bei Verwendung der beiden Duration-Datentypen (`yearMonthDuration` und `dayTimeDuration`) muss der XML-Schema-Namespaces (`http://www.w3.org/2001/XMLSchema`) in der XML-Datei deklariert werden und das Namespace-Präfix muss bei diesen Typen im XPath-Ausdruck verwendet werden. Beispiel:
`years-from-yearMonthDuration(xs:yearMonthDuration('P22Y18M'))`.

Bitte beachten Sie: Zusammenfassend ist zu Namespaces zu sagen: Bei Verwendung von Konstruktoren oder Typen aus dem XML-Schema-Namespaces müssen Sie den XML-Schema-Namespaces 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

Wenn Sie einen XPath 2.0-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 casten. In dem vom integrierten XPath-Prozessor verwendeten XPath 2.0-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.

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 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.

Bitte beachten Sie: Die XPath-Prozessoren, die vom XPath Evaluator verwendet werden, werden auch vom Altova XSLT-Prozessor verwendet. XPath 2.0 Ausdrücke in XSLT Stylesheets, die nicht implizit in den erforderlichen Datentyp konvertiert werden, müssen explizit als der benötigte Datentyp konstruiert oder gecastet 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.

Fehlerbehebung

Wenn Ihr XPath-Ausdruck einen Fehler zurückgibt, gehen Sie folgendermaßen vor:

1. Überprüfen Sie, ob die Optionen richtig eingestellt wurden (XPath-Version, Ursprung, usw.).
2. Überprüfen Sie, ob die Funktions-, Konstruktor-, Node-Namen usw. korrekt sind.
3. Überprüfen Sie, ob Präfixe erforderlich sind und ob diese Präfixe bei Funktionen, Konstruktoren und Argumenten im XPath-Ausdruck korrekt gesetzt sind.
4. Falls im XML-Dokument Namespaces deklariert sind, überprüfen Sie, ob diese korrekt sind. Die korrekten Namespaces sind:

XML-Schema: `http://www.w3.org/2001/XMLSchema`

XPath 2.0-Funktionen: `http://www.w3.org/2005/xpath-functions`

Unterstützung von XPath 2.0-Funktionen

Siehe [Anhänge](#).

2.1.7 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.

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 Wort 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 Expressions aufzurufen, klicken Sie auf die Schaltfläche . Die Verwendung von Regular Expressions zur Suche wird unter [Suchen](#) erklärt. Über die Schaltfläche **Mehr** wird das Dialogfeld [In Dateien suchen](#) geöffnet. Hier können Sie weitere Suchbedingungen und Aktionen definieren. Nähere Informationen finden Sie unter [Bearbeiten | In Dateien suchen](#).

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"](#) (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:

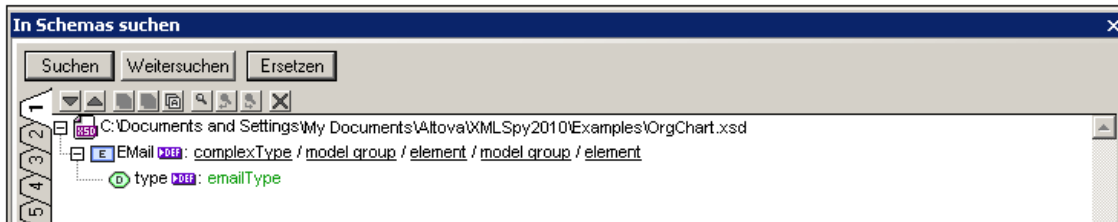
- Eine Zusammenfassung der Suchparameter, in der aufgelistet ist, welcher Suchstring gesucht wurde und welche Dateien durchsucht wurden.
- Eine Liste der gefundenen bzw. ersetzten Strings (je nachdem, ob Sie die Schaltfläche **Suchen** oder **Ersetzen** betätigt haben). Die Einträge in dieser Liste sind 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](#).

2.1.9 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](#) im Abschnitt [Ergebnisse & Informationen](#).

2.1.10 Menüleiste, Symbolleiste und Statusleiste

Menüleiste

Die Menüleiste ([siehe Abbildung](#)) 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](#).

Symbolleisten

Die Symbolleisten ([siehe Abbildung](#)) 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](#) 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](#) 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](#)). 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 XMLSpy verwenden, wird in der Statusleiste das Suffix (x64) hinter dem Applikationsnamen angezeigt. Für die 32-Bit-Version gibt es kein Suffix.

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

2.2.1 Einstellungen, Anpassung und Menüs

Sie können XMLSpy mittels verschiedener Einstellungen und Optionen konfigurieren. 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](#)
- [Anpassung](#)
- [Menus](#)

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](#)). Gehen Sie die verschiedenen Optionen selbst durch, um zu sehen, welche Optionen zur Verfügung stehen.

The screenshot shows the 'Options' dialog box in XMLSpy, organized into several sections:

- Öffnen/Neue Datei in Grid-Ansicht**: Includes a checkbox for 'Alle Zeilen erweitern'.
- Auto-Laden von geänderten Dateien**: Includes checkboxes for 'Dateien überwachen' and 'Vor Neuladen fragen'.
- Validierung**: Includes checkboxes for 'Dateien automatisch validieren' (with sub-options 'Beim Öffnen' and 'Beim Speichern'), a 'Bis zu Größe' dropdown set to '10 MB', and a checkbox for 'DTD/Schema-Dateien im Cache'.
- Projekt**: Includes a checkbox for 'Letztes Projekt beim Programmstart öffnen'.
- Datei speichern**: Includes checkboxes for 'Kommentar: "Mit XMLSpy bearbeitet" einfügen', 'Inkludiere in Diagramme: "Erstellt mit XMLSpy"', and 'Authentic: Link zu Design-Datei speichern'.
- Zeilenumbruch:**: Includes radio buttons for 'Alte beibehalten', 'CR & LF', 'CR', and 'LF'.
- Keine Ausgabe-Formatierung für:**: A list box containing 'xsl:attribute'.

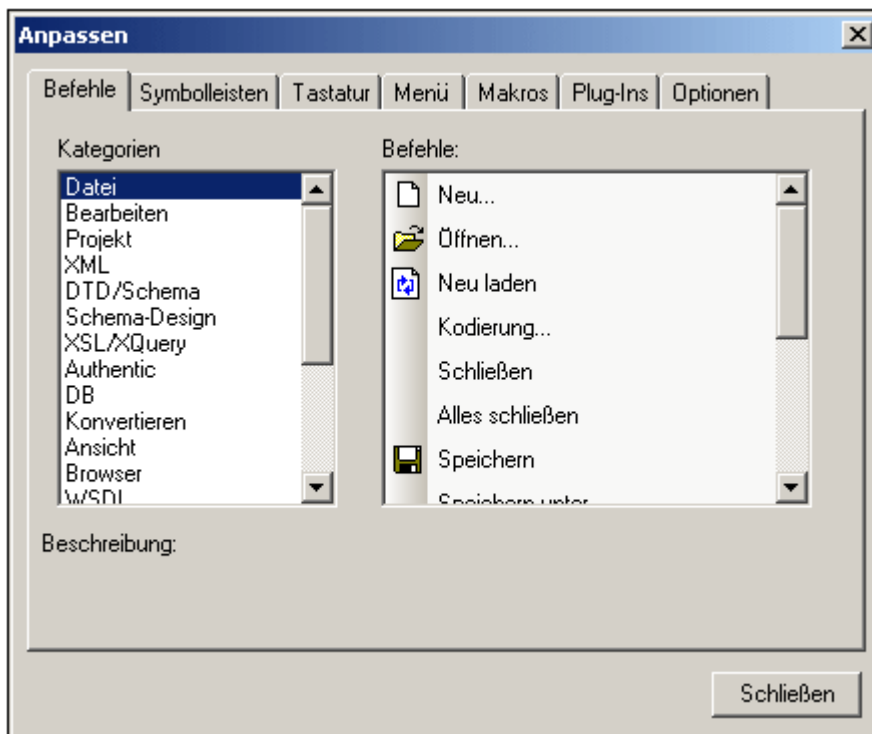
Im Folgenden finden Sie eine Zusammenfassung der wichtigsten Einstellungen. Nähere Informationen dazu finden Sie im Abschnitt "Benutzerreferenz" in der Beschreibung des

Dialogfelds "Optionen".

- *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 Schriftarteneigenschaften 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](#).
- *Kodierung:* Auf dem Register "Kodierung" wird die Standardkodierung für XML- und nicht-XML-Dateien definiert.

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](#)).



Die verschiedenen Konfigurationsoptionen sind im Abschnitt [Benutzerreferenz](#) beschrieben.

Menüs

Menübefehle sind abhängig von drei Faktoren aktiv/deaktiviert: (i) Dateiarart, (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](#)) und sie durch Ziehen auch an eine andere Stelle der Benutzeroberfläche ziehen können (siehe [Menüleiste](#), [Symbolleisten](#), [Statusleiste](#)).

2.2.2 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:\Documents and Settings\<username>\My Documents\
Altova\XMLSpy2012\Examples\
```

Der Ordner `My Documents\Altova\XMLSpy2012` 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.

Anmerkung zum XMLSpy-Hauptordner

Bei Installation von XMLSpy auf einem Rechner, wird ein Hauptordner `Altova\XMLSpy2012` unter folgendem Pfad angelegt:

```
C:\Documents and Settings\All Users\Application Data\
```

Wenn ein Benutzer XMLSpy auf diesem Rechner zum ersten Mal startet, erstellt XMLSpy eine Kopie dieses Hauptordners im Ordner `<username>\My Documents\`. Es ist daher wichtig, dass Sie beim Arbeiten mit Tutorial- oder Beispieldateien nicht im Hauptordner arbeiten, da die bearbeiteten Dateien sonst in den Benutzerordner eines Benutzers kopiert werden, der XMLSpy zu einem späteren Zeitpunkt zum ersten Mal startet.

Pfade der Tutorial-, Projekt- und Beispieldateien

Alle Tutorial-, Projekt- und Beispieldateien befinden sich im Ordner `Examples\AuthenticExamples`. 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 der Datei `Examples.spp` definiert. Diese Datei befindet sich im Ordner `Examples`.
- *Beispieldateien*: befinden sich im Ordner `Examples` und in den Unterordnern dieses Ordners.

2.2.3 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, Professional, und Standard) unterstützt werden. Außerdem finden Sie auf der Website eine Liste aller Features, die seit der vorigen Release neu hinzugekommen sind.

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](#).

3 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:

- [Erstellen eines XML-Schemas](#). 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. Enthält eine Einführung in die Arbeit mit XML-Schemas und den verschiedenen Ansichten, die in XMLSpy zum Anzeigen und Bearbeiten von XML-Schemas zur Verfügung stehen.
- Erstellen von complex und simpleTypes, globalen Elementreferenzen und Attributenumerationen [mit Hilfe von Funktionen der Schema-Ansicht](#).
- Sie lernen, wie man in der Schema-Ansicht in [Schemas navigiert](#) und wie man [Dokumentation zu Schemas generiert](#).
- [Erstellen eines XML-Dokuments](#). 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](#). Dabei müssen Sie eine XSLT-Datei zuweisen und die Transformation mit Hilfe des in XMLSpy integrierten XSLT-Prozessors durchführen.
- [Arbeiten mit XMLSpy Projekten](#), 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\<Benutzername>\Eigene  
Dateien\Altova\XMLSpy2012\Examples\Tutorial
```

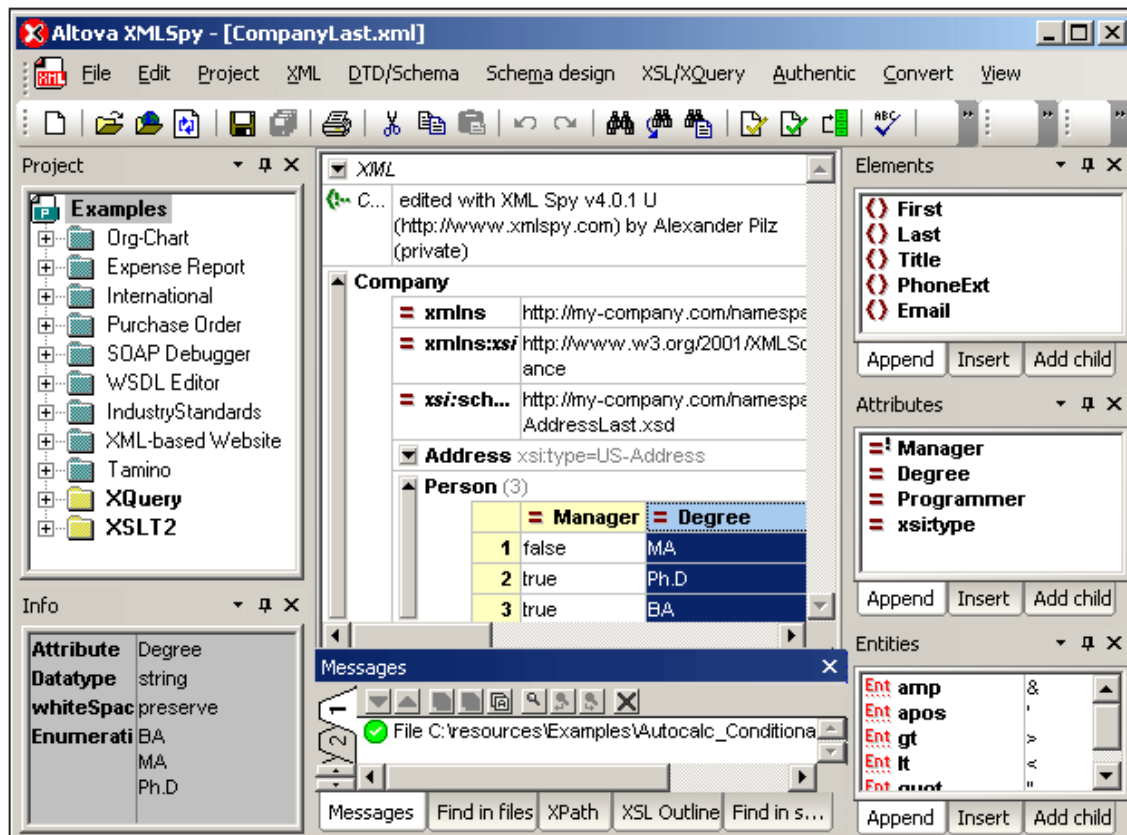
Im Verzeichnis `Examples` finden Sie viele XML-Dateien, mit denen Sie experimentieren können. Im Verzeichnis `Tutorial` sind alle Dateien abgelegt, die in dieser Anleitung verwendet werden.

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.

3.1 XMLSpy-Benutzeroberfläche

Die Oberfläche von XMLSpy ist vertikal in drei Bereiche gegliedert. Im mittleren Bereich stehen verschiedene Ansichten für Ihr Dokument bereit. Die beiden seitlichen Bereiche enthalten Informationen, Editierhilfen und Funktionen zur Dateiverwaltung.

- Das **Projekt-** und das **Info-Fenster** befinden sich auf der linken Seite.
- Der mittlere Teil besteht aus dem **Hauptfenster**. Hier bearbeiten Sie Ihr XML-Dokument. Sie können zwischen verschiedenen Ansichten wählen: [Textansicht](#), [Grid-Ansicht](#), [Schema-Ansicht](#), WSDL-Ansicht, [Authentic-Ansicht](#) und [Browser-Ansicht](#). Diese Ansichten werden in den entsprechenden Abschnitten dazu näher beschrieben.
- Auf der rechten Seite des Bildschirms befinden sich drei **Eingabenhilfen-Fenster**. Hier können Sie Elemente, Attribute oder Entities einfügen oder anhängen. Die verfügbaren Einträge ändern sich je nachdem, welche Auswahl in der Grid-Ansicht vorgenommen wurde, bzw. je nach Positionierung des Cursors in der XML-Datei.



Die Einzelheiten der Benutzeroberfläche werden im Laufe des Handbuchs näher beschrieben. Beachten Sie, dass die Oberfläche sich dynamisch je nach gewählter Ansicht ändert und je nachdem, welches Dokument im Hauptfenster aktiv ist.

3.2 Erstellen eines einfachen XML-Schemas

Ein XML-Schema beschreibt die Struktur eines XML-Dokuments. Ein XML-Dokument kann gegen ein XML-Schema 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-Schema-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 Abschnitt lernen Sie, wie man ein XML-Schemas in der Schema-Ansicht bearbeitet. Insbesondere wird auf folgende Punkte eingegangen:

- Erstellen einer neuen Schema-Datei
- Definieren von Namespaces für das Schema
- Definieren eines einfachen Content Model
- Hinzufügen von Elementen zum Content Model über Kontextmenüs und mittels Drag & Drop
- Konfiguration der Content Model-Ansicht

Nachdem Sie mit der Erstellung des Schema-Grundgerüsts fertig sind, können Sie zum [nächsten Abschnitt des Tutorials](#) übergehen, in dem Sie erfahren, wie man mit komplexeren XML-Schema-Funktionen in XMLSpy arbeitet. Auf diesen Abschnitt folgt ein Abschnitt über [Schema-Navigation und Dokumentation](#) in XMLSpy.

In diesem Abschnitt verwendete Befehle

In diesem Abschnitt arbeiten Sie ausschließlich in der Schema-Ansicht. Dabei werden die folgenden Befehle verwendet:



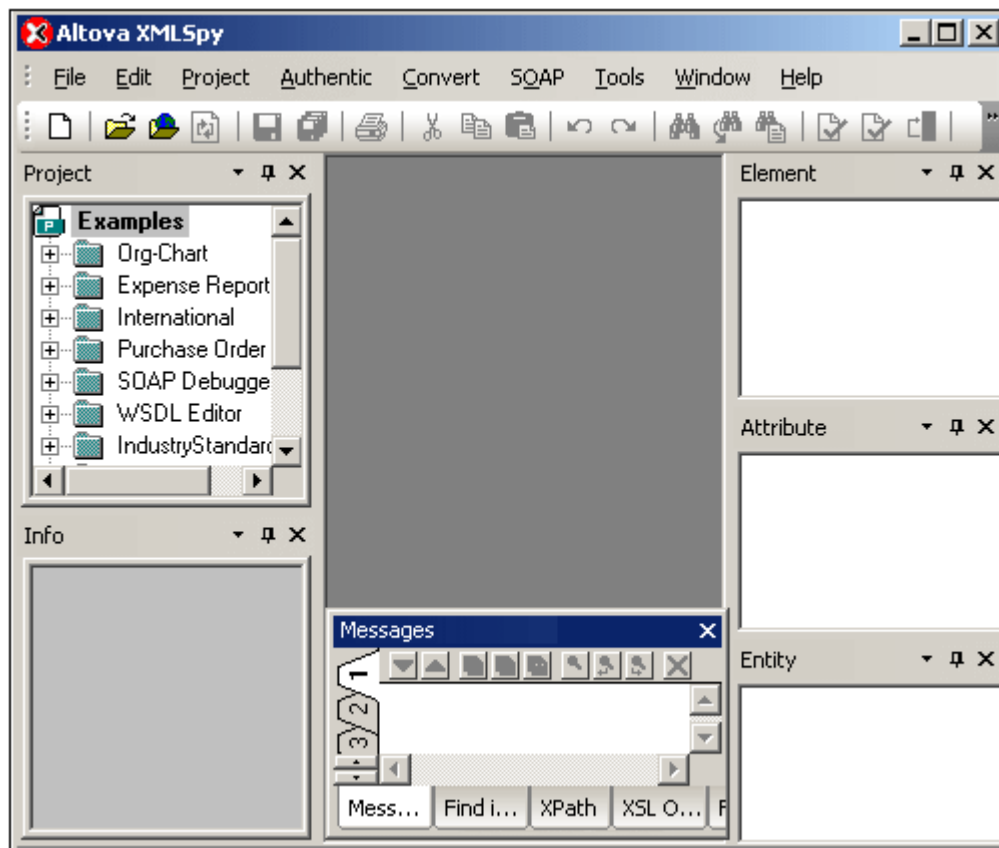
Diagramm anzeigen (oder Content Model-Ansicht anzeigen). Dieses Symbol befindet sich links von allen globalen Komponenten in der Schema-Übersicht. Wenn Sie auf das Symbol klicken, wird das Content Model der dazugehörigen globalen Komponente angezeigt.

3.2.1 Anlegen einer neuen XML-Schema-Datei

Um in XMLSpy eine neue XML-Schema-Datei anzulegen, müssen Sie zuerst XMLSpy starten und dann ein neues XML-Schema (.xsd)-Dokument erstellen.

Starten von XMLSpy

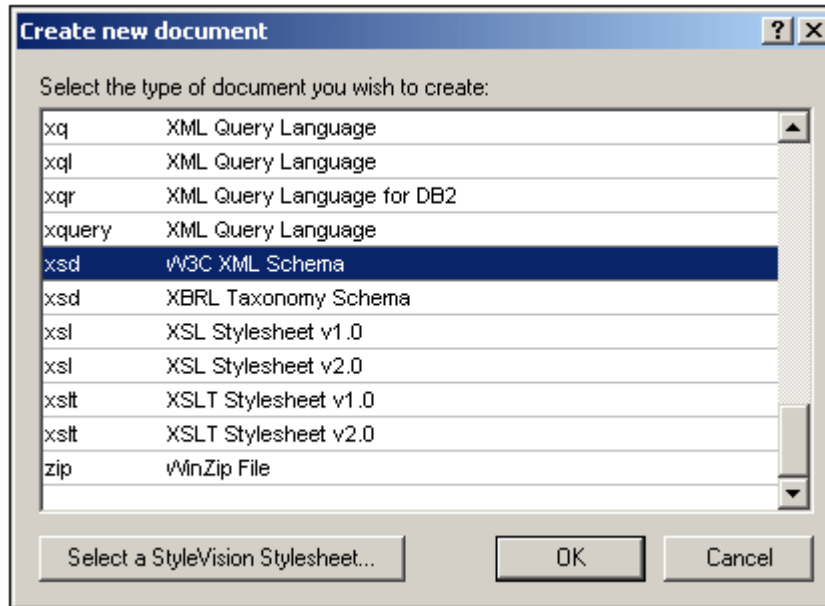
Doppelklicken Sie zum Starten von XMLSpy auf das XMLSpy-Symbol auf Ihrem Desktop oder rufen Sie XMLSpy über **Start | Programme** auf. XMLSpy wird mit einem leeren Hauptfenster gestartet, in dem kein Dokument geöffnet ist.



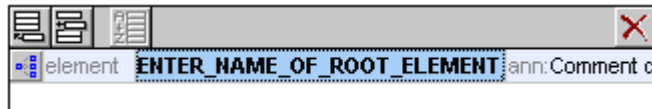
Beachten Sie, dass sich die Benutzeroberfläche in vier Teile gliedert: (i) das Projekt- und das Info-Fenster auf der linken Seite; (ii) das Hauptfenster in der Mitte; (iii) die Eingabehilfen auf der rechten Seite und (iv) die Ausgabefenster am unteren Rand.

So legen Sie eine neue Schema-Datei an:

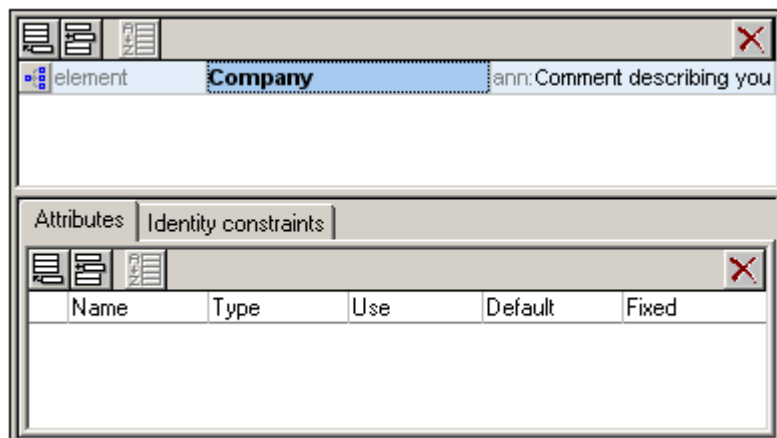
1. Wählen Sie die Menüoption **Datei | Neu**. Das Dialogfeld "Neues Dokument anlegen" wird angezeigt.



2. Wählen Sie im Dialogfeld den Eintrag `xsd W3C Schema`. (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. Sie werden aufgefordert, den Namen des Root-Elements anzugeben.



3. 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 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.)



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

3.2.2 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.

Schema settings

Default Element form: ☒ Qualified ☐ Unqualified

Default Attribute form: ☐ Qualified ☒ Unqualified

Block default:

Final default:

Version:

xml:lang: ID:

☐ No target namespace

☒ Target namespace:

Prefix	Namespace
	http://my-company.com/namespace
xs	http://www.w3.org/2001/XMLSchema

OK Cancel

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.

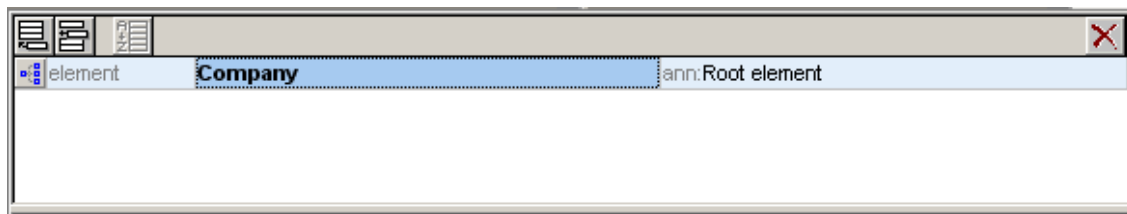
Bitte beachten Sie:

- 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.

3.2.3 Definieren eines Content Model

Sie haben in der Schema-Übersicht bereits ein globales Element mit dem Namen `Company` erstellt. Dieses Element soll ein `Address`-Element und eine unbegrenzte Anzahl an `Person`-Elementen enthalten, nämlich das Content Model. 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.

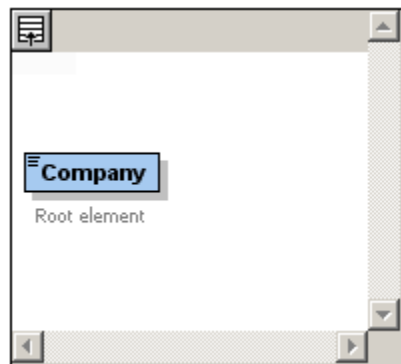


In diesem Abschnitt werden Sie das Content Model des Elements `Company` erstellen.

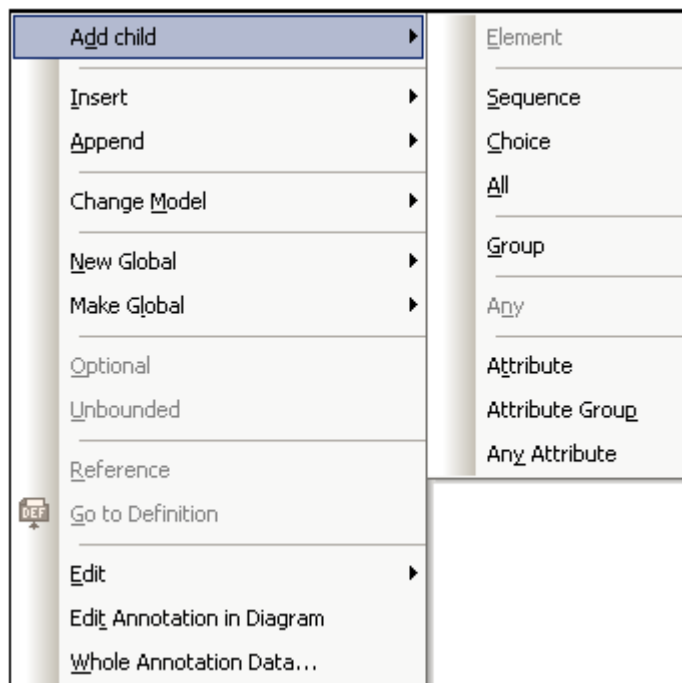
Erstellen eines einfachen Content Model

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

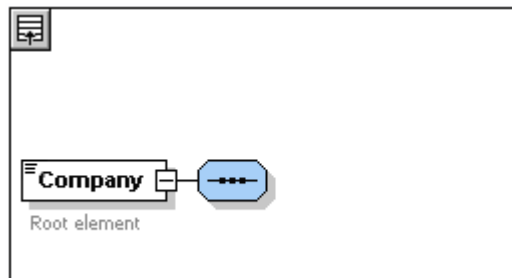
1. Klicken Sie in der Schema-Übersicht auf das Symbol "Diagramm anzeigen"  links neben dem Element `Company`. Hierdurch wird das Content Model des Elements `Company` angezeigt, das derzeit noch leer ist. (Alternativ dazu können Sie auch auf den Eintrag `Company` im Komponentenfenster doppelklicken, um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



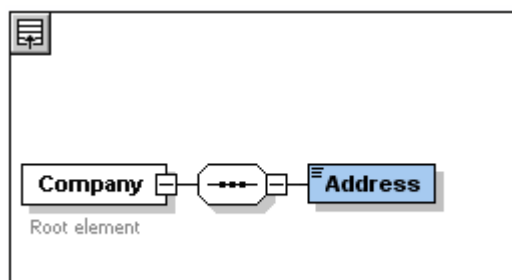
2. Ein Content Model besteht aus Kompositoren und Komponenten. Die Kompositoren definieren die Beziehung zwischen zwei Komponenten. Sie müssen nun im `Company` Content Model einen Child-Kompositor zum Element `Company` hinzufügen, um ein Child-Element hinzufügen zu können. Rechtsklicken Sie dazu auf 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.)



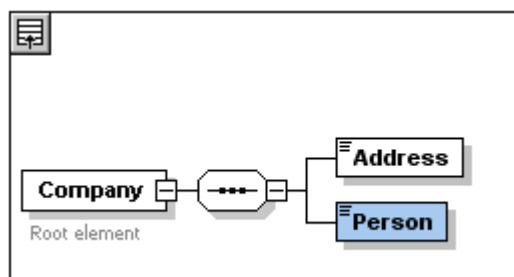
Daraufhin wird der Sequence-Kompositor eingefügt, der definiert, dass die Komponenten, die darauf folgen in der angegebenen Reihenfolge erscheinen müssen.



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.

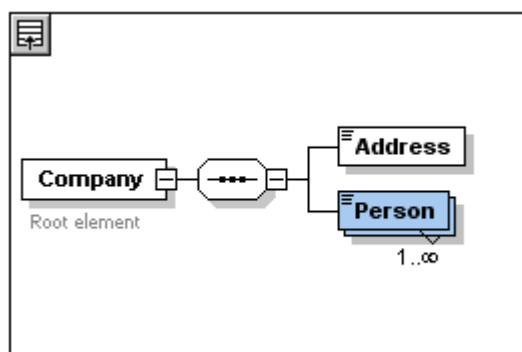


5. Rechtsklicken Sie erneut auf den Sequence-Kompositor und wählen Sie **Child hinzufügen | Element**, benennen Sie die neue Elementkomponente `Person`.



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.

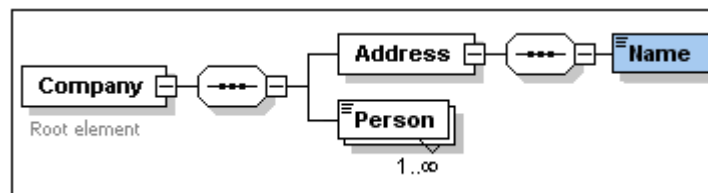


Sie können alternativ dazu in der Details-Eingabehilfe die Felder `minOcc` und `maxOcc` direkt editieren und die Anzahl der zulässigen Instanzen angeben, in diesem Fall 1 bzw. unbounded.

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".



ComplexTypes, simpleTypes und XML Schema-Datentypen

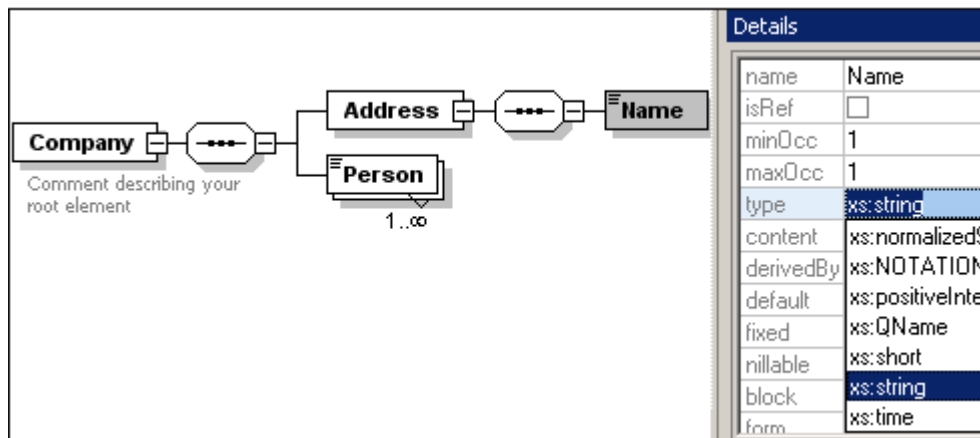
Bisher haben wir noch keinen Elementtyp explizit definiert. Klicken Sie auf das Register **Text**

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"/>
```

Bitte beachten Sie: Ein einfaches Type-Element kann jedem beliebigen von mehreren XML-Schema-Datentypen angehören. In all diesen Fällen erscheint das Symbol für Textinhalt

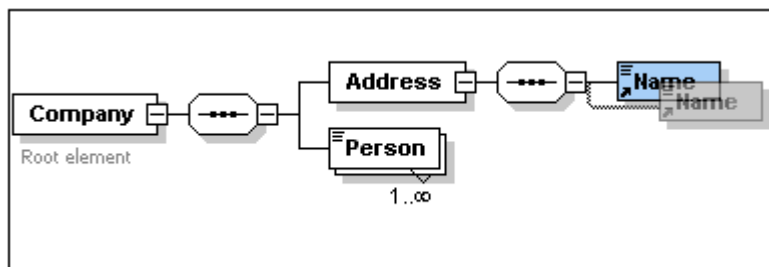
nur im Elementfeld.

3.2.4 Hinzufügen von Elementen mit Drag & Drop

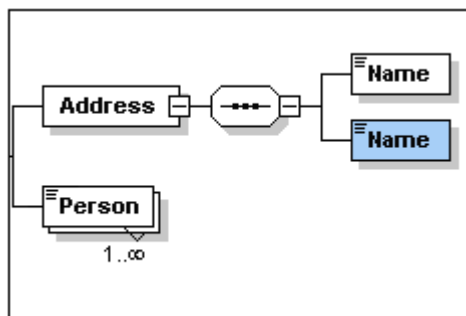
Sie haben Elemente bisher über das Kontextmenü hinzugefügt, das erscheint, wenn Sie mit der rechten Maustaste auf ein Element oder einen Kompositor klicken. Schneller lassen sich Elemente aber mit Drag & Drop erstellen. In diesem Abschnitt lernen Sie, wie man mit Drag & Drop mehr Elemente zur Definition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

So stellen Sie die 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.

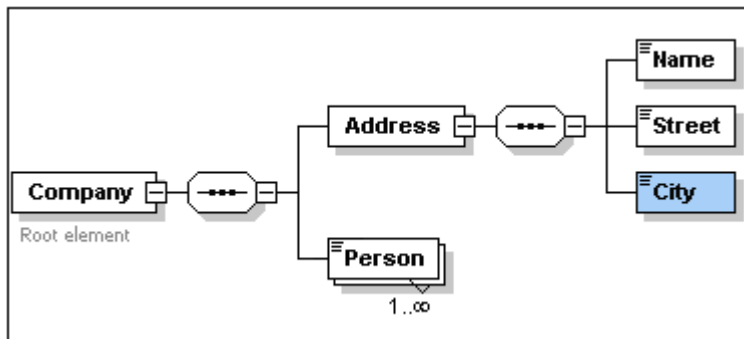


2. Lassen Sie nun die Maustaste und die STRG-Taste los um das neue Element in der Address-Sequenz anzulegen. Wurde das Element an der falschen Stelle angelegt, so ziehen Sie es einfach mit der Maus unter das Element Name.



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:



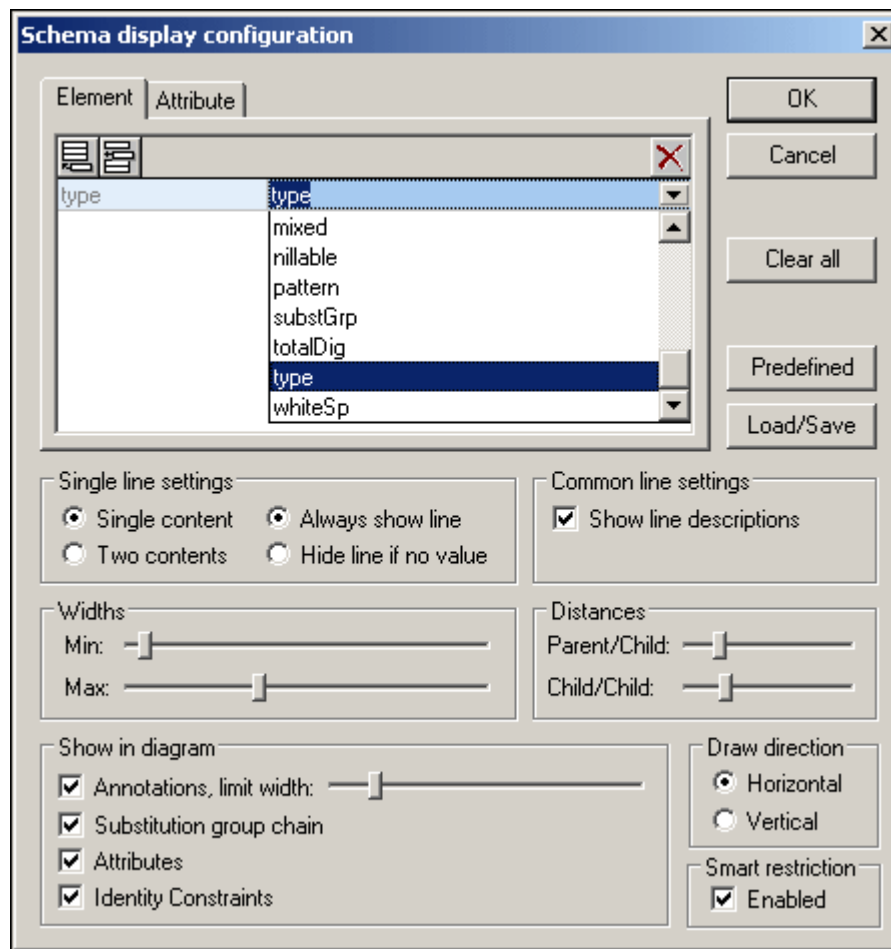
Das Element Address enthält eine Sequence mit einem Element Name, einem Element Street und einem Element City, und zwar in der obigen Reihenfolge.

3.2.5 Konfigurieren der Content Model-Ansicht

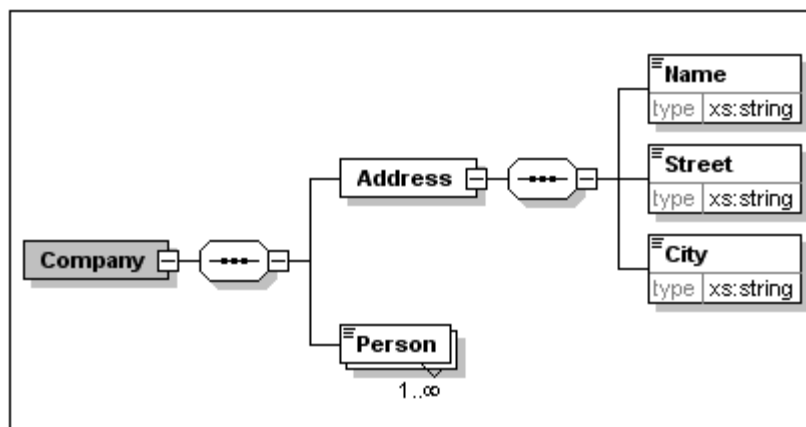
Sie können jetzt das Aussehen der Content Model-Ansicht konfigurieren. Wir wollen die Content Model-Ansicht so konfigurieren, dass der Elementtyp für jedes einzelne Element angezeigt wird.

So konfigurieren Sie die Content Model-Ansicht:

1. Gehen Sie zur Content Model-Ansicht einer Komponente (klicken Sie auf das Symbol ) , um den Befehl "Ansicht konfigurieren" zu aktivieren.
2. Wählen Sie die Menüoption Schema-Design | Ansicht konfigurieren, um das Dialogfeld "Schema-Ansicht konfigurieren" aufzurufen.

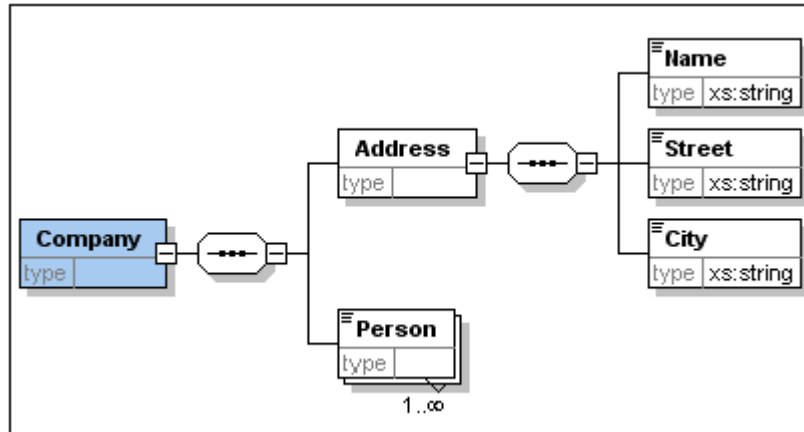


3. Klicken Sie im Element/Attribut-Fenster auf dem Register "Element" auf das Symbol  um eine Zeile zur Beschreibung einer Eigenschaft zu jedem Elementfeld hinzuzufügen.
4. Wählen Sie type aus der Auswahlliste (oder doppelklicken Sie in die Zeile und tippen Sie "type"). Dadurch wird der Datentyp jedes Elements in der Content Model-Ansicht angezeigt.
5. 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, 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.

6. Aktivieren Sie im Bereich "Zeileinstellungen (Gilt pro Zeile)" die Option "Immer zeigen".
7. 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.

Bitte beachten Sie:

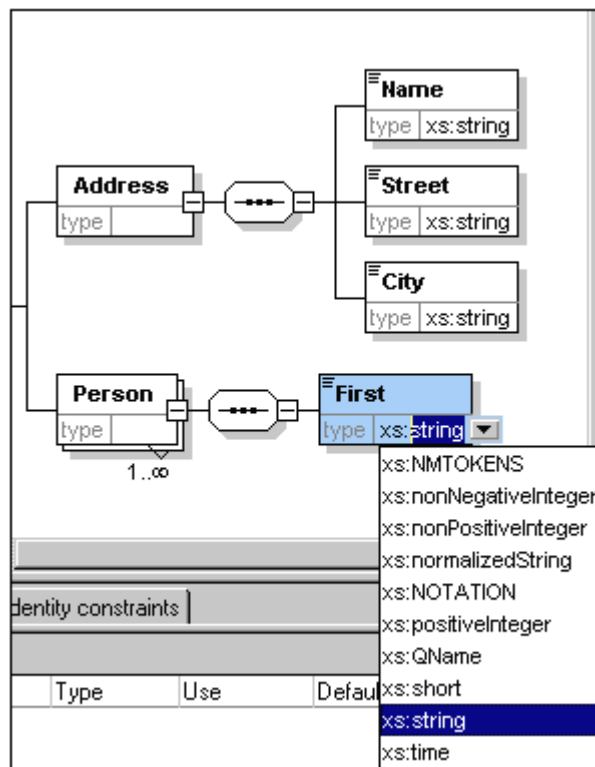
- Die Eigenschaftsbeschreibungszeilen können editiert werden. Werte, die Sie dort eingeben, werden Teil der Elementdefinition.
- Die Einstellungen, die Sie im Dialogfeld "Schema-Ansicht konfigurieren" vornehmen, gelten für die Ausgabe der Schema-Dokumentation ebenso wie für die Druckausgabe.

3.2.6 Fertigstellen des Schemas

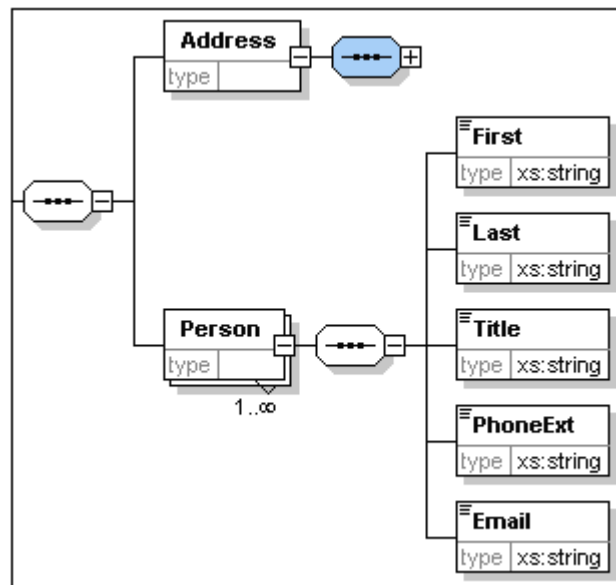
Nachdem Sie nun den Inhalt des Elements Address definiert haben, müssen Sie den Inhalt des Elements Person definieren. Das Element Person soll die folgenden Child-Elemente - alle vom Typ SimpleType - enthalten: First, Last, Title, PhoneExt und Email. Alle diese Elemente sind obligatorisch mit Ausnahme des Elements Title (das optional ist), und alle Elemente müssen in der oben angegebenen Reihenfolge aufscheinen. Alle Elemente sollen vom Datentyp xs:string sein, mit Ausnahme von PhoneExt, das vom Datentyp 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 Wertefeld von type ein.
5. Erzeugen Sie weitere vier Elemente mittels der Drag & Drop Methode. Benennen Sie diese der Reihe nach Last, Title, PhoneExt und Email.

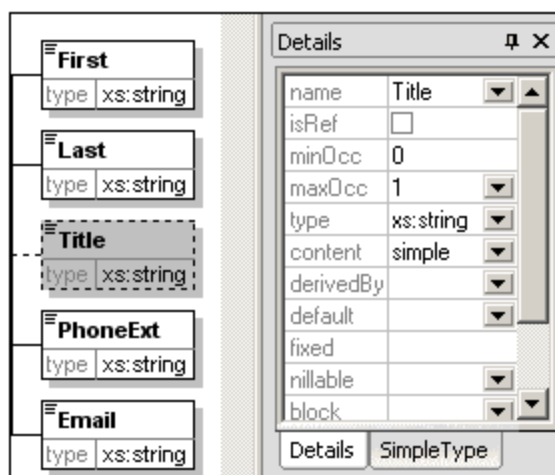


Bitte beachten Sie: Sie können mehrere Elemente auswählen, indem Sie die Strg-Taste gedrückt halten, während Sie die Elemente anklicken. Auf diese Art können Sie mehrere Elemente gleichzeitig kopieren.

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.

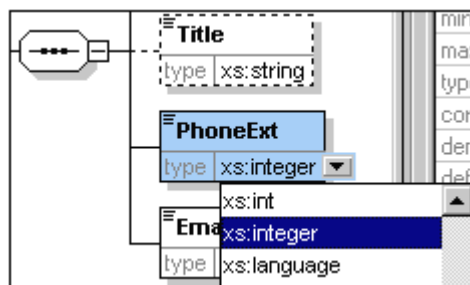


Sie sehen in der Details-Eingabehilfe, dass minOccurs=0 und maxOccurs=1 ist, d.h. das Element ist optional. Alternativ dazu können Sie das Element auch über das Kontextmenü optional machen und für minOccurs=0 einstellen, um das Element zu einem optionalen zu machen.

Den Inhalt eines Elements einschränken

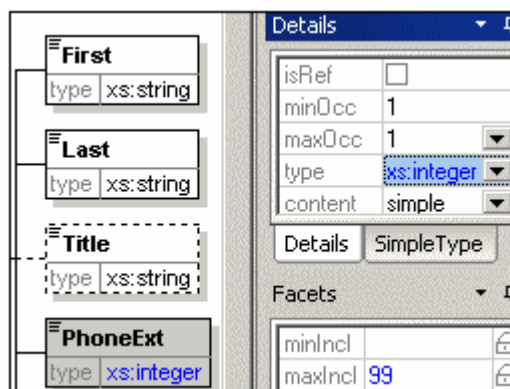
Um das Element PhoneExt als Typ xs:integer mit maximal 2 Stellen zu definieren, gehen Sie folgendermaßen vor:

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.



Änderungen werden in der Facets-Eingabehilfe angezeigt.

2. 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 alle Durchwahlnummern bis inkl. 99 gültig sind.

3. Wählen Sie die Menüoption Datei | Speichern zum Speichern der Änderungen im Schema.

Bitte beachten Sie:

- 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 Verzeichnis C:\Dokumente und Einstellungen\<Benutzername>\Eigene Dateien\Altova\XMLSpy2012\\Examples\Tutorial Ihres XMLSpy-Anwendungsordners abgelegt.

3.3 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 [complexType](#) und [simpleTypes](#), die später als die Typen von Schema-Elementen verwendet werden können.
- Erstellen von [globalen Elementen](#) und Referenzieren dieser Elemente von anderen Elementen aus
- Erstellen von [Attributen](#) 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 kommen 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.

3.3.1 Arbeiten mit ComplexTypes und SimpleTypes

Nachdem Sie ein Element definiert haben, möchten Sie es möglicherweise auch an einer anderen Stelle im Schema verwenden. Dies erreichen Sie dadurch, dass Sie diese Elementdefinition als globalen ComplexType oder als globales Element erstellen. In diesem Abschnitt werden Sie mit globalen ComplexTypes arbeiten. Wir werden zuerst einen ComplexType auf globaler Ebene erstellen und diesen dann erweitern, um ihn in einem Content Model verwenden zu können. Mehr über globale Elemente erfahren Sie weiter hinten in diesem Tutorial.

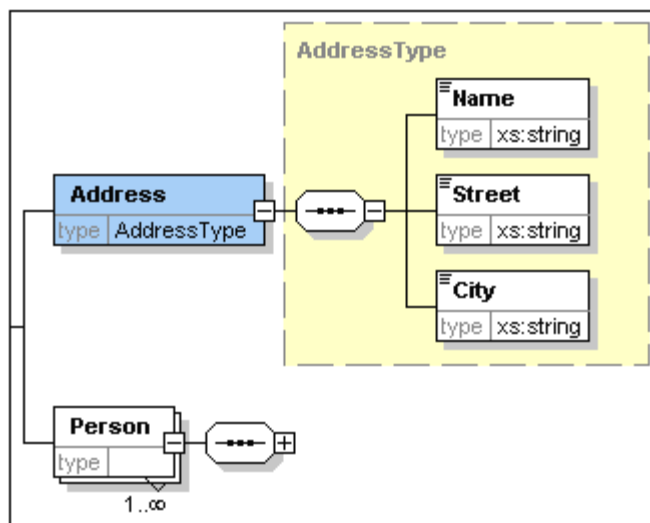
Erstellen eines globalen complex type

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:

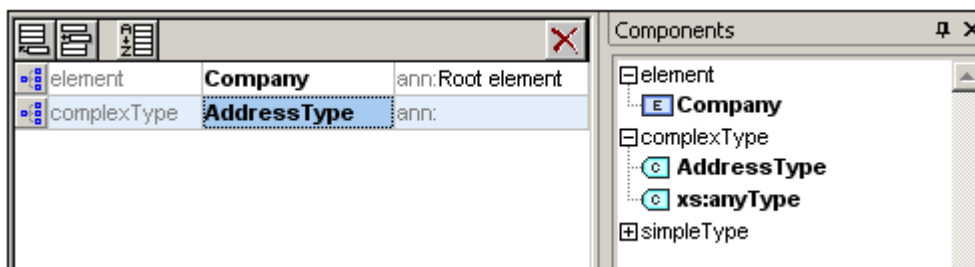
So erstellen Sie einen globalen Element vom Typ Complex Type:

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

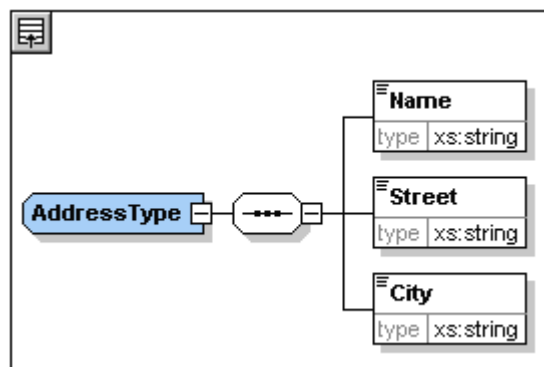
Inhalt des Elements "Address" ist das Content Model von AddressType, das in einem gelb hinterlegten Kasten angezeigt wird. Beachten Sie, dass der Datentyp des Elements Address nun AddressType ist.



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 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.



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.



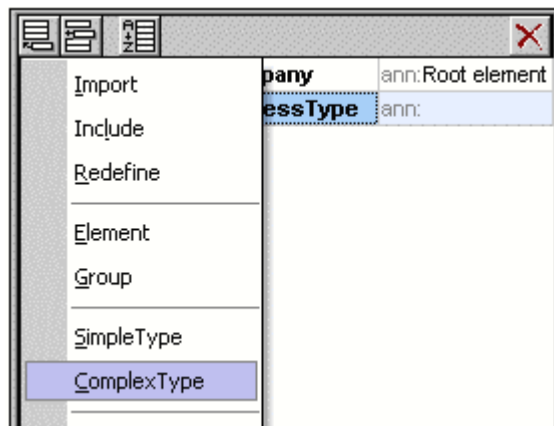
6. Klicken Sie auf das Symbol "Globale Elemente anzeigen", um zur Schema-Übersicht

zurückzukehren.

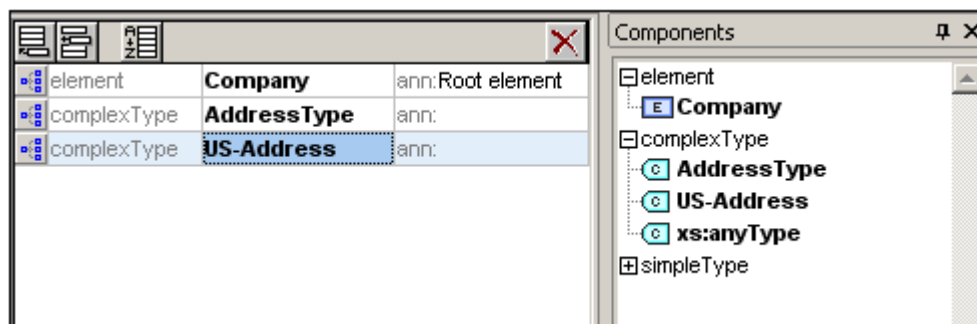
Erweitern einer "complex type" Definition

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:

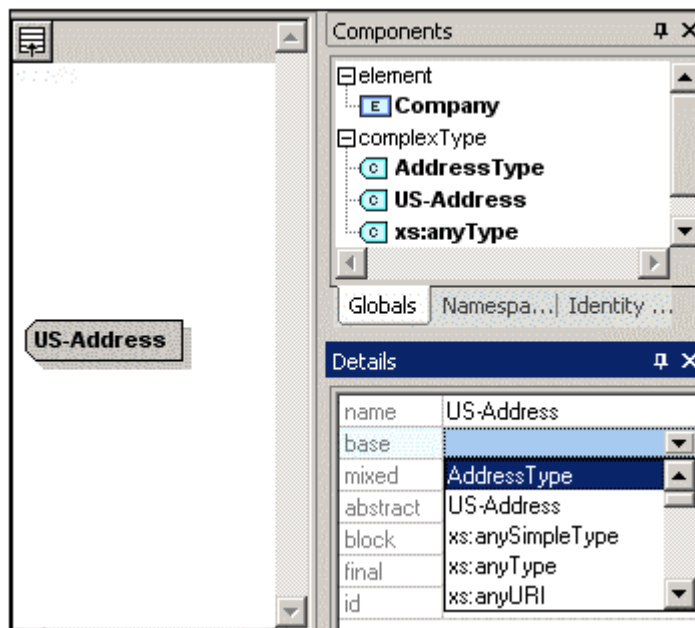
1. Wechseln Sie zur Schema-Übersicht (Wenn Sie sich in der Content Model-Ansicht befinden, klicken Sie auf "Globale Elemente anzeigen").
2. Klicken Sie auf das Symbol Anhängen (oben links im Komponentenfenster). Daraufhin wird das folgende Menü angezeigt:



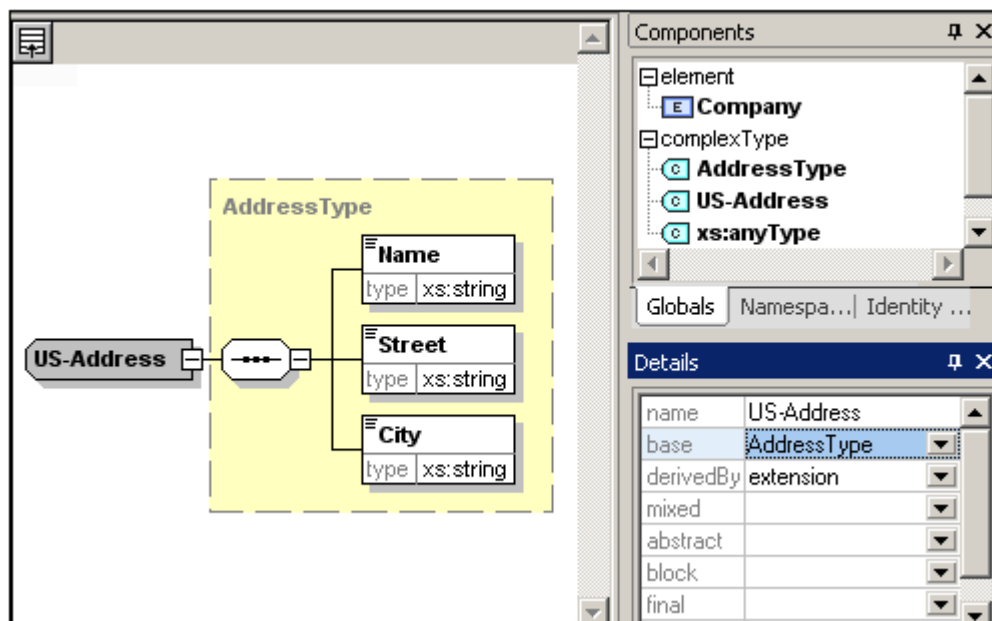
3. Wählen Sie im Menü ComplexType aus. Eine neue Leerzeile wird in der Komponentenliste angehängt, in der Sie den Namen der Komponente eingeben können.
4. Geben Sie US-Address ein und bestätigen Sie mit der Eingabetaste. (Sollten Sie den Bindestrich "-" beim Eingeben vergessen, und stattdessen ein Leerzeichen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



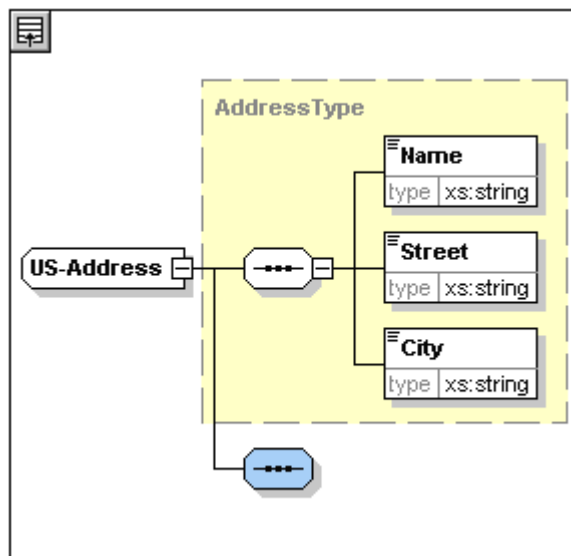
5. Klicken Sie auf das Symbol "Content Model-Ansicht"  in der Zeile US-Address, um das Content Model des neuen ComplexType zu sehen. Das Content Model ist leer (siehe Abbildung unten).
6. Klicken Sie in der Details-Eingabehilfe auf das Listefeld base und wählen Sie AddressType.



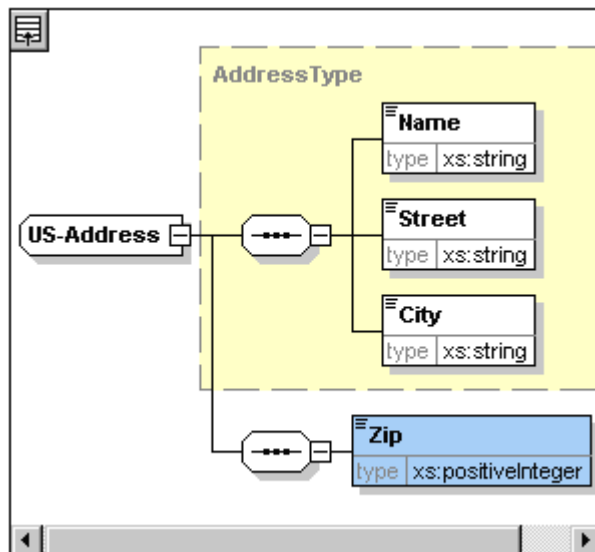
In der Content Model-Ansicht wird nun das Content Model AddressType als Content Model von US-Address (siehe Abbildung unten) angezeigt.



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 Typbeschreibungseile 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

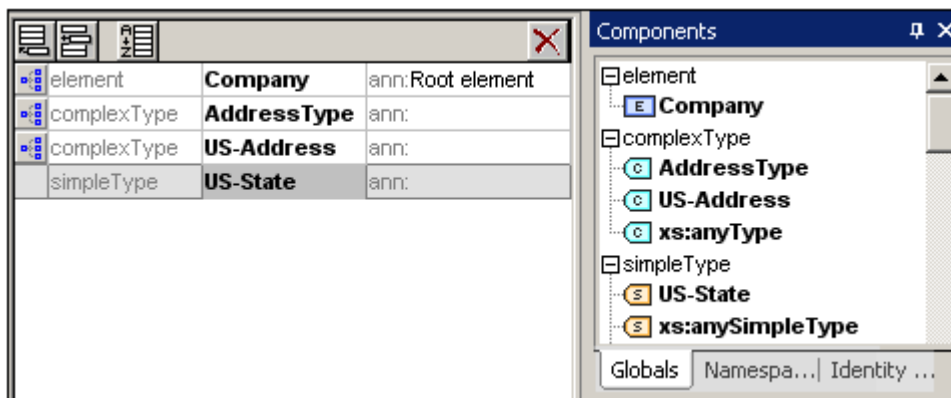
Genau wie der ComplexType US-Address auf dem ComplexType AddressType basiert, kann ein Element auch auf einem SimpleType basieren. Der Vorteil ist derselbe wie bei globalen ComplexTypes: Der SimpleType kann wiederverwendet werden. Dazu muss der SimpleType zuerst als globales Element definiert werden. 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.

Erstellen eines globalen SimpleType

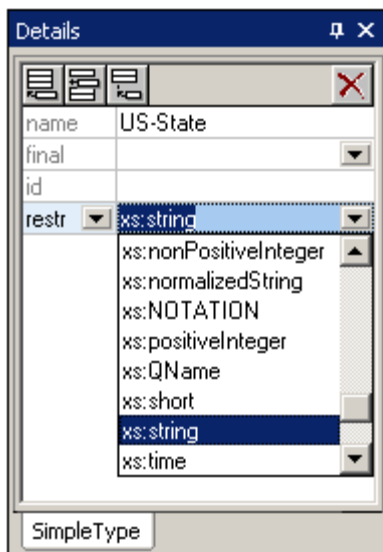
Um einen globalen SimpleType zu erstellen, müssen Sie einen neuen SimpleType an die Liste der globalen Komponenten anhängen, diesen benennen und seinen Datentyp definieren.

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 Wertfeld von restr und geben Sie xs:string ein oder wählen Sie xs:string in der Dropdown-Liste im Wertfeld restr aus.



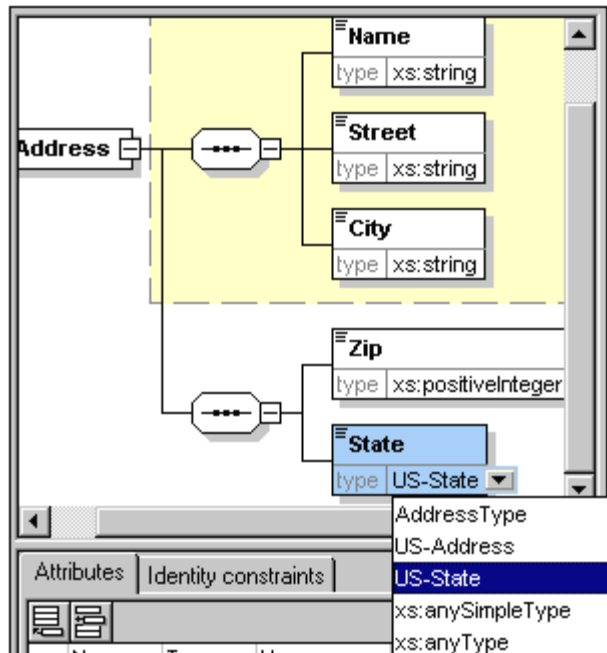
Damit wird ein SimpleType namens US-State vom Datentyp xs:string erstellt. Diese globale Komponente kann nun im Content Model von US-Address verwendet werden.

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.



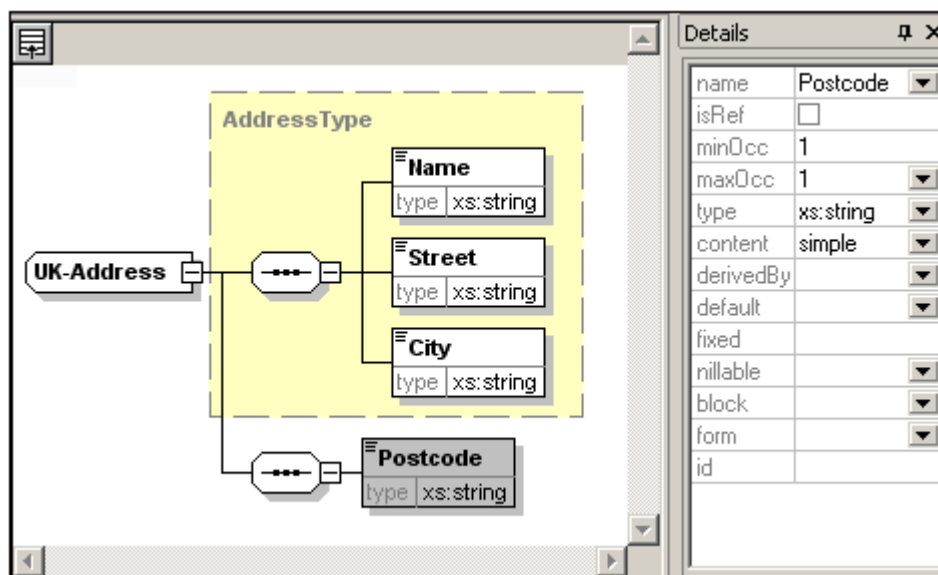
Das Element State basiert nun auf dem SimpleType US-State.

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:

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:

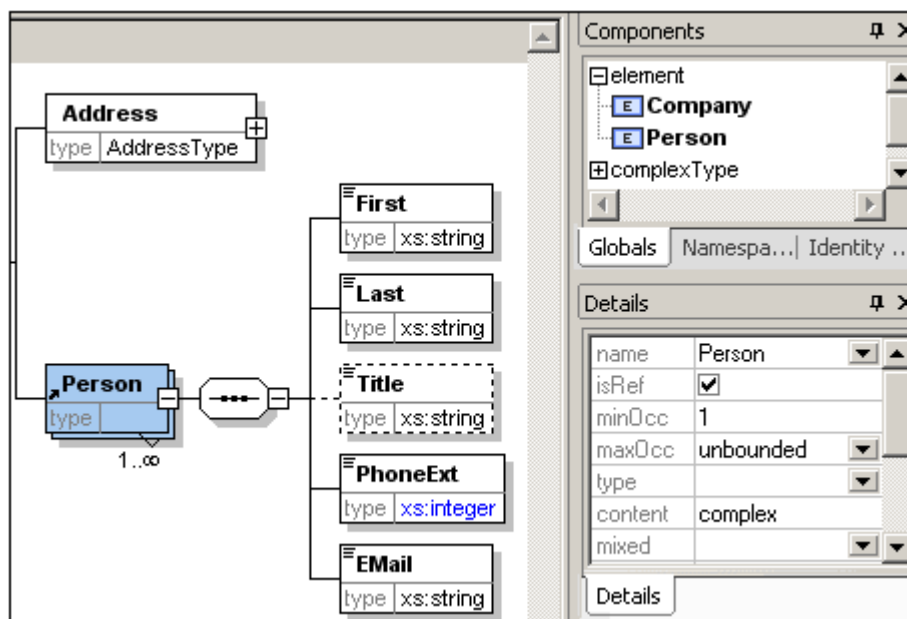


Bitte beachten Sie: In diesem Abschnitt haben Sie globale Simple und ComplexTypes erstellt und anschließend in Content Model-Definitionen verwendet. Der Vorteil von globalen Typen liegt darin, dass diese in mehreren Definitionen wiederverwendet werden können.

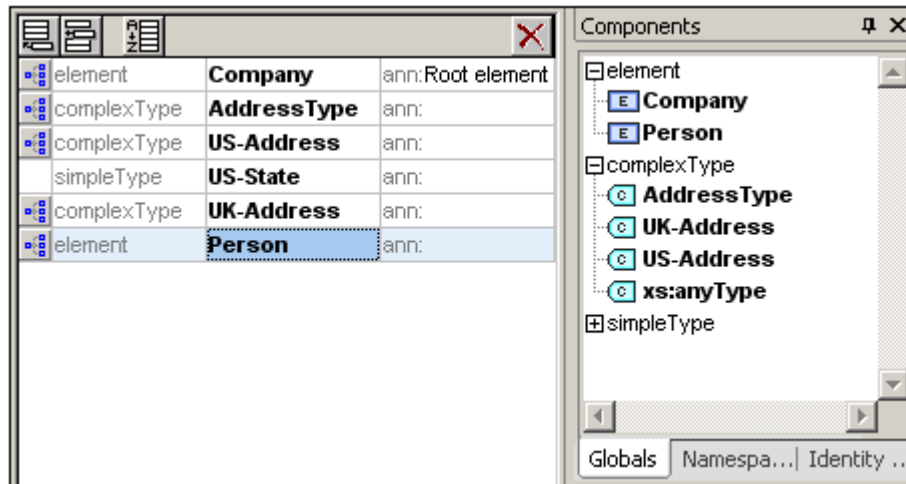
3.3.2 Referenzieren von globalen Elementen

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

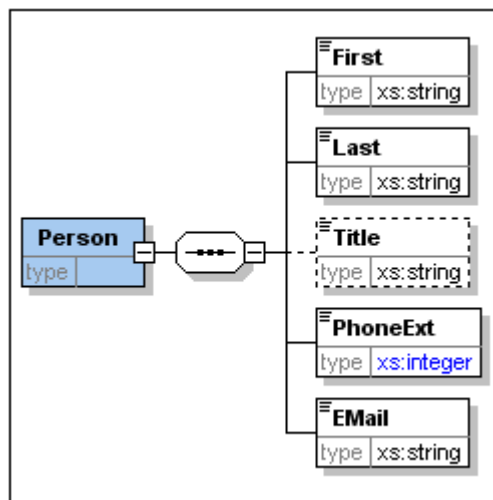
1. Wechseln Sie in die Schema-Übersicht (Globale Elemente anzeigen).
2. Klicken Sie auf das Symbol "Diagramm anzeigen" in der Zeile **Company**.
3. Klicken Sie mit der rechten Maustaste auf das Element **Person** und wählen Sie **Als global definieren | Element**. Ein kleines "link"-Symbol wird in das 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.



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.

Bitte beachten Sie:

- 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

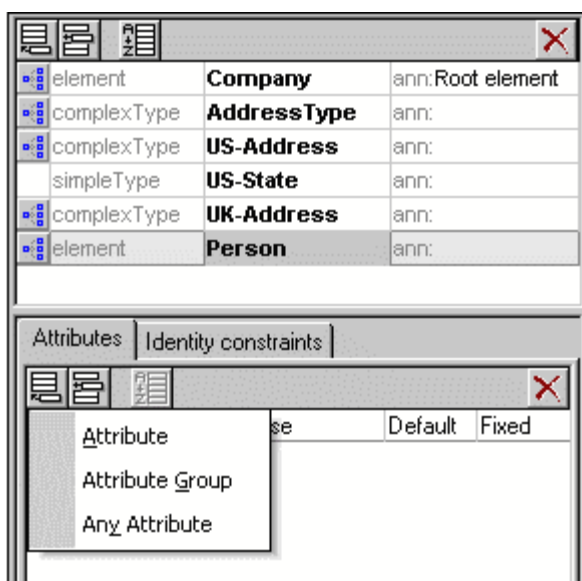
Sie das Content Model eines Elements ändern, das ein globales Element referenziert, so wird auch das Content Model des globalen Elements geändert und damit auch das Content Model aller anderen Elemente, die das globale Element referenzieren.

3.3.3 Attribute und Attribut-Enumerationen

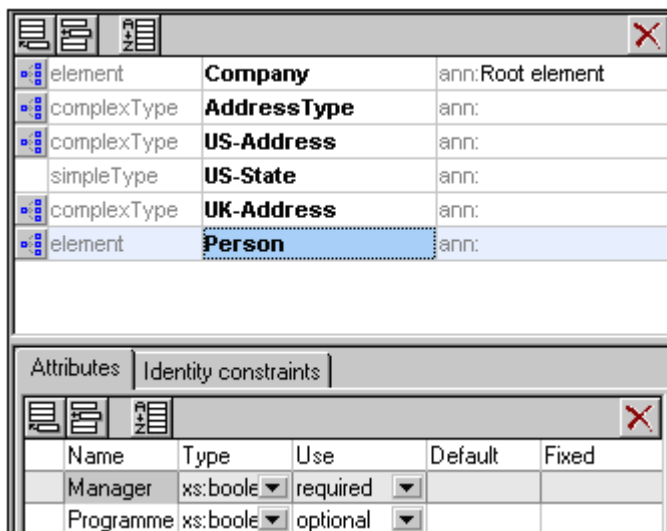
In diesem Abschnitt lernen Sie, wie man Attribute und Enumerationen für Attribute erstellt.

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`.



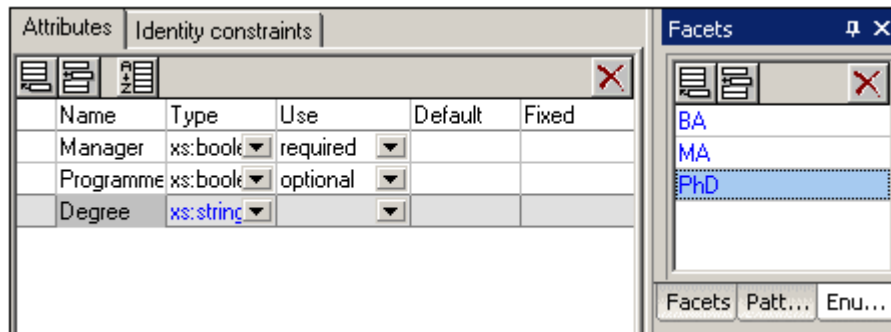
- Erstellen Sie nun auf dieselbe Art ein Attribut Programmer mit einem Type=`xs:boolean` und Use=`optional`.

Definieren von Enumerationen für Attribute

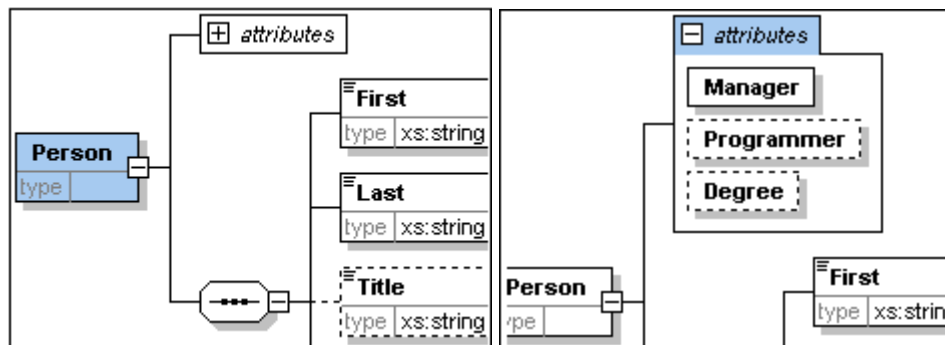
Enumerationen sind Werte, die bei einem bestimmten Attribut zulässig sind. Wenn der Wert des Attributs im XML-Instanzdokument nicht in einer im XML-Schema definierten Enumeration enthalten ist, ist das Dokument ungültig. Wir werden Enumerationen für das Attribut Degree des Elements `Person` erstellen.

Gehen Sie folgendermaßen vor:

- Klicken Sie in der Schemaübersicht auf das Element `Person`, um es zum aktiven zu machen.
- Klicken Sie auf das Anhängen-Symbol oben links im Attribut-Fenster und wählen Sie den Eintrag "Attribute".
- Geben Sie Degree als Attributnamen ein und wählen Sie `xs:string` als Attribut-Type.
- Klicken Sie in der Facets-Eingabehilfe auf das Register Enumerations (siehe Abbildung unten).



- Klicken Sie auf dem Register "Enumerations" auf das Anhängen-Symbol .
- Geben Sie `BA` (Bachelor of Arts) ein und bestätigen Sie mit der Eingabetaste.
- Mit derselben Methode geben Sie zwei weitere Werte in die Enumerations-Liste ein: `MA` und `PhD`.
- 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.

- Klicken Sie auf das Symbol "Globale Elemente anzeigen" , um zur

Schema-Übersicht zurückzukehren.

Speichern des fertigen Schemas

Bitte beachten Sie: 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.

3.4 XML-Schemas: XMLSpy-Funktionen

Nach Fertigstellung Ihres XML-Schemas sollten Sie sich nun mit einigen [Navigationskürzeln](#) vertraut machen und etwas über die [Schemadokumentation](#), 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.

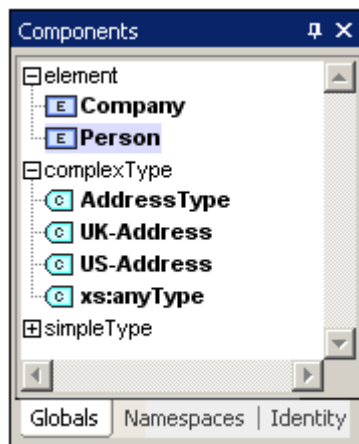
3.4.1 Schemanavigation

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).



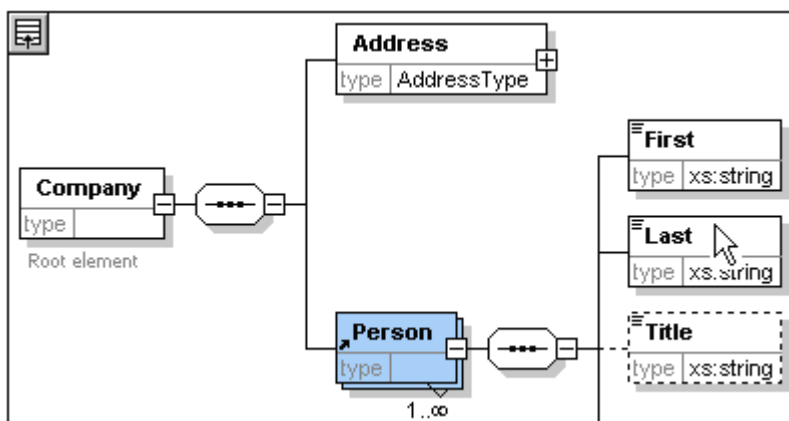
Wenn Sie in der Komponenten-Eingabehilfe auf einer der anderen globalen Komponenten (SimpleType, Attribut, Attributgruppe) doppelklicken, erscheint diese Komponente in der Schema-Übersicht (da diese Komponenten kein Content Model haben) markiert.

Das Anzeigen der Komponenten durch Doppelklick funktioniert in der Komponenten-Eingabehilfe sowohl auf dem Register "By Type" als auch auf dem Register "By Namespace".

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.

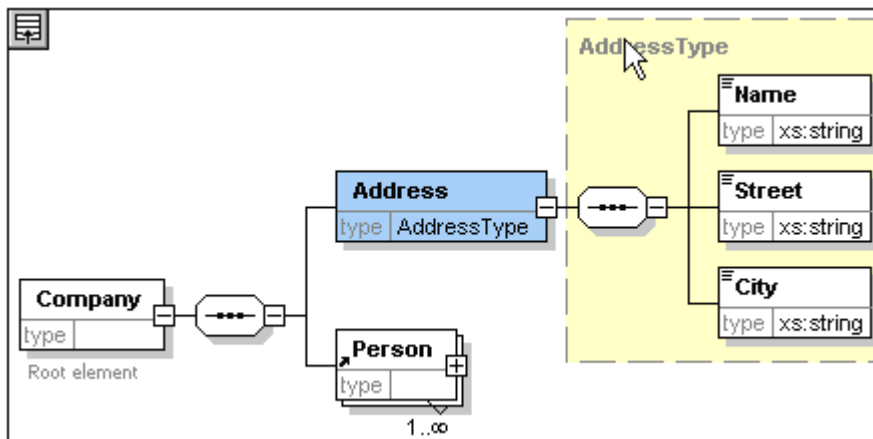
Wenn Sie z.B. während der Ansicht des Content Model von `Company` die **Strg**-Taste gedrückt halten, während Sie auf `Last` doppelklicken, wird das Content Model `Person` geöffnet und das Element `Last` darin wird markiert.



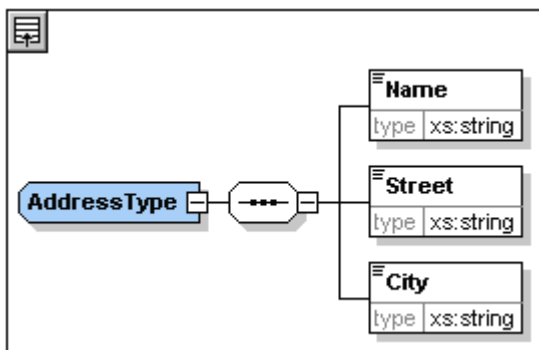
Wenn das Element `Last` markiert ist, werden alle seine Eigenschaften sofort in den entsprechenden Eingabehilfen und Informationsfenstern angezeigt.

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.



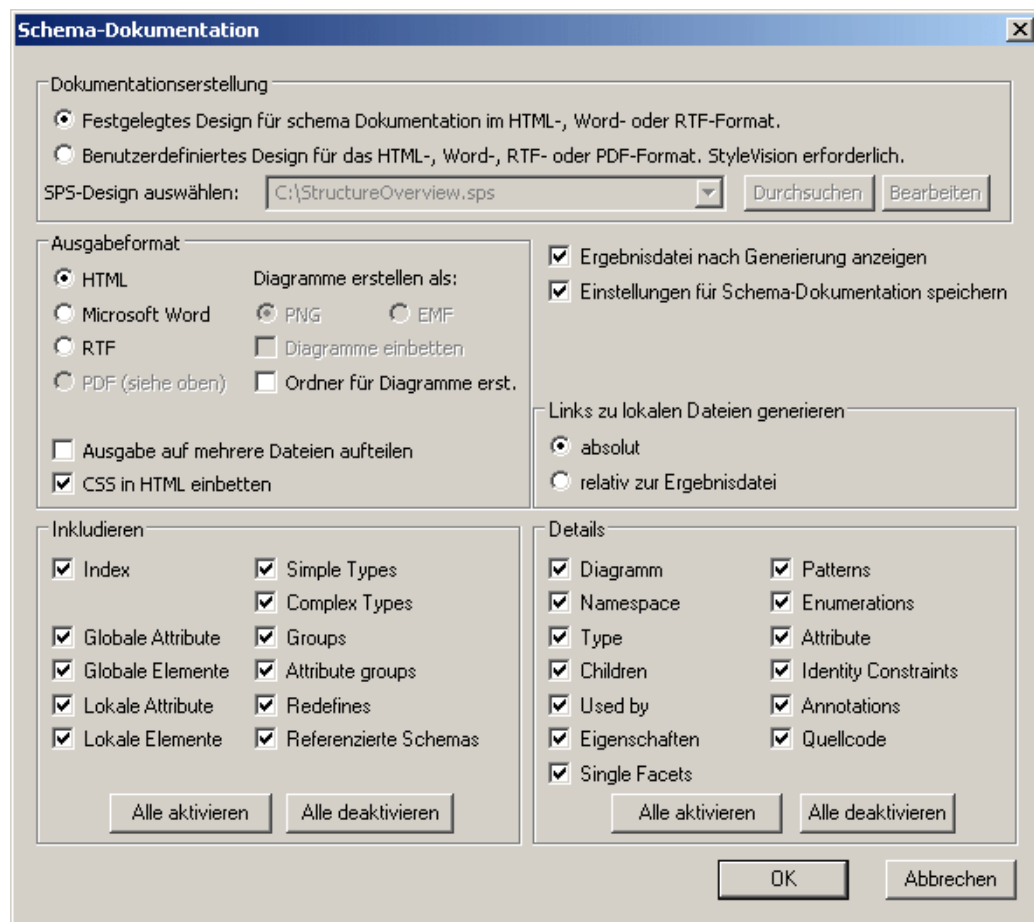
Bitte beachten Sie: Wie auch bei referenzierten globalen Elementen können Sie direkt zu einem Element innerhalb der ComplexType-Definition gehen, indem Sie die **Strg**-Taste gedrückt halten und auf das gewünschte Element im Content Model, das den ComplexType enthält, doppelklicken.

3.4.2 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 | Schema-Dokumentation generieren**, um das Dialogfeld "Schema-Dokumentation" aufzurufen.



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:\Program Files\Altova\XMLSpy2010\Examples\Tutorial\AddressLast.xsd**
 attribute form **unqualified**
 default:
 element form **qualified**
 default:
 targetNamespace: **http://my-company.com/namespace**

Elements [Complex types](#) [Simple types](#)

[Company](#) [AddressType](#) [US-State](#)

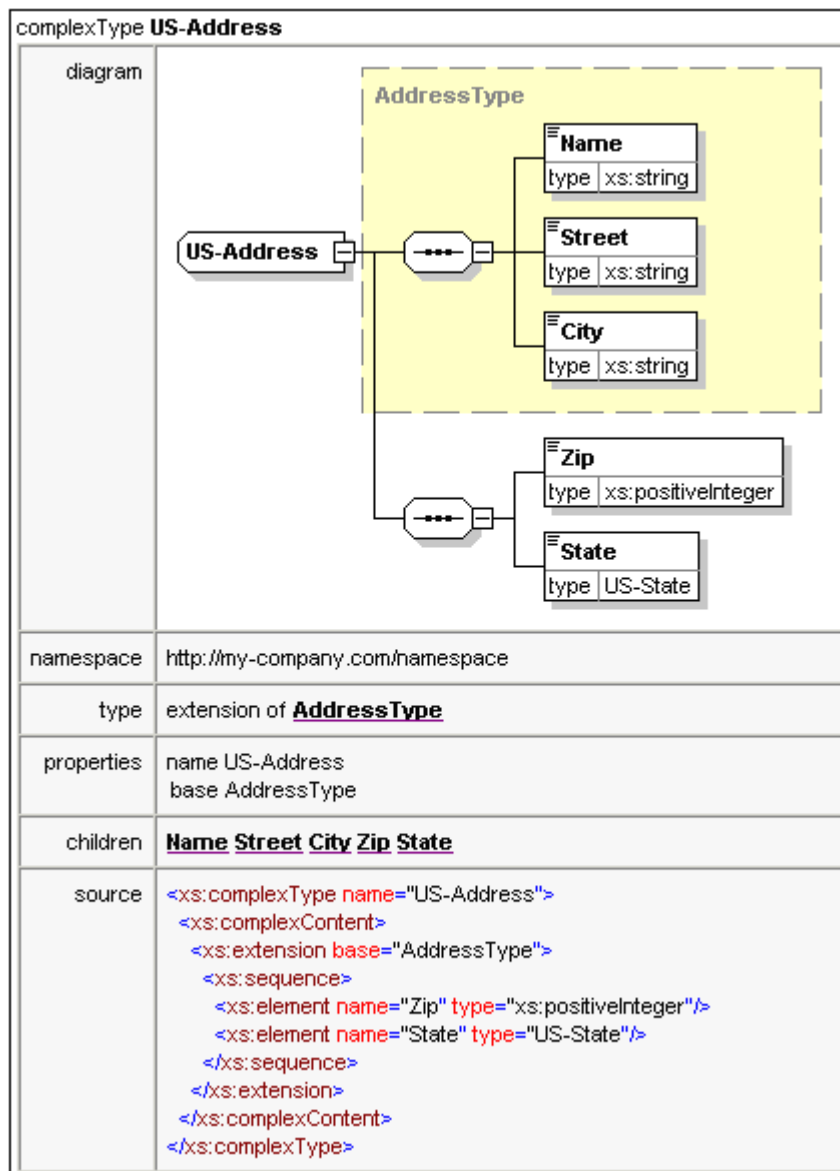
[Person](#) [UK-Address](#)

[US-Address](#)

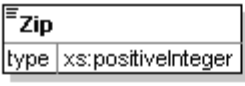
element **Company**

diagram	Root element
namespace	http://my-company.com/namespace
properties	content complex
children	Address Person
annotation	documentation Root element
source	<pre> <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> </pre>

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.



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

element US-Address/Zip	
diagram	
namespace	http://my-company.com/namespace
type	xs:positiveInteger
properties	name Zip isRef 0 content simple
source	<code><xs:element name="Zip" type="xs:positiveInteger"/></code>
element US-Address/State	
diagram	
namespace	http://my-company.com/namespace
type	<u>US-State</u>
properties	name State isRef 0 content simple
source	<code><xs:element name="State" type="US-State"/></code>
simpleType US-State	
namespace	http://my-company.com/namespace
type	xs:string
properties	name US-State
used by	element <u>US-Address/State</u>
source	<code><xs:simpleType name="US-State"> <xs:restriction base="xs:string"/> </xs:simpleType></code>

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.

3.5 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 Datenbank/Tabellen-Ansicht der Grid-Ansicht zur Verfügung.
- Sortieren von XML-Elementen mit Hilfe der Sortierfunktion der Datenbank/Tabellen-Ansicht.
- 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 Enhanced 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 Datenbank/Tabellen-Ansicht (oder einfach Tabellenansicht). Das Symbol dient zum Wechseln zwischen der Tabellen,- und der normalen Grid-Ansicht.



F7 Zum Kontrollieren, ob das Dokument wohlgeformt ist.



F8 Validiert das XML-Dokument gegen das damit verknüpfte Schema bzw. die DTD.



Öffnet die dazugehörige DTD oder XML-Schema-Datei.

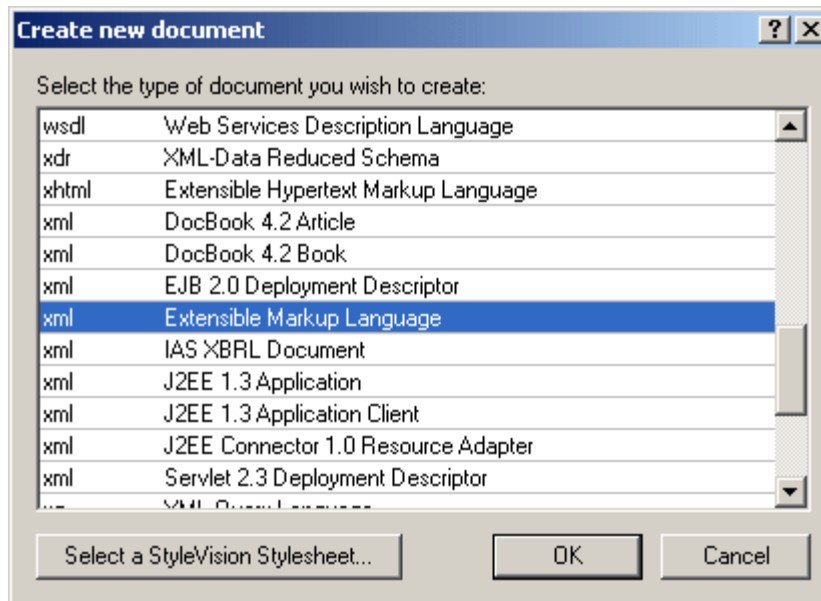
3.5.1 Erstellen eines neuen XML-Dokuments

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

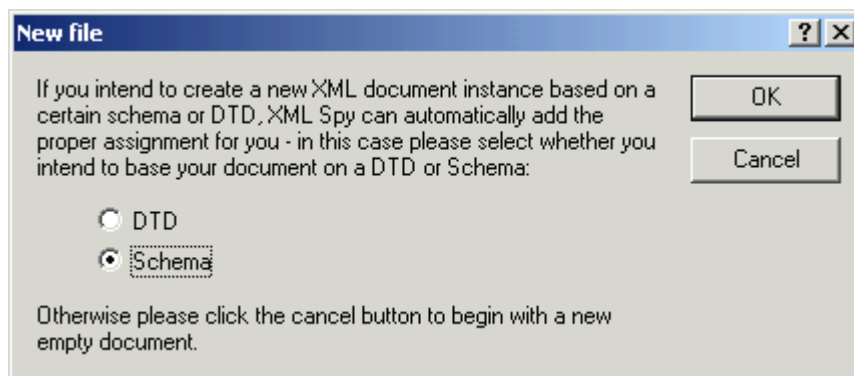
AddressLast.xsd basiert.

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. Wählen Sie `AddressLast.xsd` (Pfad siehe [Tutorial Einführung](#)) 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:

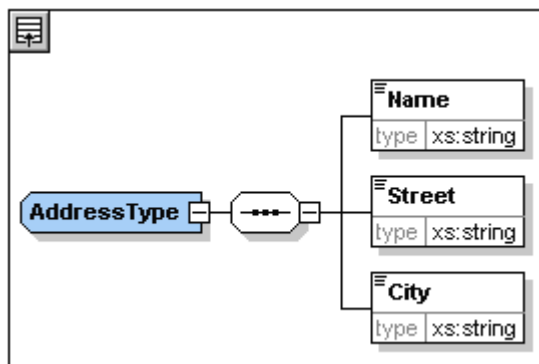
XML	
version	1.0
encoding	UTF-8
Company	
xm:ns	http://my-company.com/namespace
xm:ns:xsi	http://www.w3.org/2001/XMLSchema-
xsi:schemaLoca...	http://my-company.com/namespace AddressLast.xsd
Address	
Person Manager=	

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

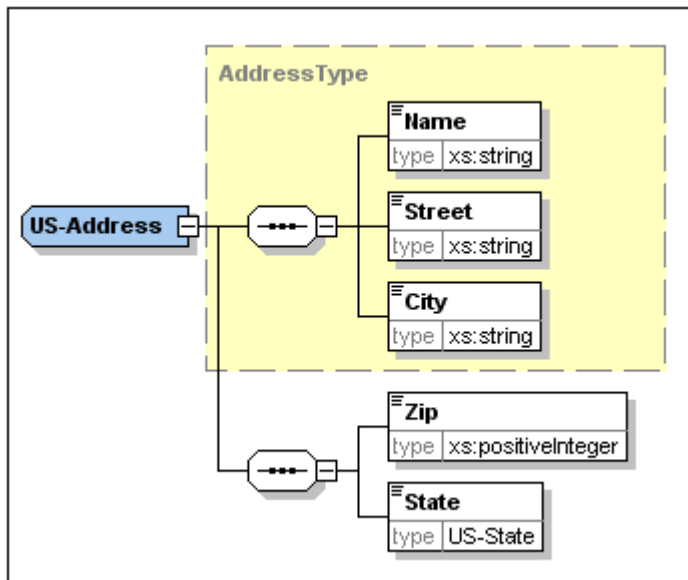
Company	
xm:ns	http://my-company.com/namespace
xm:ns:xsi	http://www.w3.org/2001/XMLSchema-inst
xsi:schema...	http://my-company.com/namespace AddressLast.xsd
Address	
Name	
Street	
City	
Person Manager=	

3.5.2 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 einen spezifischen US- oder UK-Adresstyp verwenden anstelle des generischen Adresstyps. 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.

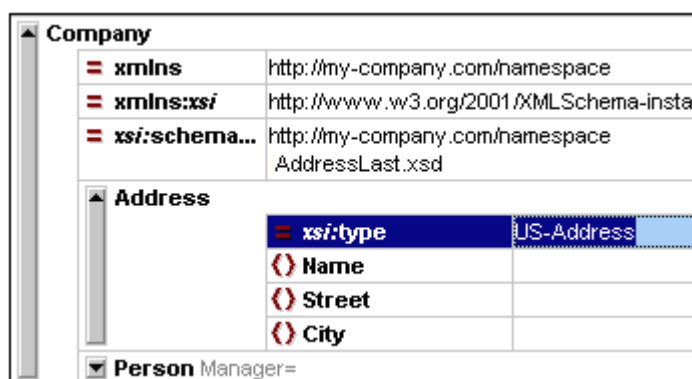
Gehen Sie folgendermaßen vor:

1. Klicken Sie im XML-Dokument mit der rechten Maustaste auf das Element `Name` und wählen Sie im Kontextmenü **Einfügen | Attribute**.



Dem Element `Address` wird ein Attributfeld hinzugefügt.

2. Stellen Sie sicher, dass als Name des Attributs `xsi:type` eingegeben wird (*Abbildung unten*).
3. Drücken Sie die Tabulatortaste um in das nächste (Werte)-Feld zu gelangen.



4. Geben Sie `US-Address` als Attributwert ein.

Bitte beachten Sie: Das vorgesetzte `xsi` ermöglicht die Verwendung von speziellen XML-Schema-spezifischen Befehlen in Ihrer XML-Dokumentinstanz. Beachten Sie, dass der

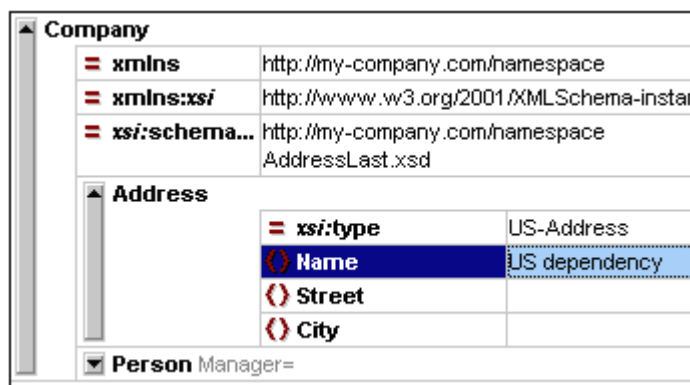
Namespace für das Präfix `xsi` automatisch zum Dokumentelement hinzugefügt wurde, als Sie Ihrer XML-Datei ein Schema zugewiesen haben. Sie haben soeben einen Typ für das Element `Address` festgelegt. Nähere Informationen dazu finden Sie in der [XML Schema-Spezifikation](#).

3.5.3 Eingabe von Daten in der Grid-Ansicht

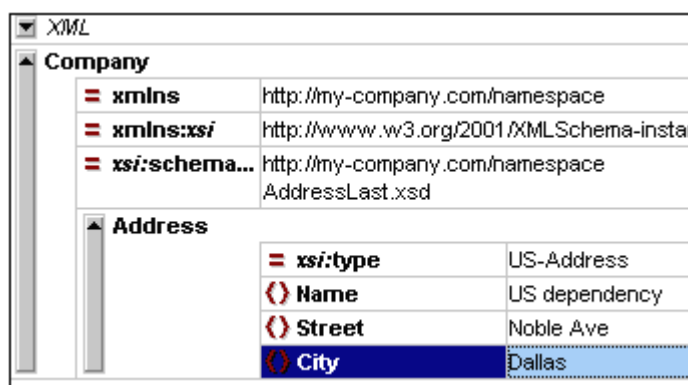
Sie können jetzt Daten in Ihr XML-Dokument eingeben.

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:



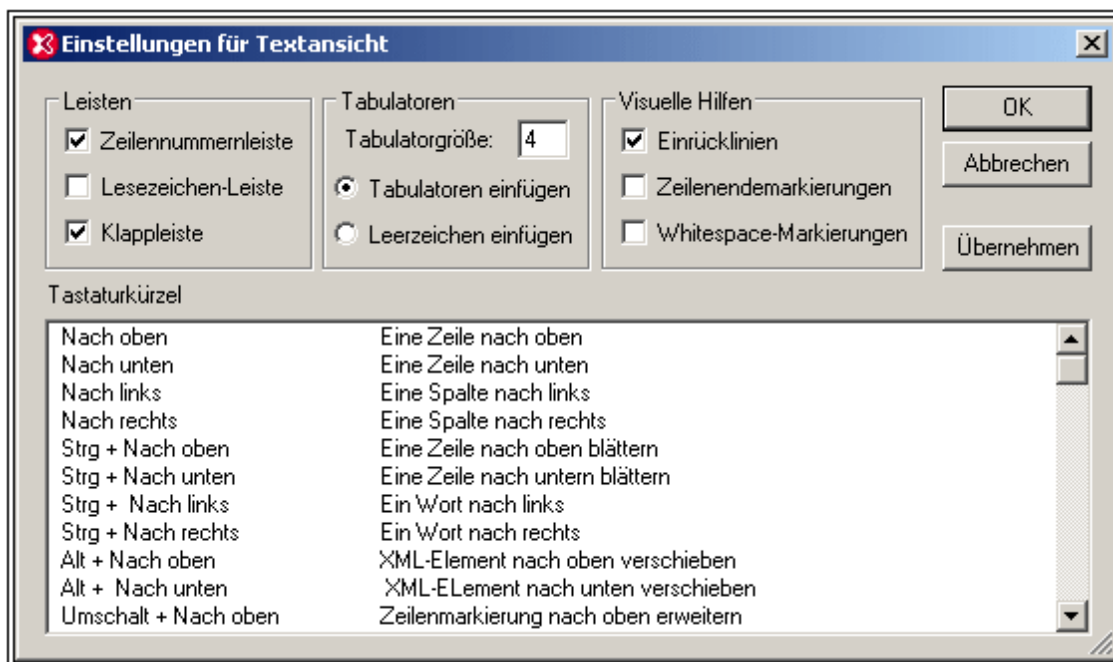
3.5.4 Eingabe von Daten in der Textansicht

Die Textansicht eignet sich aufgrund ihrer intelligenten Editierfunktionen für DTD/XML-Schema ideal zum Editieren der eigentlichen Daten und des Markup-Codes von XML-Dateien.

Funktionen zum strukturellen Layout

Zusätzlich dazu verfügt die Textansicht über eine Reihe Funktionen zum Anzeigen und

Bearbeiten der Struktur, die das Bearbeiten großer Textabschnitte erleichtern. Diese Funktionen können im Dialogfeld "Einstellungen für Textansicht" (**Ansicht | Einstellungen für Textansicht** *Abbildung unten*) aktiviert und deaktiviert werden.



In der Textansicht können die folgenden Leisten ein- und ausgeblendet werden:

- Zeilennummernleisten
- Lesezeichenleisten, in denen einzelne Zeilen durch einen Marker markiert werden können.
- Klappleisten, die Symbole enthalten, um Elemente ein- und auszublenden.

Zusätzlich dazu können visuelle Hilfen wie z.B. Einrücklinien, Zeilenendemarkierungen und Leerzeichenmarkierungen ein- und ausgeblendet werden, indem Sie die entsprechenden Kontrollkästchen im Bereich *Visuelle Hilfen* des Dialogfelds "Einstellungen für Textansicht" (*siehe Abbildung oben*) aktivieren bzw. deaktivieren.

Mit Hilfe der Lesezeichenleiste können Sie Markierungen in Ihrem Dokument setzen. Um ein Lesezeichen einzufügen, verwenden Sie den Befehl **Bearbeiten | Lesezeichen einfügen/löschen**. Sobald Lesezeichen eingefügt wurden, können Sie mit Hilfe der Befehle im Menü **Bearbeiten** zu diesen Lesezeichen navigieren.

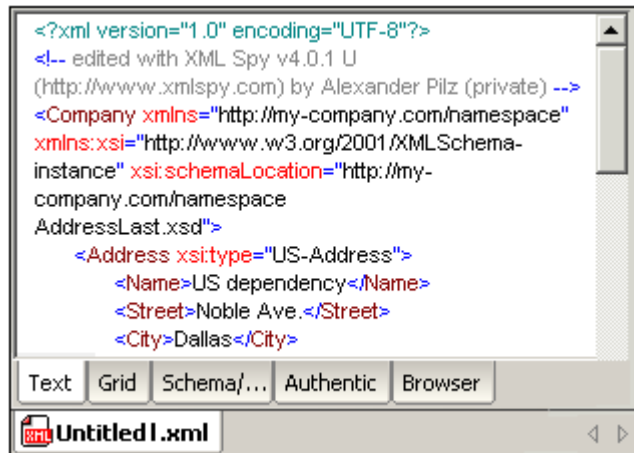
In der unten gezeigten Abbildung sehen Sie die Textansicht der XML-Datei, die sich in Bearbeitung befindet. Alle Sonderfunktionen der Textansicht sind aktiviert. Aus Gründen der Übersichtlichkeit werden diese Funktionen (Zeilennummerierung, Einrücklinien usw.) in weiterer Folge in dieser Anleitung in Textansichten nicht mehr angezeigt. Nähere Informationen zur Textansicht finden Sie im Benutzerhandbuch.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  = <Company xmlns="http://my-company.com/harr
3    C:\PROGRA~1\Altova\XMLSPY2004\Examples\
4  = <Address xsi:type="US-Address">
5      <Name>US dependency</Name>
6      <Street>Noble Ave</Street>
7      <City>Dallas</City>
8  </Address>
```

Bearbeitung in der Textansicht

In diesem Abschnitt werden wir Daten in der Textansicht eingeben und bearbeiten, um die Funktionen der Textansicht zu erläutern. Gehen Sie folgendermaßen vor:

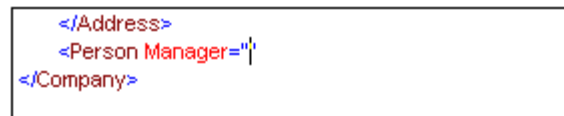
1. Wählen Sie die Menüoption **Ansicht | Text** oder klicken Sie auf das Register Text. Sie sehen nun das XML-Dokument in seiner Textform mit der Syntaxfärbung.



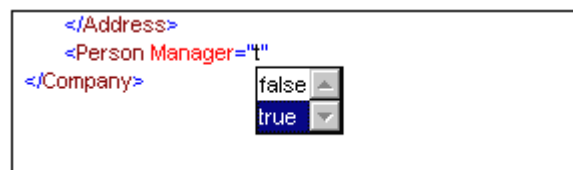
2. Setzen Sie den Cursor hinter den End-Tag des Elements "Address" und drücken Sie die Eingabetaste um eine neue Zeile einzufügen.
3. Geben Sie ein "Kleiner als..." Zeichen (<) an dieser Stelle ein. Eine Auswahlliste aller an dieser Stelle (laut Schema) zulässigen Elemente erscheint. Da an dieser Stelle 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.



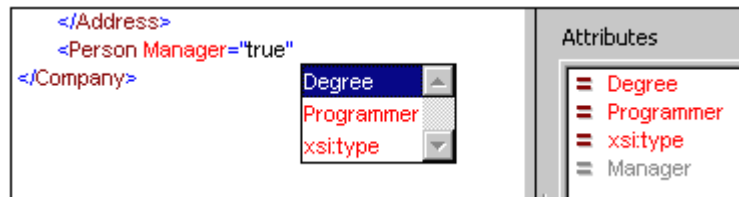
5. Wählen Sie in der Dropdown-Liste des Attributs `Manager` den Eintrag `true` aus.



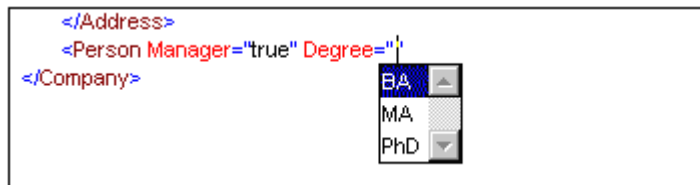
Nach Betätigen der Eingabetaste wird der Wert `true` an der Cursorposition eingefügt.

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 diesmal eine Liste von an dieser Stelle zulässigen

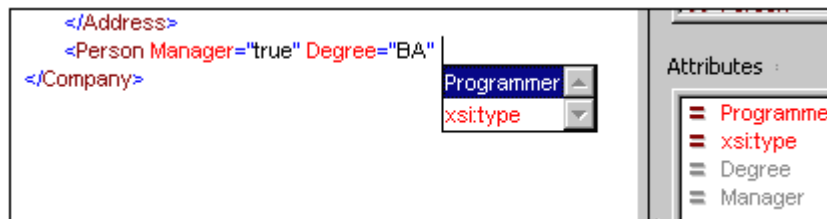
Attributen enthält. In der Attribut-Eingabehilfe sind verfügbare Attribute rot markiert. Das Attribut "Manager" ist ausgegraut, da es bereits verwendet wurde.



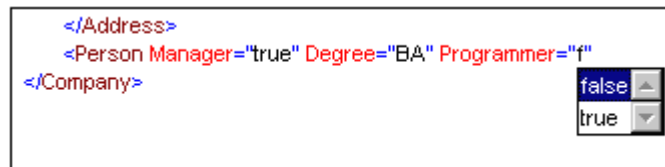
7. Wählen Sie `Degree` durch Betätigen des 'Nach unten'-Pfeiles und drücken Sie die Eingabetaste. Eine weitere Liste wird geöffnet. Hier können Sie eine der vordefinierten Enumerationen auswählen (BA, MA oder PhD).



8. Wählen Sie `BA` aus der Liste, drücken Sie die Eingabetaste und setzen Sie den Cursor durch Drücken der Ende-Taste an das Ende der Zeile. Drücken Sie die Leerzeilentaste. Nun sind `Manager` und `Degree` in der Attribut-Eingabehilfe ausgegraut.



9. Wählen Sie `Programmer` aus der Liste und drücken Sie die Eingabetaste.



10. Geben Sie den Buchstaben "f" ein und drücken Sie die Eingabetaste.
11. Setzen Sie den Cursor 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.

```

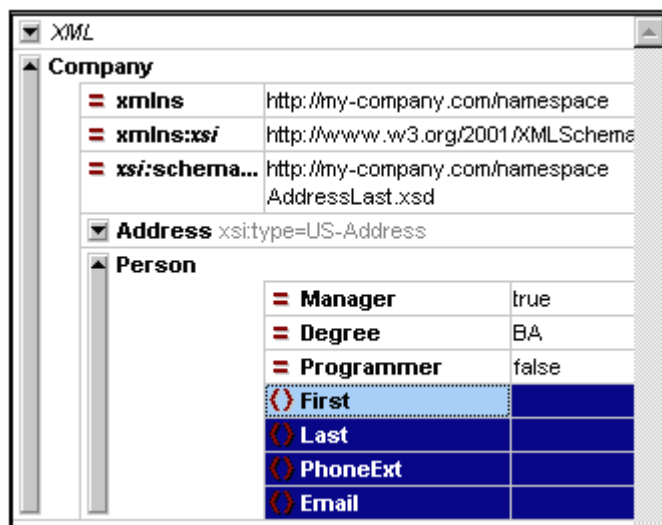
<?xml version="1.0" encoding="UTF-8"?>
<Company xmlns="http://my-company.com/namespace"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://my-company.com/namespace
AddressLast.xsd">
  <Address xsi:type="US-Address">
    <Name>US dependency</Name>
    <Street>Noble Ave</Street>
    <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**. Die neu hinzugefügten Child-Elemente von `Person` erscheinen markiert.



Wir wollen das Dokument nun validieren und etwaige Fehler korrigieren.

3.5.5 Validierung des Dokuments

XMLSpy bietet zwei Funktionen zur Überprüfung des XML-Dokuments:

- Wohlgeformtheitsprüfung
- Validierung

Wenn ein Fehler bei einer dieser beiden Überprüfungen gefunden wird, muss das Dokument entsprechend geändert werden.

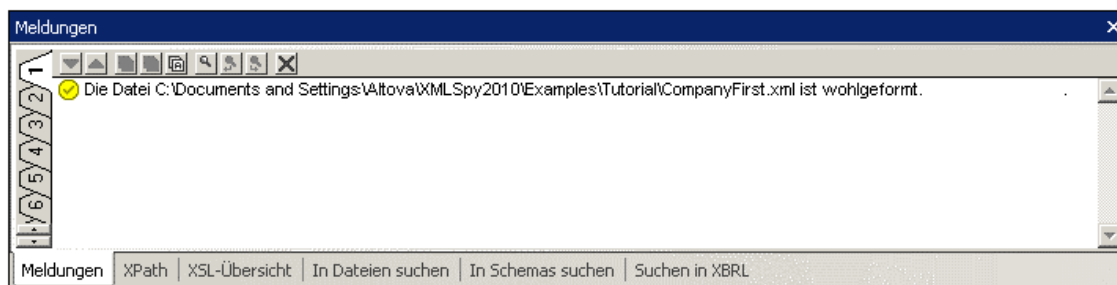
Überprüfung der Wohlgeformtheit

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 usw.).

Sie können die Wohlgeformtheit eines Dokuments in jeder Bearbeitungsansicht überprüfen. Wählen Sie zum Prüfen der Wohlgeformtheit die Menüoption **XML | Wohlgeformtheit prüfen**

oder drücken Sie die Taste **F7** oder klicken Sie auf . Am unteren Rand des Hauptfensters erhalten Sie im Fenster "Meldungen" eine Meldung, welche anzeigt, ob das Dokument wohlgeformt ist.

Beachten Sie, dass das Meldungsfenster neun Ausgaberegister enthält. Das Ergebnis der Validierung wird immer auf dem aktuellen Register angezeigt. Sie können daher die Wohlgeformtheit für eine Schemadatei auf Register 1 überprüfen und das Ergebnis aufbewahren, indem Sie zu Register 2 wechseln, bevor Sie das nächste Schema-Dokument validieren (andernfalls wird Register 1 durch das Validierungsergebnis überschrieben.)



Bitte beachten Sie: Bei der Wohlgeformtheitsprüfung nicht geprüft wird, ob die Struktur der XML-Datei schemakonform ist. Dies wird bei der Validierung der Datei überprüft.

Überprüfung der Gültigkeit (Validierung)

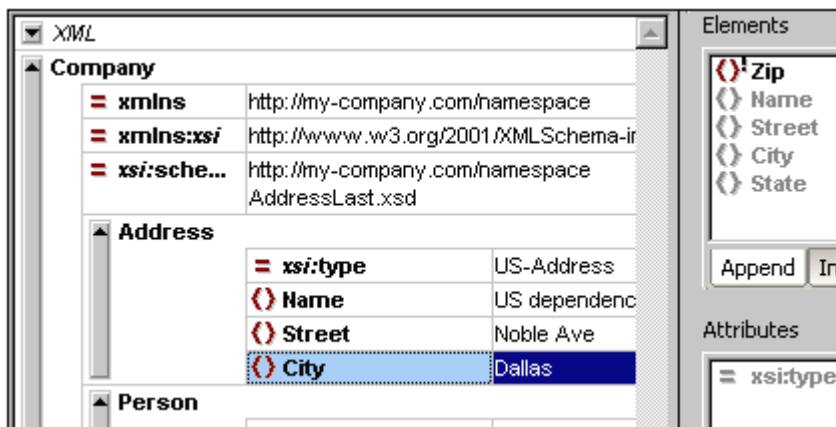
Ein XML-Dokument ist gemäß einem Schema gültig, wenn es von Struktur und Inhalt mit den Schema-Spezifikationen übereinstimmt.

Um die Gültigkeit Ihres XML-Dokuments zu überprüfen, wählen Sie zuerst die Grid-Ansicht aus und wählen Sie anschließend die Menüoption **XML | Validieren** oder drücken Sie die Taste **F8**

oder klicken Sie auf . 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, dass der ComplexType `US-Address` (dem aufgrund von `xsi:type` attribute 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

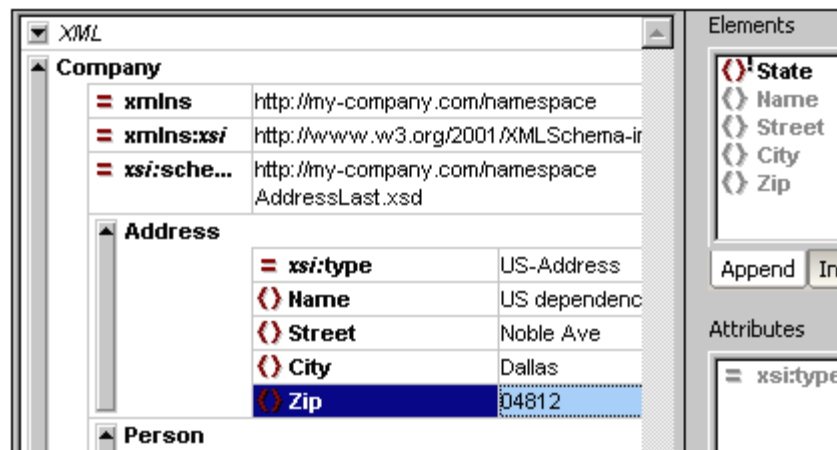
Das Element, welches den Fehler verursacht hat, in diesem Fall das Element `City`, wird markiert angezeigt.



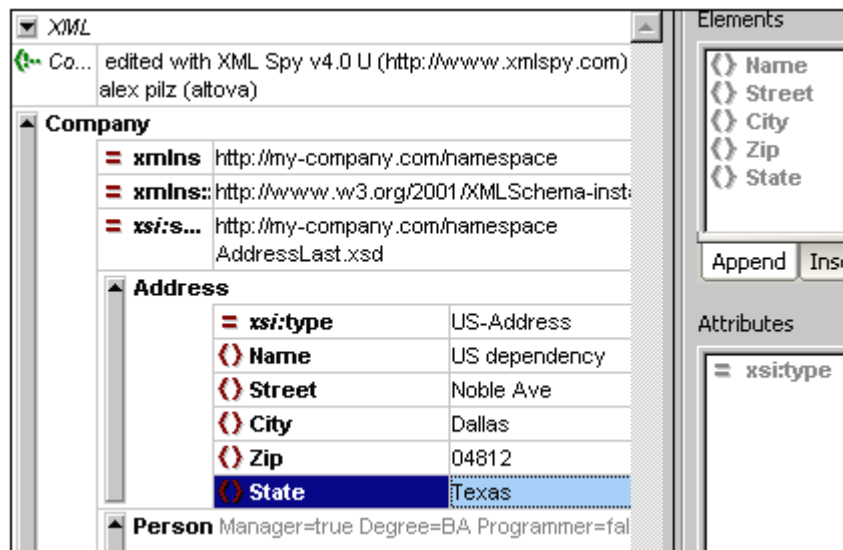
Bitte beachten Sie hierzu auch die Eingabehilfe (oben rechts), bei welcher ein Ausrufezeichen "!" vor dem Element `Zip` anzeigt, dass es sich hierbei um ein **obligatorisches** Element handelt.

Um den Validierungsfehler zu beheben, gehen Sie folgendermaßen vor:

1. Doppelklicken Sie auf das Element `Zip` in der Element-Eingabehilfe. Hierdurch wird das Element `Zip` unter dem Element `City` eingefügt. (Das Register "Anhängen" ist in der Element-Eingabehilfe aktiv.)
2. Drücken Sie die Tabulatortaste, geben Sie eine Postleitzahl (Zip) ein - (z.B. 04812) und bestätigen Sie mit der Eingabetaste. Die Element-Eingabehilfe zeigt nun an, dass das Element `State` obligatorisch ist (ein Ausrufezeichen ist dem Element vorangestellt). *Siehe Screenshot unten.*



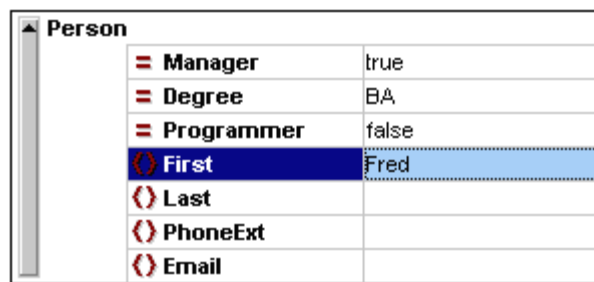
3. Doppelklicken Sie in der Element-Eingabehilfe auf das Element `State`, drücken Sie die Tabulatortaste und geben Sie den Namen des Bundesstaates ein - z.B. `Texas`. Bestätigen Sie mit der Eingabetaste. Die Element-Eingabehilfe enthält nun nur noch ausgegraute Elemente. Hierdurch wird angezeigt, dass alle obligatorischen Child-Elemente von `Address` eingefügt worden sind.



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:

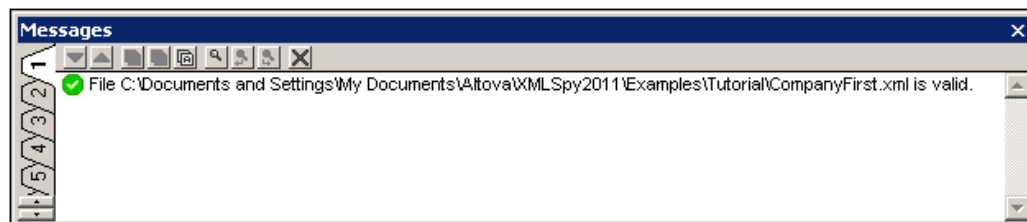
1. Klicken Sie auf das Wertefeld des Elements `First` und geben Sie den Vornamen der Person ein - z.B. `Fred`, bestätigen Sie mit der Eingabetaste.



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

Company	
xmlns	http://my-company.com/namespace
xmlns:xsi	http://www.w3.org/2001/XMLSchema-instance
xsi:schemaLoca...	http://my-company.com/namespace C:\PROGRA~1\Altova\XMLSpy2006\Examples\T utorial\AddressLast.xsd
Address	
xsi:type	US-Address
Name	US Dependency
Street	Noble Ave.
City	Dallas
Zip	04812
State	Texas
Person	
Manager	true
Degree	BA
Programmer	false
First	Fred
Last	Smith
PhoneExt	22
Email	Smith@work.com

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** und benennen Sie das Dokument - z.B. `CompanyFirst.xml`. Beachten Sie, dass es im Ordner Tutorial bereits eine fertige Datei mit dem Namen `CompanyFirst.xml` gibt, d.h. Sie sollten diese vorher umbenennen.

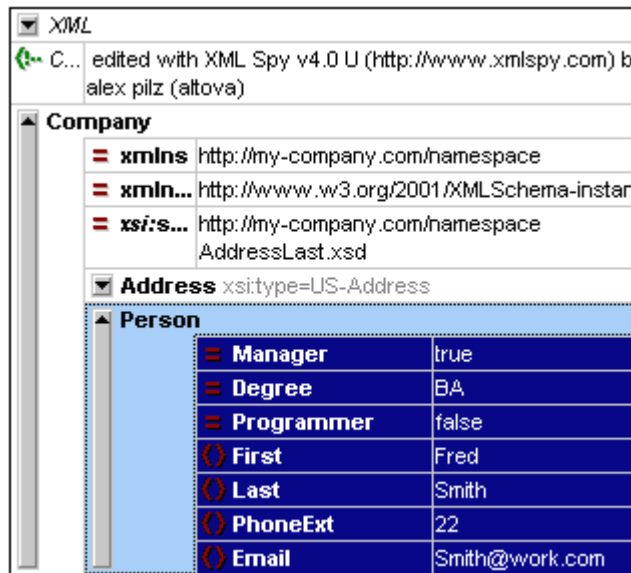
Bitte beachten Sie: Ein XML-Dokument kann auch abgespeichert werden wenn es nicht gültig ist. Beim Speichern werden Sie gefragt, ob Sie das nicht korrekte Dokument speichern wollen. Mit **Save anyway** wird das Dokument in seinem aktuellen, ungültigen Status gespeichert.

3.5.6 Hinzufügen von Elementen und Attributen

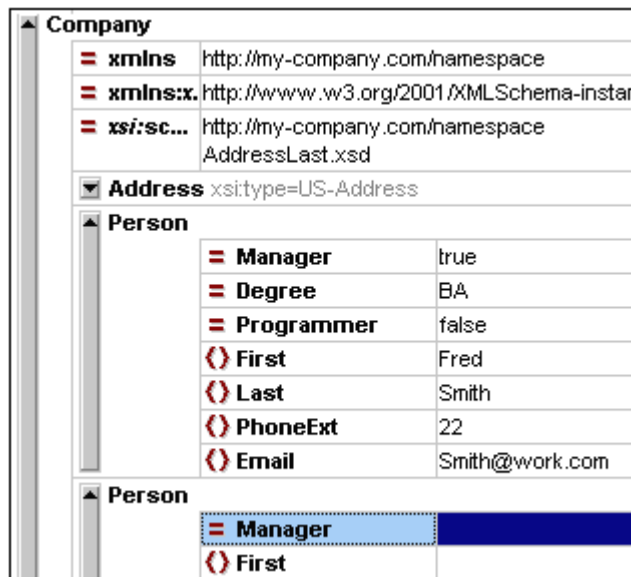
Momentan gibt es im Dokument nur ein `Person`-Element.

So fügen Sie ein neues Element `Person` hinzu:

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`, indem Sie in der Grid-Ansicht auf oder unter den Text des Elements `Person` klicken. Beachten Sie, dass das Element `Person` nun in der Element-Eingabehilfe auf dem Register "Anhängen" verfügbar ist.

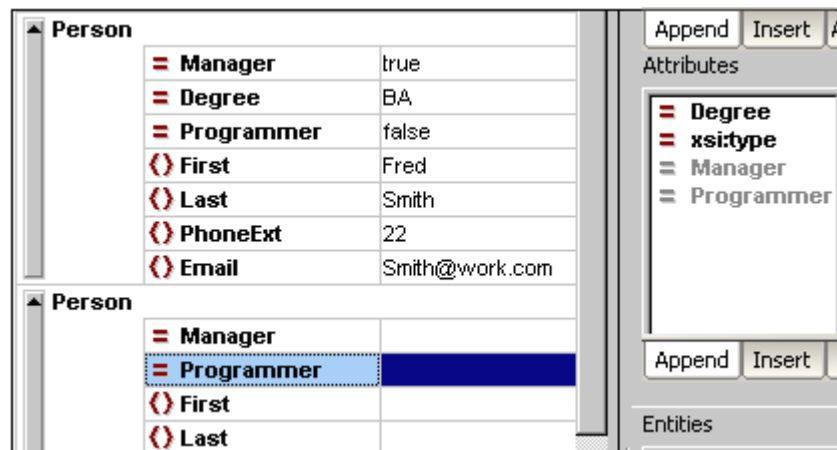


3. Doppelklicken Sie in der Element-Eingabehilfe auf das Element `Person`. Ein neues `Person`-Element sowie alle obligatorischen Child-Elemente werden angehängt (siehe Abbildung unten). Beachten Sie, dass das optionale Child-Element `Title` nicht eingefügt wird.



4. Wechseln Sie in die Grid-Ansicht und drücken Sie **F12** um das neue Element `Person` statt in der Tabellenansicht in der normalen Grid-Ansicht anzuzeigen.
5. Klicken Sie auf das Attribut `Manager` im neuen Element `Person`. Beachten Sie die vorhandenen Attribute in der Attribut-Eingabehilfe. Der Eintrag `Manager` ist ausgegraut, da er bereits verwendet wurde. Werfen Sie auch einen Blick auf das Info-Fenster, in dem nun Informationen über das Attribut `Manager` angezeigt werden. Es handelt sich hierbei um ein obligatorisches Attribut, weswegen es hinzugefügt wurde. Das Attribut `Programmer` wurde nicht hinzugefügt.
6. Doppelklicken Sie in der Attribut-Eingabehilfe auf dem Register "Anhängen" auf den Eintrag `Programmer`. Hierdurch wird ein leeres `Programmer`-Attribut hinter dem

Manager-Attribut eingefügt.



Das Attribut `Programmer` ist nun in der Attribut-Eingabehilfe ausgegraut.

Sie könnten in dieser Ansicht einen Inhalt für das Element `Person` eingeben. Wechseln wir jedoch nun in die Datenbank/Tabellen-Ansicht der Grid-Ansicht, da sich diese besser zum Editieren einer Struktur mit mehreren Instanzen wie z.B. bei `Person` eignet.

3.5.7 Editieren in der Datenbank/Tabellen-Ansicht

Die Grid-Ansicht enthält eine spezielle Ansicht namens Datenbank/Tabellen-Ansicht (in weiterer Folge Tabellenansicht genannt), die sich zum Bearbeiten von Elementen mit mehreren Instanzen eignet. Einzelne Elementtypen können als Tabelle angezeigt werden. Wenn ein Elementtyp als Tabelle angezeigt wird, werden dessen Children (Attribute und Elemente) als Spalten dargestellt, wobei die einzelnen Instanzen die Zeilen der Tabelle bilden.

Um einen Elementtyp als Tabelle anzuzeigen, wählen Sie eine beliebige Instanz dieses Elementtyps aus und klicken Sie in der Symbolleiste auf das Symbol "Als Tabelle anzeigen" (XML | Tabelle | Als Tabelle anzeigen). Elementtypen einer untergeordneten Ebene, die mehrere Instanzen haben, werden ebenfalls in Tabellenform angezeigt. Die Tabellenansicht steht in der Enhanced Grid-Ansicht zur Verfügung und dient zum Bearbeiten jeder Art von XML-Datei (XML, XSD, XSL, usw.).

Vorteile der Tabellenansicht

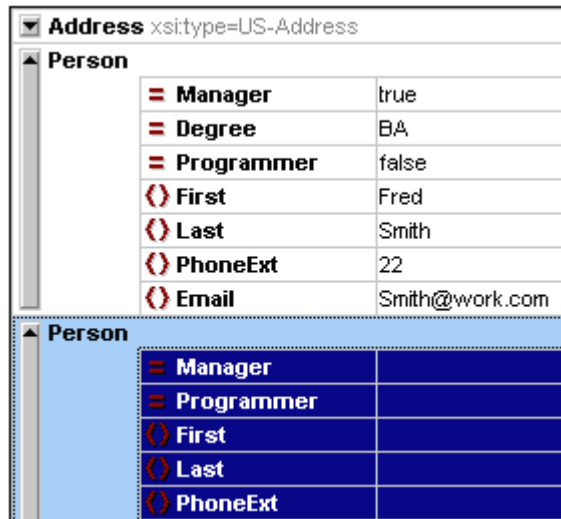
Die Tabellenansicht bietet die folgenden Vorteile:

- Sie können Spalten mit Drag & Drop untereinander austauschen. Im XML-Dokument selbst wird dadurch die relative Position von Child-Elementen oder Attributen für alle Element-Instanzen, die den Zeilen der Tabelle entsprechen, geändert.
- Sortieren von Daten nach Spalten mittels der Funktion **XML | Tabelle | Aufsteigend Sortieren** oder **Absteigend sortieren**.
- Einfügen oder Anhängen von Zeilen mittels der Funktion **XML | Tabelle | Zeile Einfügen**
- Kopieren und Einfügen von **strukturierten Daten** von und in Produkte von Drittanbietern
- Auch die intelligenten Eingabehilfen stehen in der Tabellenansicht zur Verfügung.

Anzeigen eines Elementtyps als Tabelle

So zeigen Sie das Element `Person` in Tabellenform an:

1. Wählen Sie in der Grid-Ansicht eines der `Person`-Elemente aus, indem Sie auf oder in die Nähe des `Person`-Texts klicken.



Person	
Manager	true
Degree	BA
Programmer	false
First	Fred
Last	Smith
PhoneExt	22
Email	Smith@work.com

2. Wählen Sie die Menüoption **XML | Tabelle | Als Tabelle anzeigen** oder klicken Sie auf das Symbol . Beide `Person`-Elemente werden in Form einer einzigen Tabellen angezeigt. Die Element- und Attributnamen werden in der Kopfzeile der Tabelle, die einzelnen Elementinstanzen in den Tabellenzeilen angezeigt.



Company	
xmlns	http://my-company.com/namespace
xmlns...	http://www.w3.org/2001/XMLSchema-instance
xsi:s...	http://my-company.com/namespace AddressLast.xsd
Address xsitype=US-Address	
Person (2)	
	Manager Degree Programmer
1	true BA false
2	

3. Wählen Sie die Menüoption **Ansicht | Optimale Breite** oder klicken Sie auf das Symbol , um die Spaltenbreite zu optimieren.

Bitte beachten Sie: Sie können für einzelne Elementtypen die Tabellenansicht ein- oder ausblenden. Wählen Sie dazu die Tabelle aus (klicken Sie auf den Elementnamen in der Tabelle) und klicken Sie anschließend auf das Symbol "Als Tabelle anzeigen" . Beachten Sie jedoch, dass Child-Elemente, die als Tabellen angezeigt wurden, weiterhin als Tabellen angezeigt werden.

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. Beachten Sie jedoch, dass `PhoneExt` eine Ganzzahl bis 99 sein muss, damit die Datei gültig ist. Die intelligenten Editierfunktionen sind auch innerhalb der Tabellenzellen aktiv, sodass Sie Optionen aus Dropdown-Listen auswählen können, falls Optionen verfügbar sind (Boolescher Inhalt und die Enumerationen für das Attribut `Degree`).

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.com

Bitte beachten Sie: Die Eingabehilfen sind auch für die als Tabelle dargestellten Elemente und Attribute aktiv. Wenn Sie z.B. in der Element-Eingabehilfe auf den Eintrag `Person` doppelklicken, wird eine neue Tabellenzeile hinzugefügt (D.h. es wird eine neue Instanz zum Element `Person` hinzugefügt).

Kopieren von XML Daten aus und in Drittprodukte

In XMLSpy können Kalkulationstabellendaten aus und in Produkte anderer Anbieter schnell und einfach kopiert werden. Die kopierten Daten können in XMLSpy sowie in Produkten anderer Anbieter in nativem Format der Anwendung verwendet werden. In diesem Abschnitt erfahren Sie, wie man Daten in und aus einem Excel-Datenblatt kopiert. Gehen Sie dazu folgendermaßen vor:

1. Klicken Sie auf die Zeilennummerierung 1, halten Sie die **Strg**-Taste gedrückt und klicken Sie auf die Zeilennummerierung 2. Hierdurch werden beide Zeilen markiert.

Address xs:complexType=US-Address							
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.com

2. Wählen Sie die Menüoption **Bearbeiten | Als strukturierten Text kopieren**. Der Befehl kopiert die Elemente in die Zwischenablage, und zwar genauso, wie sie am Bildschirm erscheinen.
3. Öffnen Sie Excel und fügen Sie die XML Daten (mit **Strg+V**) in ein Excel-Datenblatt ein.

A	B	C	D	E	F	G	H
TRUE	BA	FALSE	Fred	Smith	22	Smith@work.com	
FALSE	MA	TRUE	Alfred	Aldrich	33	Aldrich@work.com	

4. Fügen Sie eine neue Zeile im Excel-Arbeitsblatt ein und geben Sie eine **dreistellige Zahl** im Element `PhoneExt` ein (z.B. 444).

A	B	C	D	E	F	G	H
TRUE	BA	FALSE	Fred	Smith	22	Smith@work.com	
FALSE	MA	TRUE	Alfred	Aldrich	33	Aldrich@work.com	
TRUE	PhD	FALSE	Colin	Coletti	444	Coletti@work.com	

5. Markieren Sie die Tabellendaten in Excel und drücken Sie **Strg+C**, um die Daten in die Zwischenablage zu kopieren
6. Wechseln Sie wieder zu XMLSpy.
7. Klicken Sie in XMLSpy in die oberste linke Zelle der Tabelle und wählen Sie **Bearbeiten | Einfügen**.

Address xs:type=US-Address						
Person (3)						
	= Manager	= Degree	= Programmer	First	Last	Pho
1	TRUE	BA	FALSE	Fred	Smith	22
2	FALSE	MA	TRUE	Alfred	Aldrich	33
3	TRUE	PhD	FALSE	Colin	Coletti	444

- Die aktualisierten Daten wurden in die Tabelle übernommen.
- Ändern Sie mittels der Menüoption **Suchen | Ersetzen** (Funktionstaste **Strg+H**) die Booleschen Werte TRUE/FALSE in true/false (in Kleinbuchstaben).

Sortieren der Tabellen nach Spalteninhalt

In der Tabellenansicht können Sie Ihre Daten je nach Spalte in aufsteigender oder absteigender Reihenfolge sortieren. Wir möchten nun die Tabelle `Person` alphabetisch nach Nachnamen sortieren.

So sortieren Sie eine Tabelle nach dem Inhalt einer Spalte:

- Klicken Sie auf die Kopfzeile in der Spalte **Last**.

Address xs:type=US-Address						
Person (3)						
	= Manager	= Degree	= Programmer	First	Last	Phone
1	true	BA	false	Fred	Smith	22
2	false	MA	true	Alfred	Aldrich	33
3	true	PhD	false	Colin	Coletti	444

- Wählen Sie die Menüoption **XML | Tabelle | Aufsteigend sortieren** oder klicken Sie auf das Symbol . Die Spalte wird unter Einbeziehung der gesamten Tabelle alphabetisch sortiert. Die Spalte bleibt hierbei markiert.

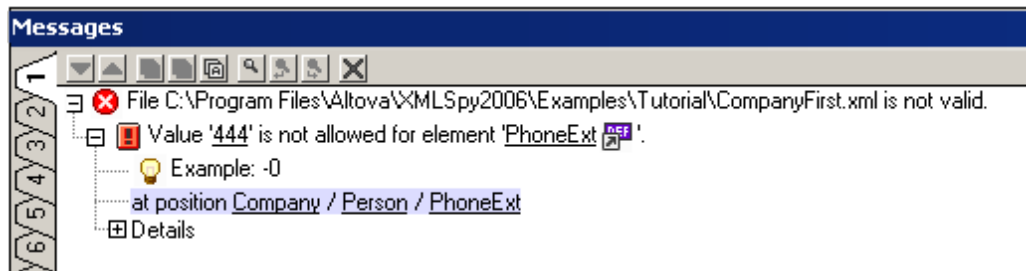
Address xs:type=US-Address						
Person (3)						
	= Manager	= Degree	= Programmer	First	Last	Phone
1	false	MA	true	Alfred	Aldrich	33
2	true	PhD	false	Colin	Coletti	444
3	true	BA	false	Fred	Smith	22

Die Tabelle wird nicht nur in der Ansicht sortiert, sondern die Daten werden auch im zugrundeliegenden XML-Dokument sortiert, d.h. die `Person`-Elemente werden nun alphabetisch nach dem Inhalt der Spalte `Last` sortiert. (Klicken Sie auf das Register "Text", um die Änderungen in der Textansicht zu sehen).

- Wählen Sie die Menüoption **XML | Validieren** oder drücken Sie die Funktionstaste **F8**. Es wird eine Fehlermeldung angezeigt, dass der Wert '444' bei einem `PhoneExt`-Elements nicht zulässig ist (siehe Abbildung). Das fehlerverursachende `PhoneExt`-Element erscheint markiert.

Erweitern Sie "Details", um zu sehen, dass `PhoneExt` nicht gültig ist, weil es nicht kleiner oder gleich dem Maximalwert 99 ist.

Bitte beachten Sie: Sie können auf die Links in der Fehlermeldung klicken, um zu der Stelle in der XML-Datei zu gelangen, an der der Fehler gefunden wurde.



Da der für Durchwahlnummern festgelegte Wertebereich diese Durchwahl nicht mehr enthält, müssen wir das XML-Schema so ändern, dass diese Nummer gültig ist. Dies wird im nächsten Abschnitt beschrieben.

3.5.8 Ändern des Schemas

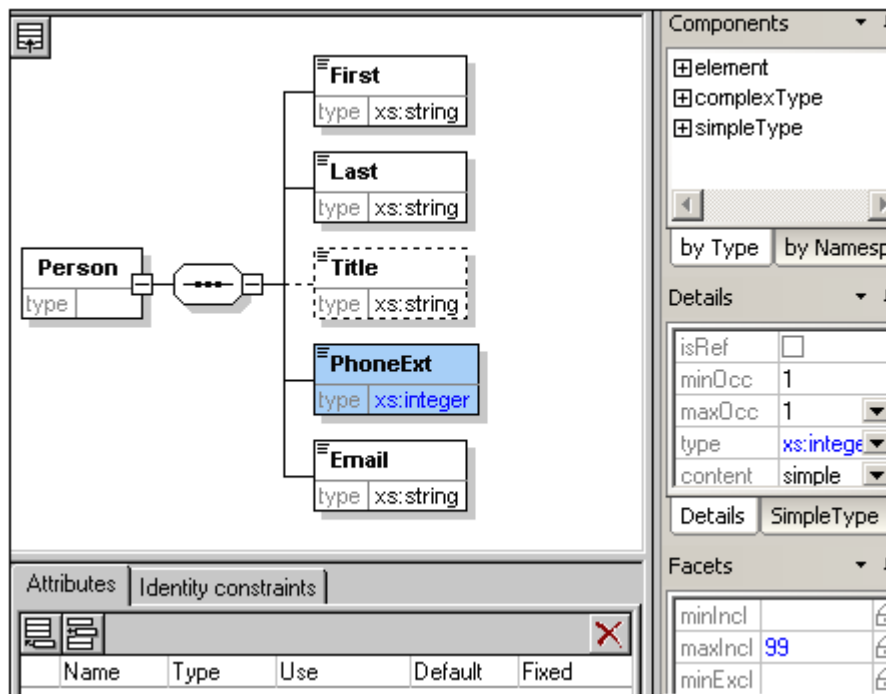
Wir haben nun gesehen, dass zweistellige Telefondurchwahlnummern nicht ausreichend sind. Wir werden daher den Wertebereich gültiger Nummern auf drei Stellen erweitern. Sie können das zugrunde liegende XML-Schema öffnen und ändern, ohne Ihr XML-Dokument schließen zu müssen.

Gehen Sie folgendermaßen vor:

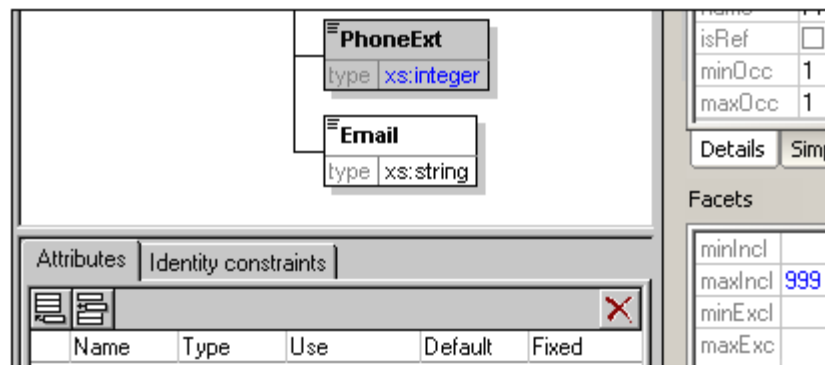
1. Wählen Sie die Menüoption **DTD/Schema | Gehe zu Definition** oder klicken Sie auf das Symbol . Das assoziierte Schemadokument (in diesem Fall AddressLast.xsd) wird geöffnet. Wechseln Sie in die Schema-Ansicht (siehe Abbildung unten).

element	Company	ann:Root element
complexType	AddressType	ann:
complexType	US-Address	ann:
simpleType	US-State	ann:
complexType	UK-Address	ann:
element	Person	ann:

2. Klicken Sie auf das Symbol "Diagramm anzeigen"  des globalen Elementes `Person` und klicken anschließend auf das Element `PhoneExt`. Die Facet-Daten werden auf dem Register "Facets" angezeigt.

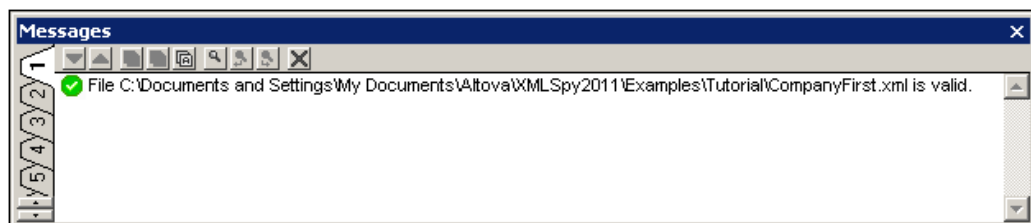


3. Doppelklicken Sie auf dem Register "Facets" auf das Wertefeld `maxIncl`, geben Sie 999 als Maximalwert ein und drücken Sie die Eingabetaste.



4. Speichern Sie das Schema-Dokument.
5. Mit **Strg+TAB** kommen Sie zurück zum XML-Dokument.

6. Klicken Sie auf die Schaltfläche , um das XML-Dokument erneut zu validieren.



Die Meldung "Diese Datei ist gültig" wird angezeigt. Das XML-Dokument ist nun mit der geänderten Schemadefinition konform.

7. Wählen Sie die Menüoption **Datei | Speichern unter...** und benennen Sie die Datei `CompanyLast.xml`. (Denken Sie daran, das Originaldokument `CompanyLast.xml`

umzubenennen z.B. in `CompanyLast_orig.xml`).

Bitte beachten Sie: Das Originaldokument `CompanyLast.xml` ist im Verzeichnis `Tutorial` abgelegt.

3.6 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.



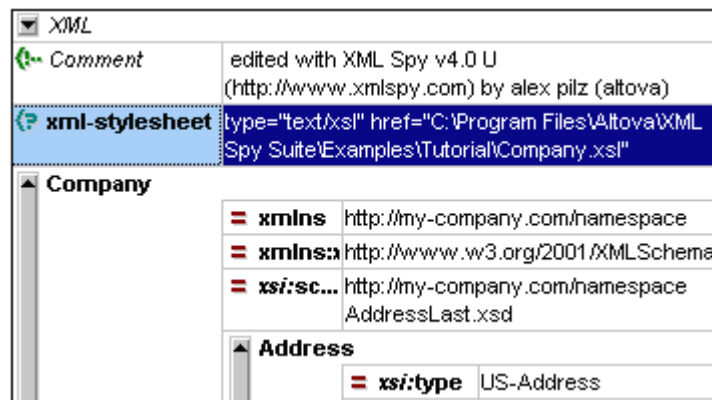
XSL/XQuery | XSL-Transformation (F10) oder die Symbolleistenschaltfläche transformiert das aktive XML-Dokument mit Hilfe des diesem zugewiesenen XSL Stylesheet. Wenn keine XSL-Datei zugewiesen wurde, werden Sie beim Aufrufen dieses Befehls aufgefordert, eine zuzuweisen.

Bitte beachten Sie: XMLSpy verfügt über zwei integrierte XSLT-Prozessoren, den Altova XSLT 1.0-Prozessor und den Altova XSLT 2.0-Prozessor. Der Altova XSLT 1.0-Prozessor dient zum Transformieren von XSLT 1.0 Stylesheets, während der Altova XSLT 2.0-Prozessor für die Verarbeitung von XSLT 2.0 Stylesheets verwendet wird. 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 XSLT 1.0 Stylesheets. Bei Aufruf des Befehls **XSL-Transformation** wird bei diesen Stylesheets automatisch der Altova XSLT 1.0-Prozessor verwendet.

3.6.1 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.xml machen" aktiviert haben, ist der Pfad relativ; andernfalls ist er absolut (wie in der obigen Abbildung).

3.6.2 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 auf das Symbol , um die Transformation mittels des in dem XML-Dokument referenzierten XSL Stylesheet zu starten. (Da die Datei `Company.xsl` ein XSLT 1.0-Dokument ist, wird für die Transformation automatisch der integrierte Altova XSLT 1.0-Prozessor ausgewählt). Das Ausgabedokument wird in der Browser-Ansicht angezeigt. Es hat den Namen `XSL_Output.html`. (Wenn die HTML-Ausgabedatei nicht generiert wird, stellen Sie sicher, dass die Standarddateierweiterung im Dialogfeld "Optionen" (**Extras | Optionen**) auf dem Register "XSL" auf `.html` gesetzt wurde.) Die Firmendaten werden in einem Block links oben angezeigt, während die Personendaten unterhalb davon in Tabellenform dargestellt werden.

Your Company

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

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

Text Browser

 CompanyLast.xml  AddressLast.xsd  XSL Output.html

Bitte beachten Sie: Falls in der Ausgabedatei nur eine Tabellenkopfzeile aber keine Tabellendaten zu sehen sind, stellen Sie bitte sicher, dass Sie den Target Namespace für Ihr Schema wie am Beginn des Tutorials unter [Definieren von Namespaces](#) beschrieben, definiert haben. Der Namespace muss in allen drei Dateien (Schema, XML und XSL) **identisch** sein.

3.6.3 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 "Lime (Hellgrün)" in "Yellow (Gelb)" geändert werden.

Gehen Sie folgendermaßen vor:

1. Klicken Sie auf das Register `CompanyLast.xml` um es aktiv zu setzen und stellen Sie sicher, dass Sie sich in der Grid-Ansicht befinden.
2. Wählen Sie die Menüoption **XSL/XQuery | Gehe zu XSL**.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
xmlns:my="http://my-company.com/namespace">

<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="lime">
        <thead align="center">
          <td><strong>First</strong></td>
          <td><strong>Last</strong></td>
          <td><strong>Ext.</strong></td>
```

Mit diesem Befehl wird die Datei Company. XSL geöffnet, die in der XML-Datei referenziert wird.

3. Gehen Sie zur Zeile `<table border="1" bgcolor="lime">` und ändern Sie den Eintrag von `bgcolor="lime"` in `bgcolor="yellow"`.

```
<h1><center>Your Company</center></h1>
<xsl:apply-templates select="//my:Address"/>
<table border="1" bgcolor="yellow">
  <thead align="center">
    <td><strong>First</strong></td>
    <td><strong>Last</strong></td>
```

4. Wählen Sie die Menüoption **Datei | Speichern**, um die Änderungen an der XSL-Datei zu speichern.
5. Klicken Sie auf das Register CompanyLast. xml um die XML-Datei aktiv zu setzen, und wählen Sie **XSL/XQuery | XSL-Transformation** oder drücken Sie die Funktionstaste **F10**. In der Browser-Ansicht von XMLSpy erscheint eine neue Datei namens XSL Output. html. Die Hintergrundfarbe der Tabelle ist gelb.

Your Company

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

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

6. Wählen Sie die Menüoption **Datei | Speichern** und benennen Sie die Datei Company. html.

3.7 Projektverwaltung

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

3.7.1 Vorteile von Projekten

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

- Dateien und URLs können durch gemeinsame Erweiterungen oder andere Kriterien in Ordner gruppiert werden.
- Batchprozesse können bestimmten Ordnern oder dem ganzen Projekt zugewiesen 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 eine sofortige Transformation der darin befindlichen Dateien zu erlauben.
- Es können Zielordner für die XSL-Transformationsdateien definiert werden.

Alle Einstellungen können unter **Projekt | Projekteigenschaften...** definiert werden. 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...**). Weitere Informationen erhalten Sie im Abschnitt [Versionskontrolle](#) der Online-Hilfe.
- [Persönliche, netzinterne](#) und [Webordner](#) können in Projekte eingefügt werden, wodurch die Batchvalidierung ermöglicht wird.

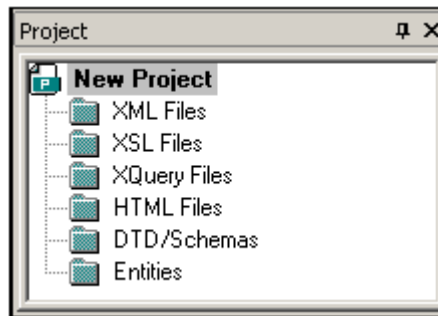
3.7.2 Erstellen eines Projekts

Im Hauptfenster ist nun eine Reihe von Tutorial-Dateien offen. 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.

Erstellen eines einfachen Projekts

Um ein einfaches Projekt zu erstellen, gehen Sie folgendermaßen vor:

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



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`. Beachten Sie, dass durch Verarbeitungsanweisungen 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`.

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 (mit der 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 hinzufügen**. Die XML-Datei wird anhand ihres Dateityps (`.xml`) in den Ordner XML Files hinzugefügt.
3. Fügen Sie nun auf dieselbe Art eine HTML-Datei und eine XSD-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 | Speichern** oder durch Auswahl einer beliebigen Datei oder eines Ordners im Projektfenster und Klicken auf das Symbol "Speichern" in der Symbolleiste (oder **Datei | Speichern**).

Bitte beachten Sie: 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 in Projekt einfügen** 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.

3.8 Fertig!

Herzlichen Glückwunsch!

Wir hoffen, dass Ihnen diese Anleitung geholfen hat, um die Grundzüge von XMLSpy kennen zu lernen. Weitere Informationen finden Sie in der kontextsensitiven Online-Hilfe. Oder drucken Sie die PDF-Version des Tutorials aus, die sich im Anwendungsordner von XMLSpy befindet (tutorial.pdf).

4 Bearbeitungsansichten

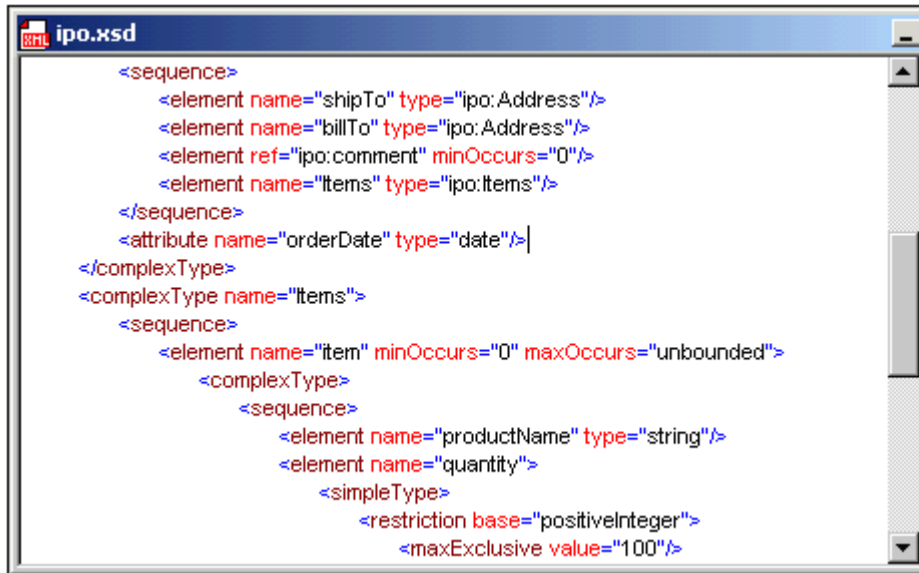
XMLSpy enthält leistungsstarke Bearbeitungsansichten. Zusätzlich zur Textansicht mit ihren intelligenten Bearbeitungsfunktionen gibt es grafische Ansichten, mit denen die Bearbeitung von Dokumenten einfacher wird. Abhängig davon, welche Dokumentart gerade in XMLSpy aktiv ist, stehen im Hauptfenster eine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

In diesem Abschnitt beschreiben wir die verschiedenen Bearbeitungsansichten in XMLSpy:

- [Textansicht](#)
- [Grid-Ansicht](#)
- [Schema-Ansicht](#)
- WSDL-Ansicht
- XBRL-Ansicht
- [Authentic-Ansicht](#)
- [Browser-Ansicht](#)

4.1 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](#) beschrieben.

Die allgemeinen Funktionen der Textansicht wurden folgendermaßen gegliedert:

- [Formatieren in der Textansicht](#): Hier wird beschrieben, wie die Schriftarteeigenschaften, die Einrückung und der Zeilenumbruch im Dokument definiert werden.
- [Dokumentanzeige](#): 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](#): Hier werden die Funktionen beschrieben, die bei der Bearbeitung zur Verfügung stehen, insbesondere die intelligenten Bearbeitungsfunktionen.
- [Eingabehilfen](#) 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.

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**.

4.1.1 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 Elementen verwenden. Wenn Sie unterschiedliche Farben für unterschiedliche Textelemente verwenden, ist die Syntaxfärbungsfunktion aktiviert. Textschriftarten werden im [Dialogfeld "Optionen" auf dem Register "Schriftarten"](#) angepasst. Eine Beschreibung dazu finden Sie in dieser Dokumentation im Abschnitt [Benutzerreferenz | Optionen | Textschriftarten](#).

Einrückung

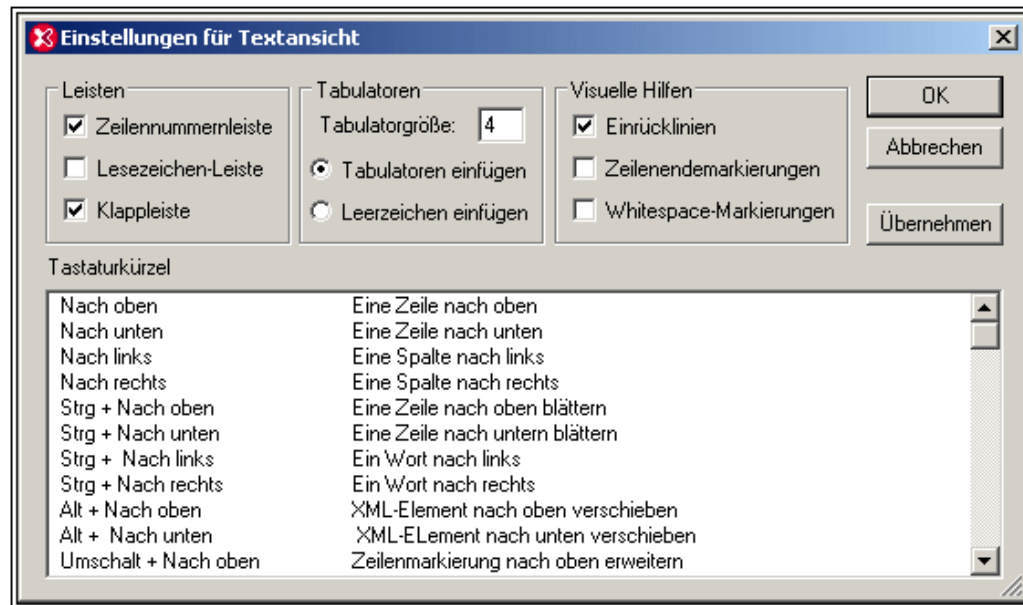
Wohlgeformte Dokumente können mit Pretty-Print dargestellt werden. D.h. das Dokument wird so formatiert, dass die hierarchische Struktur des Dokuments mit Hilfe neuer Zeilen und Einrückungen dargestellt wird (*siehe Screenshot unten*).



Um das Dokument auf diese Art anzuzeigen, gehen Sie folgendermaßen vor:

1. Aktivieren Sie im Dialogfeld "Optionen" auf dem Register "Ansicht" für Pretty-Print die Option "Einrückung verwenden". Dadurch werden bei Pretty-Print Einrückungen im Dokument vorgenommen, um die hierarchische Struktur darzustellen. Jede untergeordnete Ebene wird mit einer weiteren Einrückung gekennzeichnet. Wenn die Option "Einrückung verwenden" nicht aktiviert ist, beginnt jede Zeile im Dokument mit der Einrückung Null.
2. Wählen Sie im Dialogfeld "Einstellungen für Textansicht" (*Abbildung unten*) entweder die Option "Tabulatoren einfügen" oder "Leerzeichen einfügen". Damit legen Sie fest, ob für die Einrückung bei Pretty-Print Tabulatoren oder Leerzeichen verwendet werden. Wenn Leerzeichen definiert werden, wird jede weitere untergeordnete Ebene mit einer

zusätzlichen Anzahl an Leerzeichen eingerückt. Diese Anzahl wird im Dialogfeld "Einstellungen für Textansicht" unter Tabulatorgröße definiert.



3. Wählen Sie den Befehl [Bearbeiten | Pretty-Print](#) aus oder klicken Sie in der Symbolleiste auf die Schaltfläche **Pretty Print**. Daraufhin wird der Dokumenttext, je nachdem, was im Dialogfeld "Optionen" auf dem Register "Ansicht" ausgewählt wurde (i) mit oder ohne Einrückungen angezeigt; (ii) wenn im Dialogfeld "Optionen" auf dem Register "Ansicht" "Einrückungen" ausgewählt sind, dann werden die Einrückungen durch die Einstellungen im Dialogfeld "Einstellungen für Textansicht" im Bereich "Tabulatoren" definiert. Unnötige vorangestellte oder angehängte Leerzeichen werden entfernt, wenn Sie "Pretty-Print" auswählen.

Anmerkung: Pretty-Print wird auch im Hintergrund angewendet, wenn Sie das Dokument speichern oder die Ansicht wechseln. Wenn das Dokument nicht wohlgeformt ist, erhalten Sie eine entsprechende Fehlermeldung. Korrigieren Sie den Fehler und führen Sie dann Pretty-Print aus. Der Abstand der Einrückung wird durch die Einrücklinien gekennzeichnet. Dabei 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.

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](#). Die entsprechende Schaltfläche dazu befindet sich in der [Symbolleiste "Text"](#).

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 erweitern und reduzieren (*siehe Abbildung unten*). Die Klappleiste kann über das Dialogfeld "Einstellungen für Textansicht" (*siehe Abbildung oben*) ein- und ausgeblendet werden. Beachten Sie, wie die Zeilennummerierung in Zeile 14 und 24 zusammen mit den eingeklappten Nodes ebenfalls eingeklappt wurde.



Mit dem Befehl **Alle ein-/ausklappen** in der Symbolleiste "Text" werden **alle** Nodes in ihrer erweiterten Form angezeigt bzw. auf das Dokumentelement der obersten Ebene reduziert. Wenn ein Node erweitert wurde, stehen die folgenden Optionen zur Verfügung, wenn Sie auf das Ein-Ausklapp-Symbol (+/- Symbol) des Node klicken:

- **Einfacher Klick:** Klappt den Node ein. Wenn Sie erneut darauf klicken, wird er wieder erweitert angezeigt, sodass seine untergeordneten Nodes wieder, 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 bis zur Ebene der untergeordneten Nodes. Der angeklickte Node selbst bleibt geöffnet und zeigt die untergeordneten Nodes in reduzierter Form an. Dies funktioniert für alle Nodes in einer Verzweigung der Hierarchie mit Ausnahme des letzten reduzierbaren Node in dieser Verzweigung.

- **Strg+Klick:** Erweitert alle untergeordneten reduzierten Nodes und die diesen untergeordnete Nodes, d.h. alle untergeordneten Nodes des betreffenden Node werden erweitert.

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.

```

5 ...<CompanyLogo href="nanonull.gif" /> EOL
6 ...<Name>Organization.Chart</Name> EOL
7 ...<Office> EOL
8 .....<Name>Nanonull, Inc.</Name> EOL
9 .....<Desc> EOL

```

Vergrößern/Verkleinern

Durch Scrollen mit der Maus 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](#).

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.

4.1.3 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](#)
- [Start-Tag and End-Tag-Suche](#)
- [Intelligentes Editieren](#)
- [Autokomplettierung](#)
- [Verschieben von gleichrangigen Elementen relativ zu einander](#)
- [Auswahl eines ganzen Elements](#)
- [Drag-and-Drop und Kontextmenüs](#)
- [Suchen und Ersetzen](#)
- [Unbegrenztes Rückgängigmachen](#)
- [Rechtschreibprüfung](#)

Für einige Dokumenttypen (wie z.B. [XML](#) und [XQuery](#)) 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](#).

Syntaxfärbung

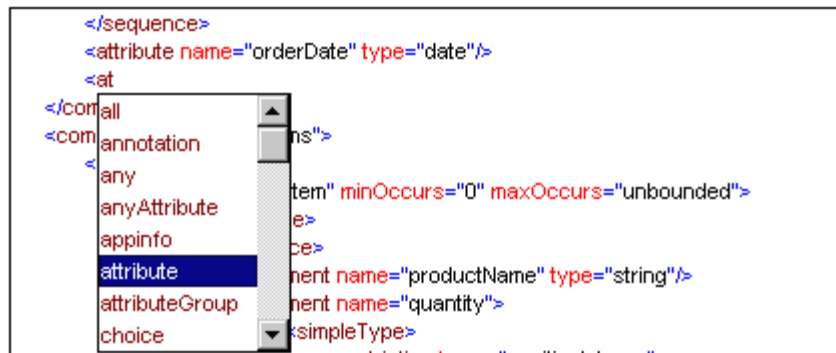
Die Syntaxfärbung richtet sich nach der Art des XML Nodes, d.h. je nachdem, ob es sich beim XML Node um ein Element, Attribut, Inhalt, einen CDATA-Abschnitt, Comment oder eine Processing Instruction handelt. Man unterscheidet zwischen vier Gruppen von Dokumenttypen: (i) allgemeinen XML-Dokumenten (einschließlich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

Start-Tag und End-Tag

Wenn Sie den Cursor innerhalb des Start- oder End-Tags eines XML-Elements setzen, wird der andere Tag des Paares markiert, wenn Sie **Strg+E** drücken. 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 leichter zu finden.

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.



Das Pop-up-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 Pop-up-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 Pop-up-Fenster werden die aufgezählten Werte angezeigt.
- Wenn Sie das Zeichen </ (den Beginn eines End-Tags) eingeben, wird im Pop-up-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/>`.

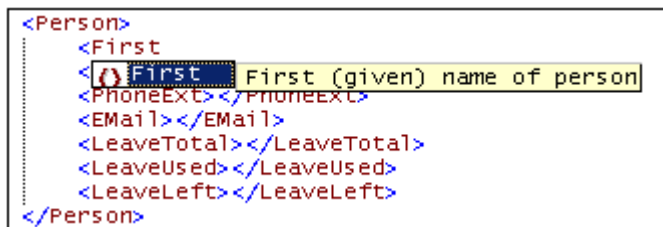
Autokomplettierung

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, 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 Werte. 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" auf dem Register "Bearbeiten"](#) (**Extras | Optionen | Bearbeiten**) aktiviert und deaktiviert werden.

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

Wenn sich der Cursor innerhalb eines Elements befindet, können Sie das ganze Element durch Drücken von **Strg+Umschalt+E** auswählen.

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](#), [Kopieren](#), [Einfügen](#), [Löschen](#), [Als Mail senden](#) und [Gehe zu Zeile/Zeichen](#)) mit einem Rechtsklick direkt über ein Kontextmenü aufrufen.

Suchen und Ersetzen

Mit Hilfe der Befehle [Suchen](#) und [Ersetzen](#) können Sie Text schnell finden und ändern. Diese Befehle unterstützen auch Regular Expressions, sodass Ihnen damit leistungsstarke Suchfunktionen zur Verfügung stehen. (Nähere Informationen siehe [Bearbeiten | Suchen](#).)

Unbeschränktes Rückgängigmachen

XMLSpy bietet Ihnen für alle Editiervorgänge unbeschränktes [Rückgängigmachen](#) und [Wiederherstellen](#).

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](#) und [Rechtschreibung - Optionen](#).

4.1.4 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](#) 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
- **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](#).

4.2 Grid-Ansicht

In der Grid-Ansicht (*Abbildung unten*) wird die hierarchische Struktur von XML-Dokumenten und DTDs mittels verschachtelter Datenbehälter angezeigt. Diese können für eine bessere Übersicht über die gesamte Dokumentstruktur erweitert oder reduziert werden. In der Grid-Ansicht können Sie Inhalt und Struktur eines Dokuments einfach bearbeiten.

XML

Company

=	xmlns	http://my-company.com/namespace
=	xmlns:xsi	http://www.w3.org/2001/XMLSchema-instance
=	xsi:schema...	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	()	Last
1	false		MA		true			Alfred		Aldrich
2	true		Ph.D		false			Colin		Coleman
3	true		BA		false			Fred		Smith

Ein hierarchisch dargestelltes Objekt (z.B. XML-Deklaration, Dokumenttyp-Deklaration oder ein Element, das Child-Elemente enthält), wird durch eine graue senkrechte Leiste auf der linken Seite dargestellt, die einen kleinen nach oben weisenden Pfeil enthält. Durch Klicken auf die Leiste können Sie das Datenelement [erweitern](#) oder [reduzieren](#). Ein Element wird durch das Symbol gekennzeichnet, ein Attribut durch das Symbol .

Anzeige und Navigation

- Der Inhalt eines hierarchischen Elements hängt von seiner Art ab. Bei Elementen handelt es sich dabei z.B. normalerweise um Attribute, Character Data, Comments und Child-Elemente. Attribute werden immer aufgelistet und erscheinen in der Reihenfolge der Input-Datei. Den Attributen folgen alle weiteren Objekte genau in der Reihenfolge, in der diese in der Quelldatei angegeben wurden. Sie können beliebig mittels Drag & Drop neu arrangiert werden. Diese Änderung wird auch in den Quelldaten übernommen.
- Falls ein Element nur Character Data enthält, werden diese Daten in der Zeile angezeigt, in der sich auch das Element befindet. Dieses Element wird dann nicht als hierarchisch angesehen. Character Data für andere Elemente werden eingerückt (zusammen mit Attributen und eventuellen Child-Elementen) und als "Text" gekennzeichnet.
- Bei reduzierten Elementen werden ihre Attribute und deren Werte in derselben Zeile in Grau angezeigt. Diese Attribut-Vorschau ist besonders hilfreich beim Editieren von XML-Dokumenten mit einer großen Anzahl von Elementen desselben Namens, die sich nur durch Inhalt oder Attribute unterscheiden (z.B. Datenbank-Applikationen).
- Mit Hilfe der Pfeiltasten können Sie die Laufleiste in der Grid-Ansicht bewegen.
- Mit Hilfe der + und – Tasten der Zahlentastatur können Sie Datenelemente erweitern und reduzieren.

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](#) 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.

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

Intelligentes Editieren

Die Grid-Ansicht gestattet intelligentes Editieren, wenn das XML-Dokument auf einem Schema basiert. Nähere Informationen dazu finden Sie unter [Editieren in der Grid-Ansicht](#). Die in der Grid-Ansicht verfügbaren Eingabehilfen sind im Folgenden beschrieben.

Tabellen der Grid-Ansicht

In der Grid-Ansicht können sich wiederholende Elemente in Form einer [Tabelle](#) angezeigt werden. Diese Funktion steht für jede Art von XML-Datei zur Verfügung: XML-, XSD-, XSLT-Datei usw..

4.2.1 Bearbeiten in der Grid-Ansicht

Beim Editieren eines schemabasierten (DTD oder XML-Schema) XML-Dokuments in der Enhanced Grid-Ansicht stehen Ihnen auf Basis von aus dem Schema stammenden Informationen intelligente Editierfeatures zur Verfügung. Diese Features sind im Folgenden aufgelistet.

Einfügen oder Anhängen von Element- oder Attributnamen

Zum Einfügen oder Anhängen eines Attributs oder Elements relativ zum ausgewählten Datenelement, können Sie entweder die Befehle im XML-Menü oder die Symbole in der

Attribut- und Element-Symboleiste



verwenden. Falls diese Symbolleiste nicht sichtbar ist, wählen Sie die Menüoption **Extras | Anpassen** und aktivieren Sie auf dem Register **Symbolleisten** den Eintrag **Attr & Elem**. Eine ausführliche Beschreibung zur Verwendung der Befehle der Symbolleiste und des XML-Menüs finden Sie im Abschnitt [Menü "XML"](#).

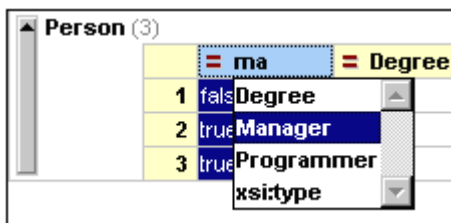
Um ein Element oder Attribut einzufügen oder anzuhängen, gehen Sie folgendermaßen vor:

1. Wählen Sie das Datenelement, relativ zu welchem das Element oder Attribut eingefügt oder angehängt werden soll.
2. Klicken Sie auf das entsprechende Symbol in der Element- und Attributsymboleiste oder wählen Sie den Befehl im [XML-Menü](#) (**Einfügen**, **Anhängen** oder **Child hinzufügen**) aus. Daraufhin wird im Raster ein neuer Eintrag erstellt und es erscheint ein Popup-Fenster mit verfügbaren Element- oder Attributoptionen.
3. Wählen Sie im Popup-Fenster das gewünschte Element oder Attribut aus und bestätigen Sie die Auswahl durch Klicken oder Drücken der **Eingabetaste**. Alternativ dazu können Sie den Namen des gewünschten Elements oder Attributs auch eingeben.

Sie werden feststellen, dass die verschiedenen [Eingabehilfen](#) je nachdem, was in der Grid-Ansicht gerade ausgewählt ist, ständig aktualisiert werden. Im Info-Fenster werden jeweils wichtige Informationen zum ausgewählten Element/Attribut angezeigt.

Bearbeiten eines Element- oder Attributnamens

Wenn Sie den Namen eines Elements oder Attributs bearbeiten, wird ein Popup-Menü mit den verfügbaren Optionen geöffnet. Diese Optionen sind abhängig von der Position des Elements und dem im Schema definierten Content Model. Ein ähnliches Popup-Menü wird angezeigt, wenn der Inhalt eines Elements oder Attributs durch eine Enumeration oder Choice eingeschränkt wird.



So bearbeiten Sie einen Element- oder Attributnamen:

1. Doppelklicken Sie auf den Element- oder Attributnamen. Es erscheint ein Popup-Fenster mit den verfügbaren Optionen.
2. Wählen Sie das gewünschte Element oder Attribut aus dem Popup-Menü aus.
3. Bestätigen Sie die Auswahl durch Drücken der **Eingabetaste** oder Klicken auf den Namen. (mit **Esc** verwerfen Sie die Änderungen.)

Hinzufügen von Namespace-Präfixen

Mit dem Befehl [XML | Namespace-Präfix](#) können Sie in der Grid-Ansicht ein Namespace-Präfix zum ausgewählten Node und all seinen untergeordneten Elementen oder Attributen hinzufügen. Der Namespace muss separat deklariert werden.

Kontextmenü der Grid-Ansicht

Zusätzlich zu den Befehlen, die über die verschiedenen Menüs und Symbolleisten zur Verfügung stehen, bietet die Grid-Ansicht auch ein Kontextmenü, in dem die am häufigsten verwendeten Befehle aus den verschiedensten Menüs zusammengefasst sind. Klicken Sie mit der rechten Maustaste auf das Datenelement, für das Sie eine Aktion durchführen wollen, um das Kontextmenü zu öffnen.

Rechtschreibprüfung

Dokumente können in der Grid-Ansicht 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](#) und [Rechtschreibung - Optionen](#).

4.2.2 Tabellen der Grid-Ansicht

Die **Tabellen-Ansicht** ist in die Enhanced Grid-Ansicht integriert. Sich wiederholende Elemente werden in dieser Ansicht in einem Tabellenformat angezeigt. Die Tabellenansicht unterscheidet sich von der normalen Grid-Ansicht insofern, als darin für jeden Child-Typ des in Tabellenform dargestellten Elements eine Spalte angelegt wird. Sie können anschließend für ganze Spalten oder Auswahlbereiche Eigenschaften ändern.

Sehen Sie sich als Beispiel das folgende XML-Dokument an:

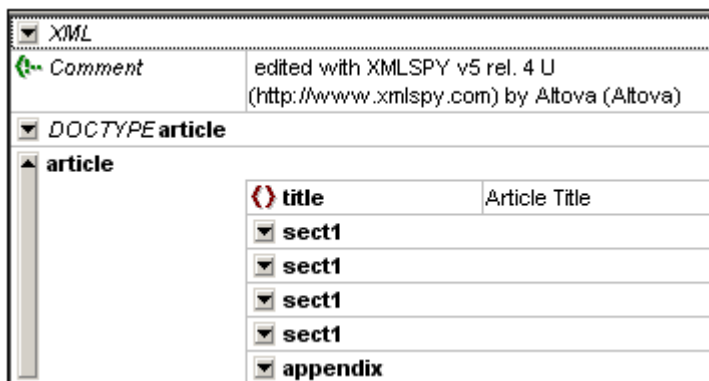
```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <!-- edited with XMLSPY v5 rel. 4 U (ht
3  <!DOCTYPE article SYSTEM "C:\Program Fi
4  <article>
5      <title>Article Title</title>
6      <sect1>
7          <title>Section 1</title>
8          <simpara/>
9      </sect1>
10     <sect1>
11         <title>Section 2</title>
12         <para/>
13     </sect1>
14     <sect1>
15         <title>Section 3</title>
16         <para/>
17         <para/>
18     </sect1>
19     <sect1>
20         <title>Section 4</title>
21         <para/>
22     </sect1>
23     <appendix>
24         <title/>
25         <para/>
26     </appendix>
27 </article>

```

Das Dokument-Element ist `article` und `article` hat die folgende Sequenz von Child-Elementen: ein `title`-Element, vier `sect1`-Elemente und ein `appendix`-Element. Jedes `sect1`- und `appendix`-Element hat ein `title`-Element gefolgt von einer Reihe von `para` oder `simpara`-Elementen.

In der normalen Grid-Ansicht wird dieses Dokument folgendermaßen dargestellt:



Hier sehen Sie nun die Tabellenansicht dieses Dokuments - besser gesagt des Elements `article`. (Um zu dieser Ansicht zu gelangen, wählen Sie das Element `article` in der normalen Grid-Ansicht aus (durch Anklicken) und klicken Sie auf das Symbol **Als Tabelle anzeigen** . Alternativ dazu wählen Sie den Menübefehl **XML | Tabelle | Als Tabelle anzeigen (F12)**.)

	title	sect1	appendix
1	Article Title	sect1	appendix
		sect1	
		sect1	
		sect1	

Beachten Sie, dass jedem Child-Element von `article` (dem als Tabelle angezeigten Element) - also `title`, `sect1` und `appendix` - eine Spalte zugewiesen wurde und dass jede Instanz jedes Child-Typs in der entsprechenden Spalte aufgelistet ist. Beachten Sie außerdem, dass die Tabellenstruktur nur bis zur Ebene der Child-Elemente geht (die `sect1`-Elemente selbst werden nicht als Tabelle dargestellt). Um die `sect1`-Elemente als Tabelle darzustellen, wählen Sie eines der `sect1`-Elemente in der Spalte `sect1` aus und klicken Sie auf . Die `sect1`-Elemente werden nun in Tabellenform dargestellt (siehe Abbildung unten): Jedem der Child-Elemente wird eine Spalte in der `sect1`-Tabelle zugewiesen.

	title	simpara	para
1	Section 1		
2	Section 2		para (1)
3	Section 3		para (2)
4	Section 4		para (1)

Wenn es für eine der vier Instanzen von `sect1` ein Child-Element gibt (`title`, `simpara` oder `para`), so hat die entsprechende Zelle einen weißen Hintergrund (z.B. `simpara` in der ersten Instanz von `sect1`). Wenn es für eine Instanz kein Child-Element gibt, so hat die entsprechende Zelle einen grauen Hintergrund (z.B. `para` in der ersten Instanz von `sect1`). Daraus ersieht man, dass für jeden unterschiedlichen Child-Typ aller `sect1`-Instanzen eine eigene Spalte angelegt wurde.

Bitte beachten Sie:

- Auch Attribute werden als Child Nodes betrachtet. Auch für Attribute werden Spalten erstellt.
- Sie können zwischen der normalen Grid-Ansicht und der Tabellenansicht wechseln, indem Sie das gewünschte Element auswählen und auf klicken oder **F12** drücken.
- Wenn Sie das Dokument in der Tabellenansicht ansehen und in die Textansicht wechseln, wird es beim Zurückwechseln in die Grid-Ansicht in der normalen Grid-Ansicht angezeigt.

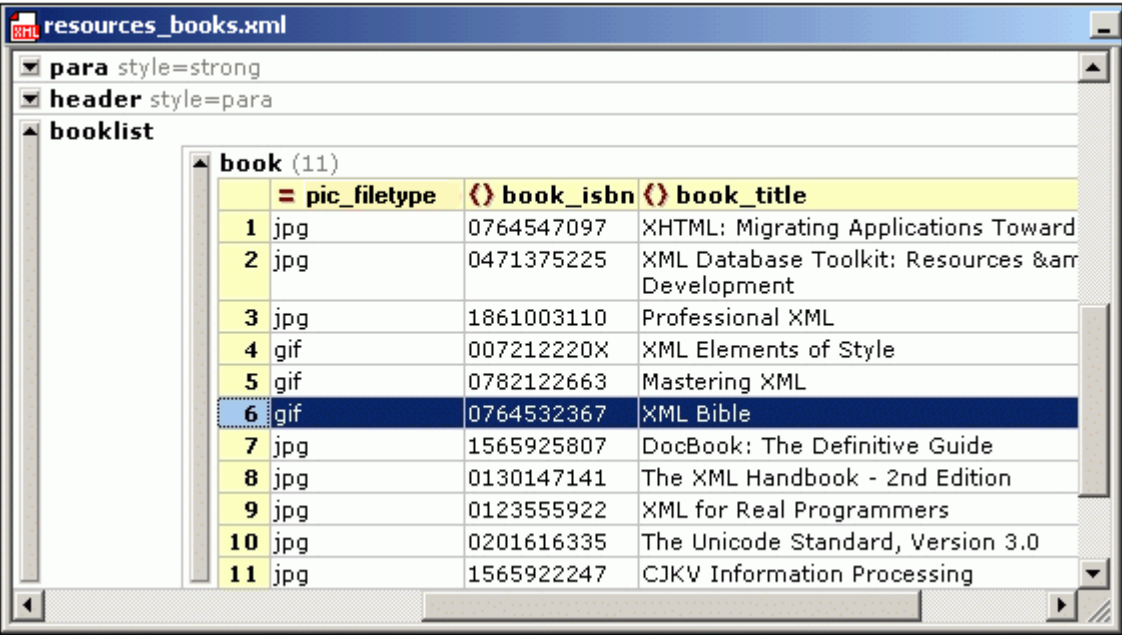
Bearbeitung von Tabellendaten

Sie können Tabellendaten auf folgende Art bearbeiten:

- Spalten mit Drag-and-Drop der Spaltenüberschrift verschieben.
- Spaltendaten für Text-Node mit Hilfe des Menübefehls [XML | Tabelle | Aufsteigend sortieren](#) (auch absteigend sortieren) sortieren.
- Zeilen mit Hilfe des Menübefehls [XML | Tabelle | Zeile einfügen](#) (auch Zeile anhängen) einfügen (oder anhängen).

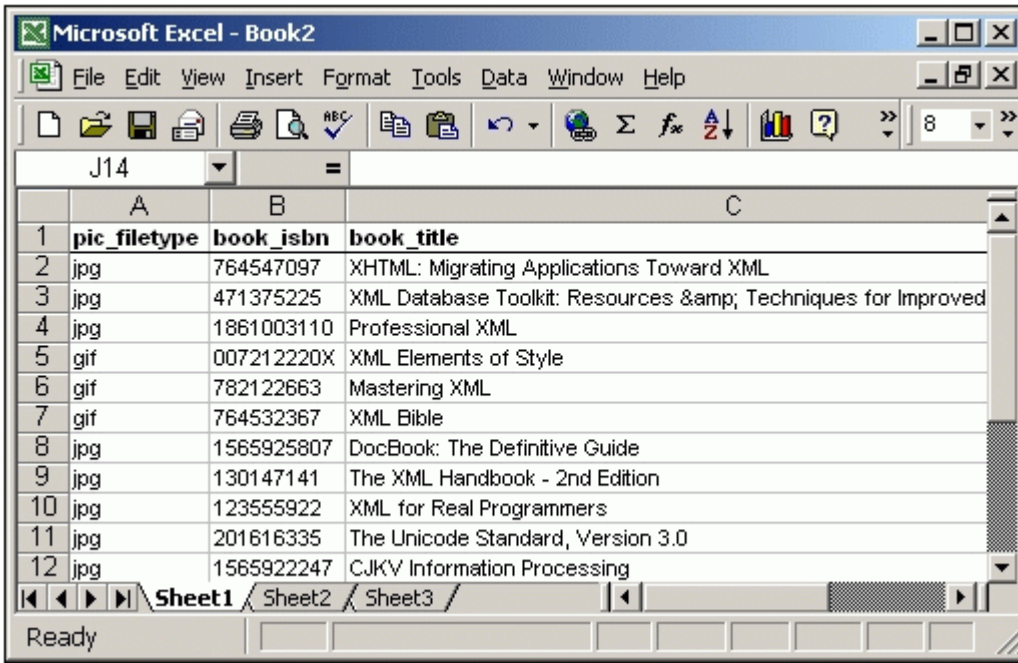
Verschieben von Daten zwischen der Tabellenansicht und externen Applikationen

Die Tabellenstruktur der Tabellenansicht eignet sich auch dazu, Daten aus einem Tabellenkalkulationsprogramm zu übernehmen und umgekehrt. Um Daten aus der Tabellenansicht in eine andere Applikation zu kopieren, wählen Sie die gewünschten Nodes in der Tabelle aus und kopieren Sie die Daten mit Hilfe des Befehls [Als strukturierten Text kopieren](#) direkt in z.B. ein Excel Sheet. (Sie können Nodes in der Tabellenansicht durch Klicken auf die Zelle selbst, auf die Spalten- oder Zeilenüberschrift (siehe unten) oder die gesamte Tabelle auswählen. Wenn Sie die gesamte Tabelle oder Spaltenüberschriften auswählen, wird auch der Text der Spaltenüberschriften ausgewählt; ansonsten nicht)



	= pic_filetype	book_isbn	book_title
1	jpg	0764547097	XHTML: Migrating Applications Toward
2	jpg	0471375225	XML Database Toolkit: Resources & Development
3	jpg	1861003110	Professional XML
4	gif	007212220X	XML Elements of Style
5	gif	0782122663	Mastering XML
6	gif	0764532367	XML Bible
7	jpg	1565925807	DocBook: The Definitive Guide
8	jpg	0130147141	The XML Handbook - 2nd Edition
9	jpg	0123555922	XML for Real Programmers
10	jpg	0201616335	The Unicode Standard, Version 3.0
11	jpg	1565922247	CJKV Information Processing

Im oben gezeigten Screenshot sehen Sie 11 book-Elemente die in der Tabellenansicht in Tabellenform angezeigt werden. Um die gesamte Tabelle in ein Excel Sheet zu kopieren, müssten Sie das Element `book(11)` auswählen, es als strukturierten Text kopieren und in ein Excel Sheet einfügen. Wenn Sie Zelle A1 auswählen und die Tabelle einfügen, sieht das Ergebnis aus, wie unten gezeigt.



	A	B	C
1	pic_filetype	book_isbn	book_title
2	jpg	764547097	XHTML: Migrating Applications Toward XML
3	jpg	471375225	XML Database Toolkit: Resources & Techniques for Improved
4	jpg	1861003110	Professional XML
5	gif	007212220X	XML Elements of Style
6	gif	782122663	Mastering XML
7	gif	764532367	XML Bible
8	jpg	1565925807	DocBook: The Definitive Guide
9	jpg	130147141	The XML Handbook - 2nd Edition
10	jpg	123555922	XML for Real Programmers
11	jpg	201616335	The Unicode Standard, Version 3.0
12	jpg	1565922247	CJKV Information Processing

Der Datenaustausch funktioniert in beide Richtungen. Sie können daher auch Daten aus jedem Tabellenkalkulationsprogramm kopieren und in der Tabellenansicht direkt in eine Tabelle einfügen. Gehen Sie dabei folgendermaßen vor:

1. Wählen Sie in der externen Applikation einen Bereich aus und kopieren Sie ihn (in die Zwischenablage - in Windows-Systemen mit **Strg+C**).
2. Wählen Sie in Ihrem XML-Dokument in der Tabellenansicht eine einzelne Zelle aus.
3. Fügen Sie die kopierten Daten mit **Strg+V** ein.

Die Daten werden in XMLSpy in die Tabelle eingefügt. Die Tabellenstruktur der Originaltabelle bleibt dabei erhalten und die Tabelle wird beginnend in der Zelle eingefügt, die in der Tabellenansicht ausgewählt ist. 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 Zellen benötigt werden, werden diese erstellt.
- Werden jedoch mehr Spalten für die neuen Daten benötigt, werden diese **nicht** erstellt.
- Die Daten in den Zellen werden Inhalt der Elemente, für die die entsprechenden Zellen stehen.

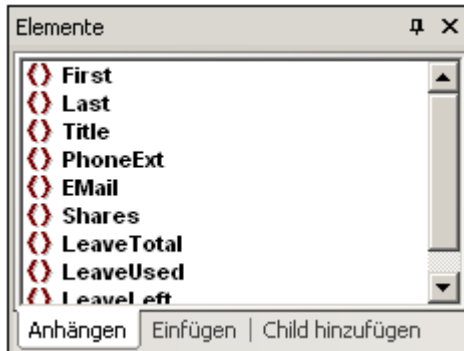
Für komplexere Datenaustauschaufgaben bietet XMLSpy außerdem einige einzigartige Konvertierungsfunktionen zum direkten [Importieren](#) oder [Exportieren](#) von XML-Daten aus jeder beliebigen Textdatei, jedem Word-Dokument oder jeder Datenbank.

4.2.3 Eingabehilfen in der Grid-Ansicht

Element-Eingabehilfe

In der Grid-Ansicht hat die Element-Eingabehilfe drei Register: Anhängen, Einfügen und Child hinzufügen. Auf dem Register **Anhängen** werden Elemente angezeigt, die nach allen Geschwisterelementen des aktuellen Elements angehängt werden können; auf dem Register **Einfügen** werden alle Elemente angezeigt, die vor dem aktuellen Element eingefügt werden können und auf dem Register **Child hinzufügen** werden alle jene Elemente angezeigt, die als

Child-Element des aktuellen Elements eingefügt werden können.

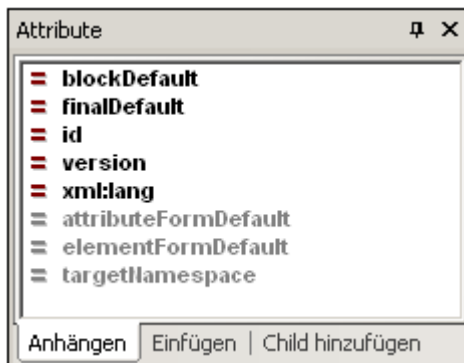


Um ein Element einzufügen, wählen Sie das entsprechende Register und doppelklicken Sie auf das gewünschte Element. Obligatorische Elemente werden durch ein vorangestelltes Ausrufezeichen gekennzeichnet. Geschwisterelemente von zulässigen Elementen, die selbst an der Cursorposition nicht eingefügt werden können, sind ausgegraut und deaktiviert.

Bitte beachten Sie: Im Dialogfeld **Optionen (Extras | Optionen | Bearbeiten)** können Sie festlegen, dass obligatorische Child-Elemente eingefügt werden können, wenn ein Element eingefügt wird.

Attribut-Eingabehilfe

Die Attribut-Eingabehilfe enthält eine Liste aller für das derzeit bearbeitete Element verfügbaren Attribute. Obligatorische Attribute sind durch ein Ausrufezeichen "!" vor dem Namen des Attributs gekennzeichnet. Wurde für dieses Element bereits ein Attribut angegeben, wird dieses Attribut grau angezeigt.



- Um die Attribute auf den Registern "Anhängen" und "Einfügen" zu verwenden, wählen Sie in der Grid-Ansicht ein existierendes Attribut oder ein Element, das ein Child-Element des Parent-Elements des Attributs ist, aus und doppelklicken Sie in der Eingabehilfe auf das gewünschte Attribut.
- Um die Attribute aus dem Register "Child hinzufügen" zu verwenden, wählen Sie in der Grid-Ansicht das Parent-Element des Attributs und doppelklicken Sie in der Eingabehilfe auf das gewünschte Attribut.

Bitte beachten Sie: Existierende Attribute, die bereits hinzugefügt wurden und zu diesem Element kein zweites Mal hinzugefügt werden dürfen, werden grau angezeigt.

Entities-Eingabehilfe

In der Entities-Eingabehilfe werden alle geparsten oder nicht geparsten Entities angezeigt, die innerhalb der Zeile oder in einer externen DTD deklariert sind. Wenn in der Grid-Ansicht ein Text- oder Attribut-Node ausgewählt ist, ist das Register "Child hinzufügen" leer, da solche

Nodes laut Definition keine Child-Elemente enthalten können.

Um in der Grid-Ansicht eine Entity einzufügen, platzieren Sie den Cursor an der gewünschten Position im Textfeld oder wählen Sie das gewünschte Feld aus; wählen Sie anschließend das entsprechende Register aus und doppelklicken Sie auf die Entity. Beachten Sie die folgenden Regeln:

- Wenn der Cursor in ein Textfeld (z.B. auch ein Attributwertfeld) gesetzt wird, wird die Entity an der Cursorposition eingefügt.
- Ist ein Element mit einem nur-Text-Child-Element (d.h. #PCDATA oder simpleType) ausgewählt und der Cursor befindet sich nicht im Textfeld, wird vorhandener Text durch die Entity **ersetzt**.
- Wenn ein nicht-Text-Feld ausgewählt ist, wird die Entity an einer Stelle als Text erstellt, die dem ausgewählten Register der Eingabehilfe entspricht.

Bitte beachten Sie: Wenn Sie eine interne Entity hinzufügen, müssen Sie Ihr Dokument speichern und erneut öffnen, damit die Entity in der Entities-Eingabehilfe angezeigt wird.

4.3 Schema-Ansicht

XML-Schemas können in der Schema-Ansicht angezeigt und bearbeitet werden. Die Schema-Ansicht selbst verfügt über zwei Arten von Ansichten:

- eine **Schemaübersicht**, in der alle globalen Komponenten (globale Elemente, complexTypes usw.) in einer einfachen Tabelle dargestellt werden. ([siehe Abbildung](#))
- Ansichten der Content Models der einzelnen globalen Komponenten (**Content Model-Ansicht**) ([siehe Abbildung](#)).
-

Sie können von der Schema-Übersicht zum Content Model einer bestimmten globalen Komponente wechseln, indem Sie auf das Symbol für diese globale Komponente klicken. Um zur Schema-Übersicht zurückzukehren, klicken Sie auf die Schaltfläche "Globale Elemente anzeigen"

 in der Content Model-Ansicht.

Gliederung dieses Abschnitts

Dieser Abschnitt wurde in die folgenden Unterabschnitte gegliedert

- [Schema-Übersicht](#): Hier wird beschrieben, wie globale Komponenten in der Schema-Übersicht angezeigt werden und wie sie bearbeitet werden.
- [Content Model-Ansicht](#): Hier wird beschrieben, wie die Content Models einzelnen globaler Komponenten bearbeitet werden
- [Eingabehilfen](#): Hier wird erklärt, wie die Eingabehilfen in der Schema-Ansicht strukturiert sind.
- [Identity Constraints](#): Hier wird beschrieben, wie Identity Constraints in der Schema-Ansicht angezeigt und bearbeitet werden.
- [Smart Restrictions](#): Dies ist eine XMLSpy-Bearbeitungsfunktion, mit der die grafische Erstellung und Bearbeitung von Typen, die von ihrem Basistyp abgeleitet wurden, erleichtert wird; in diesem Abschnitt wird beschrieben, wie die Smart Restrictions-Funktion verwendet wird
- Eine Anleitung, wie Sie [die mit dem Präfix xml: versehenen Attribute](#) `base`, `id`, `lang` und `space` mit grafischen Mitteln zu Schemakomponenten hinzufügen können.
- [Zurück und Vorwärts: Navigieren von einer Position zu einer anderen](#): Hier wird eine Funktion der Schema-Ansicht erläutert, mit der Sie zwischen zuvor angezeigten Positionen navigieren können.

Herstellen einer Verbindung zu SchemaAgent

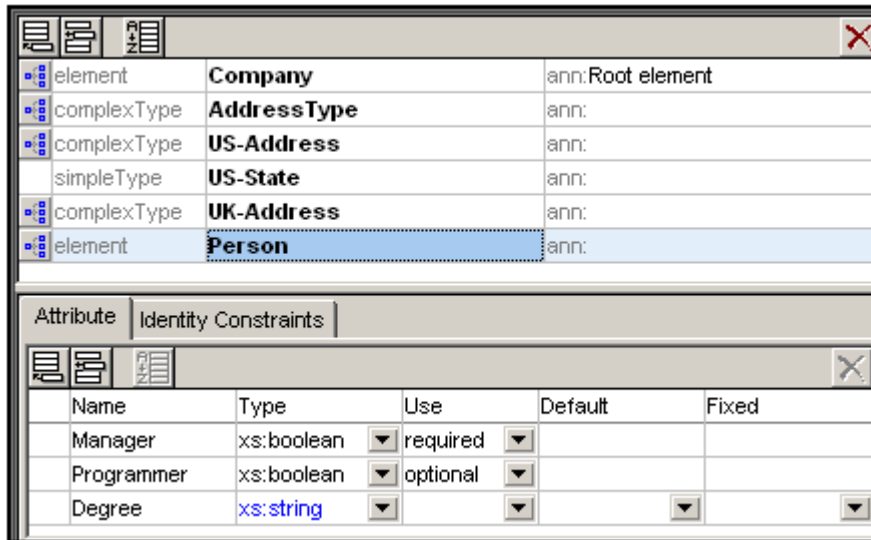
Sie können von XMLSpy aus eine Verbindung zu SchemaAgent herstellen, um Komponenten aus anderen Schemas auf der GUI darzustellen und diese Komponenten im Schema, das Sie gerade bearbeiten zu verwenden. Eine Anleitung zum Arbeiten mit in XMLSpy finden Sie im Abschnitt [Arbeiten mit SchemaAgent](#).

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](#).

4.3.1 Schema-Übersicht

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



Sie können globale Komponenten einfügen, anhängen oder löschen und ihre Eigenschaften bearbeiten. Verwenden Sie zum Einfügen, Anhängen oder Löschen die entsprechenden Schaltflächen am oberen Rand der Schema-Übersicht. Um Eigenschaften zu bearbeiten, wählen Sie die gewünschte Komponente in der Schema-Übersichtsliste aus und editieren Sie die Eigenschaften entweder über die Eingabehilfen (im rechten Bereich der Ansicht) oder das Attribute-/Identity Constraints- Fenster (im unteren Bereich der Ansicht).

Sie können das Content Model einer globalen Komponente in der [Content Model-Ansicht](#) bearbeiten, die Sie durch Klicken auf das Symbol "Content Model-Ansicht"  links von der globalen Komponente aufrufen können.

In der Schema-Übersicht stehen Ihnen die folgenden Editierfunktionen zur Verfügung:

- Sie können Komponenten in der Schema-Übersichtsliste mit Drag & Drop verschieben.
- Sie können mit Hilfe der Pfeiltasten navigieren.
- Sie können globale Komponenten, Attribute und Identity Constraints mit Hilfe der Funktionen Ausschneiden/Kopieren-und-Einfügen an eine andere Stelle verschieben und oder von einem Schema in ein anderes verschieben bzw. kopieren.
- Durch Rechtsklick auf eine Komponente öffnen Sie ein Kontextmenü, über das Sie die Annotationsdaten dieser Komponente ausschneiden, kopieren, einfügen oder bearbeiten können.

Globale Komponenten in der Schema-Übersicht

Auf der obersten Ebene eines XML-Dokuments (d.h. auf Ebene der Child-Elemente des `schema`-Elements) können die folgenden fünf Basiskomponenten definiert werden:

- Annotation
- Typ-Definition (simple oder complex)
- Deklaration (Element oder Attribut)
- Attribute Group
- Model Group

Diese Komponenten auf der obersten Ebene werden **globale Komponenten** genannt. In der **Schema-Übersicht** werden allen globalen Komponenten in Ihrem Schema in Tabellenform aufgelistet. Einige globale Komponenten (wie z.B. complexTypes, Element-Deklarationen und Model Groups) können ein Content Model haben, das Struktur und Inhalt der Komponente beschreibt. Andere globale Komponenten (wie z.B. Annotationen, simpleTypes und Attribut-Gruppen) haben kein Content Model. Komponenten, die ein Content Model haben können, sind durch ein  Symbol links vom Komponentennamen gekennzeichnet. Wenn Sie auf dieses Symbol klicken, wird die [Content Model-Ansicht](#) für diese globale Komponente angezeigt.

Grundlegende Begriffe

- *SimpleType und complexType.* Ein simpleType dient dazu, alle Attribute und Elemente zu definieren, die nur Text und kein Attribut enthalten. Ein simpleType hat daher kein Content Model, sondern nur Text (der durch den Datentyp eingeschränkt werden kann). Ein complexType hat dagegen mindestens ein Child-Element oder -Attribut. Wenn Sie ein Child-Element für ein Element deklarieren, wird diesem automatisch der Typ complexType zugewiesen.
- *Globale und lokale Komponenten.* Bei einer globalen Komponente kann es sich um jede der fünf oben angeführten Arten handeln. Eine globale Komponente kann in der Schema-Übersicht definiert werden und scheint daraufhin sofort in der Liste der globalen Komponenten in der Schema-Übersicht auf. Wenn es sich bei der globalen Komponente um einen complexType, eine Element-Deklaration oder eine Model Group handelt, können Sie anschließend ihr Content Model definieren, indem Sie es in der Content Model-Ansicht bearbeiten. Sobald eine globale Komponente definiert ist, kann sie von lokalen Komponenten referenziert werden. Eine lokale Komponente wird direkt im Content Model einer Komponente erstellt. Beachten Sie, dass eine lokale Komponente in der Content Model-Ansicht (über das Kontextmenü) in eine globale Komponente konvertiert werden kann.

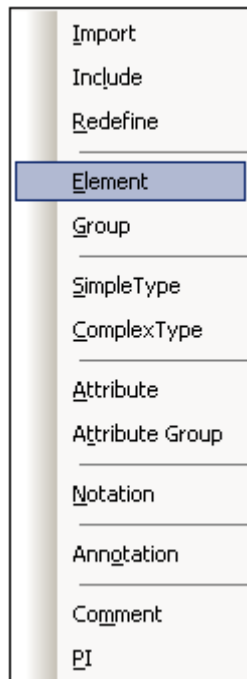
Kommentare und Processing Instructions

Kommentare und Processing Instructions (Verarbeitungsanweisungen) innerhalb von simpleTypes und complexTypes in XML-Schema-Dokumenten werden gesammelt und an das Ende des einschließenden Objekts verschoben. In solchen Fällen wird daher empfohlen, Annotationen anstelle von Kommentaren zu verwenden.

Erstellen globaler Komponenten

So erstellen Sie in der Schema-Übersicht eine globale Komponente:

1. Klicken Sie oben in der Schema-Übersicht auf das Symbol "Einfügen"  oder "Anhängen" . Daraufhin wird ein Menü angezeigt, aus dem Sie verschiedene Komponententypen (Element, simple type, complex type, model group usw.) auswählen können.



2. Wählen Sie den gewünschten Komponententyp aus. In der Liste der globalen Komponenten wird ein Eintrag dieses Typs erstellt.
3. Geben Sie den Namen der Komponente hier ein und drücken Sie die **Eingabetaste**. Der Name der neuen globalen Komponente wird zur entsprechenden Liste bzw. zu den entsprechenden Listen (Elm, Grp, Com, Sim, usw.) in der Komponenten-Navigator-Eingabehilfe hinzugefügt. Sie können das Content Model der neuen globalen Komponente entweder durch Doppelklicken auf den Komponentennamen im Komponenten-Navigator oder durch Klicken auf das  Symbol links vom Komponentennamen in der Liste der globalen Komponenten bearbeiten.

Bitte beachten Sie:

- Sie können außerdem eine globale Komponente erstellen, während Sie in der Content Model-Ansicht arbeiten. Klicken Sie mit der rechten Maustaste auf eine beliebige Stelle im Fenster und wählen Sie **Neues Global | Element**.
- Beim Bearbeiten in der Content Model-Ansicht können Sie ein lokales Element zu einem globalen - oder sogar zu einem complexType machen, wenn das Element ein Child-Element oder -Attribut hat. Wählen Sie das lokale Element aus, rechtsklicken Sie in das Fenster und wählen Sie **Als global definieren | Element** oder **Als global definieren | Complex Type**.

Löschen globaler Komponenten

So löschen Sie eine globale Komponente:

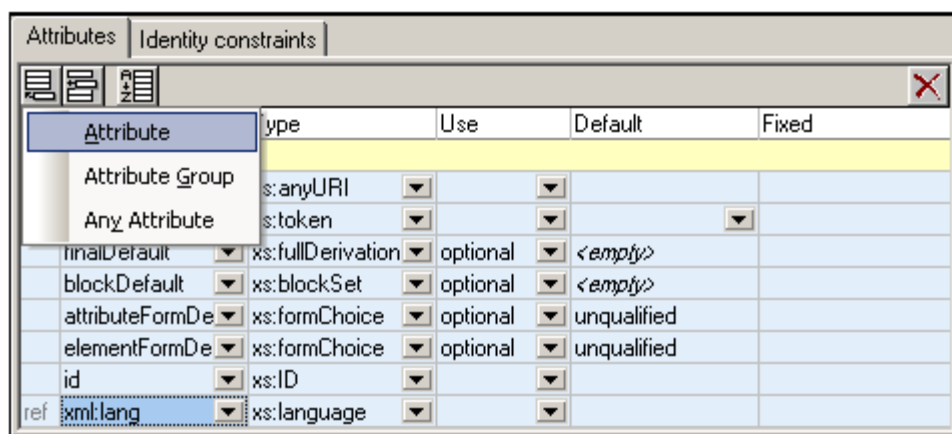
1. Wählen Sie die globale Komponente in der Schema-Übersicht in der Liste der globalen Komponenten aus.
2. Drücken Sie die **Entfernen**-Taste oder klicken Sie am oberen Rand der Schema-Übersicht auf das Symbol **Löschen** .

Attribute und Identity Constraints von Komponenten

Sie können Attribute und Identity Constraints für Komponenten entweder in der Schema-Übersicht oder in der Content Model-Ansicht definieren. In der Schema-Übersicht werden die Attribute und Identity Constraints einer Komponente im Attribute-/Identity Constraints-Fenster im unteren Bereich des Schema-Übersichtsfensters angezeigt und können hier editiert werden. In der Content Model-Ansicht können Attribute und Identity Constraints entweder im Attribute-/Identity Constraints-Fenster am unteren Rand des Content Model-Fensters oder innerhalb des Content Model-Diagramms selbst angezeigt werden, wo Sie diese bearbeiten können. Über eine Einstellung im [Dialogfeld Schema-Ansicht konfigurieren](#), die Sie über den Menübefehl **Schema-Design | Ansicht konfigurieren** aufrufen, können Sie einstellen, wie Attribute und Identity Constraints in der Content Model-Ansicht angezeigt werden sollen.

Definieren von Attributen für eine Komponente

Attribute für eine Komponente lassen sich über das Attribute-/Identity Constraints-Fenster am unteren Rand des Schema-Übersichtsfensters definieren.



So definieren Sie Attribute für eine globale Komponente, für die Attribute zulässig sind:

1. Wählen Sie die globale Komponente in der Liste der globalen Komponenten aus.
2. Wählen Sie im Attribute-/Identity Constraints-Fenster das Register "Attribute".
3. Klicken Sie auf das Symbol "Anhängen"  oder "Einfügen"  links oben auf dem Register "Attribute".
4. Wählen Sie im Popup-Menü, das daraufhin angezeigt wird, den gewünschten Attributtyp aus. In der Attributliste wird ein Eintrag erstellt.
5. Geben Sie in dem soeben erstellten Eintrag die Eigenschaften des Attributs an.

Bitte beachten Sie: Sie können Attribute für globale Komponenten auch in der Content Model-Ansicht definieren. Wählen Sie die globale Komponente aus und definieren Sie anschließend Attribute wie oben beschrieben.

Definieren von Identity Constraints für eine Komponente

Zum Definieren von Identity Constraints für eine Komponente wird das Attribute-/Identity Constraints-Fenster im unteren Bereich des Schema-Übersichtsfensters verwendet. Eine Beschreibung zur Verwendung von Identity Constraints finden Sie unter [Identity Constraints](#).

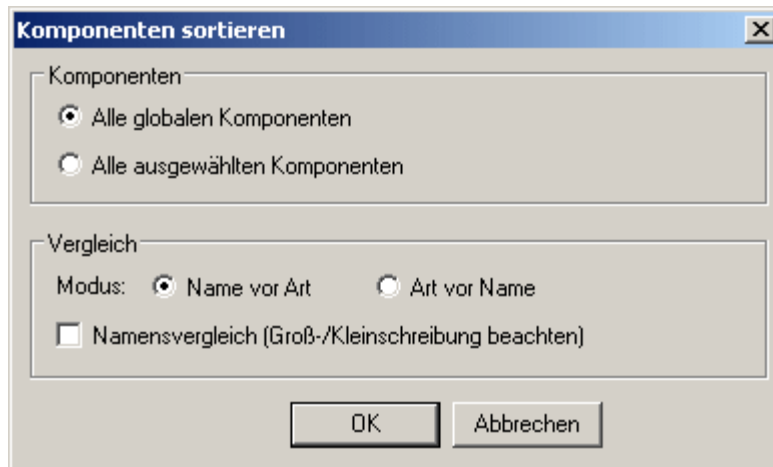
Sortieren von globalen Komponenten, Attributen und Identity Constraints

Globale Komponenten können alphabetisch, in aufsteigender Reihenfolge, nach lokalem Namen und Art sortiert werden. Befolgen Sie dazu folgende Methode:

1. Wenn Sie eine Auswahl von globalen Komponenten sortieren möchten, markieren Sie

die globalen Komponenten, die Sie sortieren wollen, indem Sie sie mit gedrückter **Strg**-Taste anklicken. (Sie können auch einen Bereich markieren, indem Sie die **Umschalttaste** drücken und das erste und das letzte Element des Bereichs anklicken.) Die ausgewählten globalen Komponenten werden markiert. Wenn Sie alle globale Elemente sortieren möchten, können Sie eine oder mehrere globale Komponenten markieren. Eine Option zum Sortieren aller Komponenten ist verfügbar.

2. Klicken Sie auf das Symbol **Sortieren**  am oberen Rand des Schema-Übersichtsfensters. Dadurch wird das Dialogfeld "Komponenten sortieren" aufgerufen (*Abbildung unten*).



3. Wählen Sie im Bereich "Komponenten" ob alle Komponenten oder nur die Auswahl sortiert werden soll. Die Option *Alle ausgewählten Komponenten* ist nur aktiviert wenn die Anzahl der markierten globalen Komponenten größer als eins ist.
4. Im Bereich "Vergleich" können Sie wählen, ob die Sortierung zuerst anhand des lokalen Namens oder der Komponentenart durchgeführt werden soll. Wenn es mehr als eine Komponente mit dem gleichen lokalen Namen oder der gleichen Komponentenart gibt (je nachdem, was die erste Bedingung ist), werden diese nach dem zweiten Kriterium sortiert.
5. Wenn beim Namensvergleich die Groß-/Kleinschreibung beachtet werden soll, aktivieren Sie das entsprechende Kontrollkästchen. Wenn diese Option ausgewählt ist, wird Kleinschreibung vor Großschreibung aufgelistet.

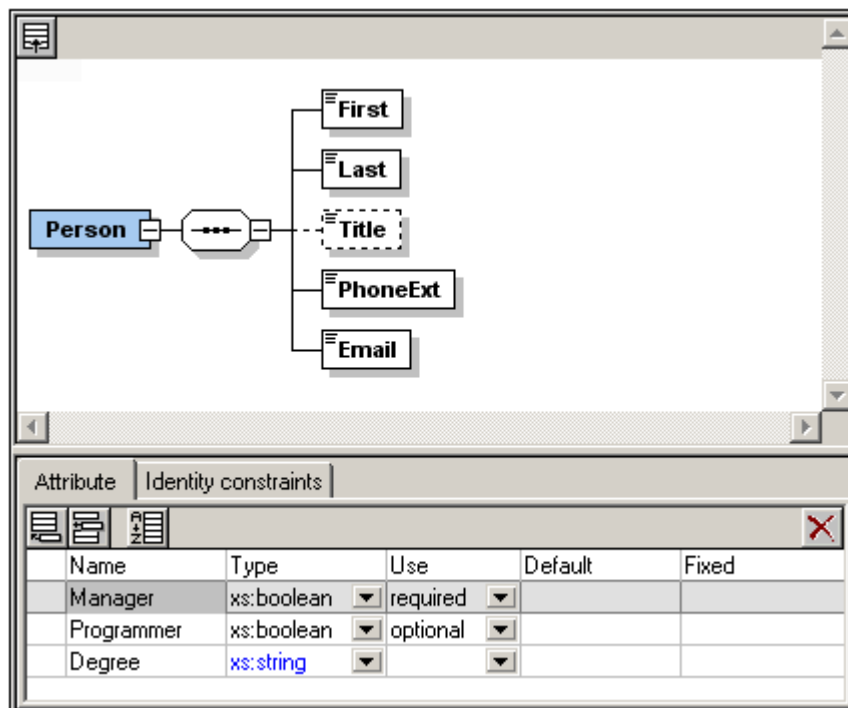
Das **Sortieren von Attributen und Identity Constraints** funktioniert auf gleiche Weise wie das Sortieren von globalen Komponenten. Das Symbol **Sortieren**  befindet sich in der Symboleiste des Attribut/Identity Constraints-Fensters und ist aktiviert, wenn das Fenster zumindest zwei gleichrangige benannte Komponenten (wie Attribute oder Schlüssel) enthält.

4.3.2 Content Model-Ansicht

Ein Content Model ist eine Beschreibung der Struktur und des Inhalts eines Elements. Globale Komponenten, die ein Content Model haben, (wie z.B. Elemente, complexTypes und Model Groups), sind durch ein  Symbol links vom Komponentennamen gekennzeichnet. Wenn Sie auf dieses Symbol klicken, wird die Content Model-Ansicht für diese globale Komponente angezeigt. Alternativ dazu (i) wählen Sie eine Komponenten aus und anschließend die Menüoption **Schema-Design | Diagramm anzeigen**, oder (ii) doppelklicken Sie im [Komponentennavigator](#) (standardmäßig die Eingabehilfe oben rechts) 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 Komponenten innerhalb des aktuellen Modells navigieren, indem Sie die **Strg**-Taste gedrückt

halten und auf die gewünschte Komponente doppelklicken.

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](#)) anpassen.



Die Content Model-Ansicht verfügt über die folgenden Editierfeatures:

- Jede Ebene (Elemente oder Elementgruppen) im Baumdiagramm ist mittels eines Kompositors mit benachbarten Ebenen verbunden.
- Sie können Objekte des Baumdiagramms (Kompositoren, Elemente, Elementgruppen) mittels Drag-and-Drop verschieben.
- Sie können Objekte aus der Baumstruktur mit Hilfe der Tastenkombinationen Strg + C und Strg + V kopieren bzw. einfügen
- Objekte (Kompositoren, Elemente und Elementgruppen) können über das Kontextmenü (wird durch Rechtsklick auf ein Objekt aufgerufen) hinzugefügt werden.
- 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) Komponententyp.
- Sie können die Eigenschaften eines Objekts in der Details-Eingabehilfe (Kompositoren, Elemente, Elementgruppen) und im Attribute-/Identity Constraints-Fenster bearbeiten.
- Die Attribute und Identity Constraints einer Komponente werden in einem Fenster am unteren Rand des Hauptfensters angezeigt. Attribute und Identity Constraints können auch im Content Model-Diagramm anstatt des Attribute/Identity Constraints-Fensters angezeigt werden. Diese Ansichtsoption lässt sich im Dialogfeld [Schema-Ansicht konfigurieren](#) einstellen.

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

Um zur Schema-Übersicht zurückzukehren, klicken Sie auf das Symbol **Globale Elemente**

anzeigen  oder wählen Sie die Menüoption **Schema-Design | Globale Elemente anzeigen**

Bearbeiten in der Content Model-Ansicht

In der **Content Model-Ansicht** können Sie das Content Model der folgenden drei Komponententypen grafisch mit wenigen Mausklicks definieren:

- ComplexTypes,
- Element-Deklarationen und
- Model Groups

Alle anderen Schema-Komponenten (Annotationen, Attribut-Deklarationen, simpleTypes usw.) haben kein Content Model.

In der Content Model-Ansicht werden die verschiedenen Teile des Content Model grafisch dargestellt. Diese Teile lassen sich in zwei große Gruppen unterteilen: **Kompositoren** und **Komponenten**. Normalerweise wird ein Kompositor hinzugefügt und anschließend die gewünschten Child-Komponenten.

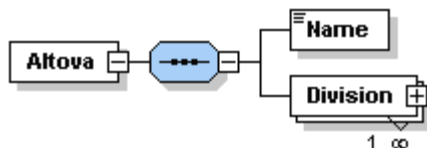
Kompositoren

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:

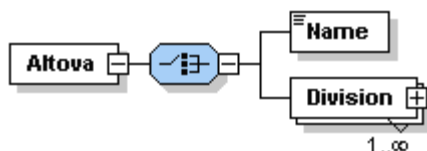
1. Rechtsklicken Sie auf das Element, zu dem Sie Child-Elemente hinzufügen möchten.
2. Wählen Sie den Befehl **Child hinzufügen | Sequence** (oder **Choice** oder **All**).

Daraufhin wird ein Kompositor hinzugefügt, der folgendermaßen aussieht:

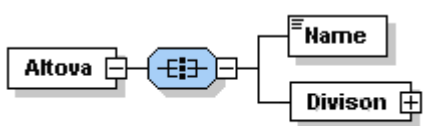
- **Sequence**



- **Choice**



- **All**

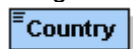


Um den Kompositor zu ändern, rechtsklicken Sie darauf und wählen Sie **Modell ändern | Sequence** (oder **Choice** oder **All**). Nachdem Sie den Kompositor hinzugefügt haben, müssen Sie (ein) Child-Element(e) oder eine Model Group hinzufügen.

Komponenten im Content Model

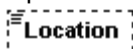
Unten finden Sie eine Liste von Komponenten, die in Content Models verwendet werden. Die grafische Darstellung der einzelnen Komponenten gibt Ihnen genaue Informationen über die Art der Komponente und ihre strukturellen Eigenschaften.

- Obligatorisches Einzelelement



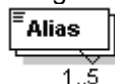
Details: Das Rechteck zeigt an, dass es sich um ein Element handelt, die durchgezogene Umrandung bedeutet, dass das Element "required" also obligatorisch ist. Das Fehlen eines Zahlenbereichs weist darauf hin, dass es ein Einzelelement ist (d.h. `minOcc=1` und `maxOcc=1`). Der Name des Elements ist `Country`. 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



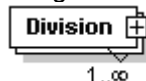
Details: Rechteck = Element; gestrichelte Umrandung = optional; fehlender Zahlenbereich = `minOcc=0` und `maxOcc=1`. Der Element-Name ist `Location`.
Bitte beachten Sie: Im Kontextmenü können Sie ein obligatorisches Element über die Menüoption **Optional** in ein optionales umwandeln.

- Obligatorisches mehrmals vorkommendes Element

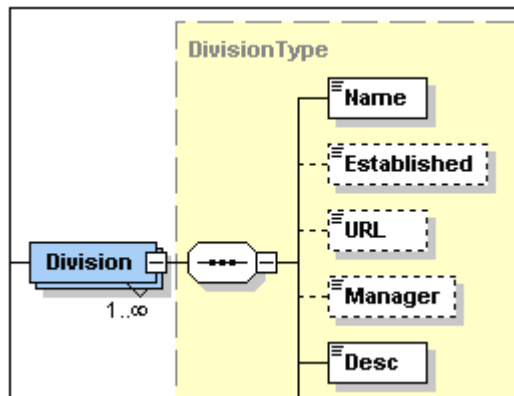


Details: Rechteck = Element; durchgezogene Umrandung = obligatorisch; Zahlenbereich 1..5 = `minOcc=1` und `maxOcc=5`. Der Element-Name ist `Alias`.

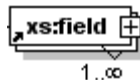
- Obligatorisches mehrfach vorkommendes Element, das Child-Elemente enthält



Details: Rechteck = Element; durchgezogene Umrandung = obligatorisch; Zahlenbereich 1..unendlich = `minOcc=1` und `maxOcc=unbounded`; Plus-Zeichen = complex content (d.h. es hat mindestens ein Child-Element oder -Attribut). Der Element-Name ist `Division`.
Bitte beachten Sie: 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.



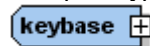
- Element, das ein globales Element referenziert



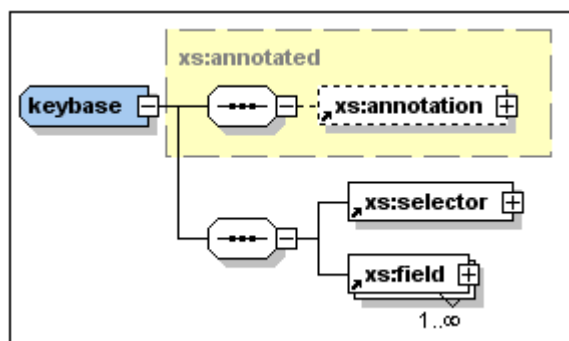
Details: Pfeil in der linken unteren Ecke = Element, das ein globales Element referenziert; Rechteck = Element; durchgezogene Umrandung = obligatorisch; Zahlenbereich 1..unendlich = $\minOcc=1$ und $\maxOcc=unbounded$; Plus-Zeichen = complex content (d.h. mindestens ein Child-Element oder -Attribut). Der Element-Name ist `xs:field`.

Bitte beachten Sie: 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 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 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.

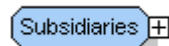


Der oben dargestellte complexType `keybase` wurde in der Schema-Übersicht mit einem Base Type `xs:annotated` deklariert. Der Base Type wird als Rechteck mit einer grauen gestrichelten Umrandung und gelbem Hintergrund dargestellt. Anschließend wurden in der Content Model-Ansicht die Child-Elemente `xs:selector` 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.

Bitte beachten Sie: Der Base Type eines Content Models wird als Rechteck mit grauer 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.

Bitte beachten Sie: 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.

Bitte beachten Sie: 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.

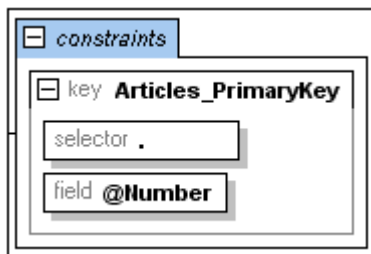
- Attribute



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.

Bitte beachten Sie: Attribute können in der Details-Eingabehilfe bearbeitet werden. Sie können Attribute im Diagramm der Content Model-Ansicht oder in einem Fenster unterhalb der Content Model-Ansicht anzeigen. Durch Klicken auf das Symbol "Attribute anzeigen"  können Sie zwischen den beiden Ansichten hin- und herwechseln. Wenn Attribute im Diagramm der Content Model-Ansicht angezeigt werden, werden alle Attribute eines Einzelements in einem Rechteck mit gestrichelter Umrandung dargestellt. 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.

- Identity Constraints



Details: Identity Constraints werden durch das kursiv gedruckte Wort 'constraints' in einem erweiterbaren Rechteck dargestellt.

Bitte beachten Sie: 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 im Content Model-Diagramm nicht angezeigt, jedoch in der Details-Eingabehilfe. 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 "Constraints anzeigen"  ein- und ausblenden..

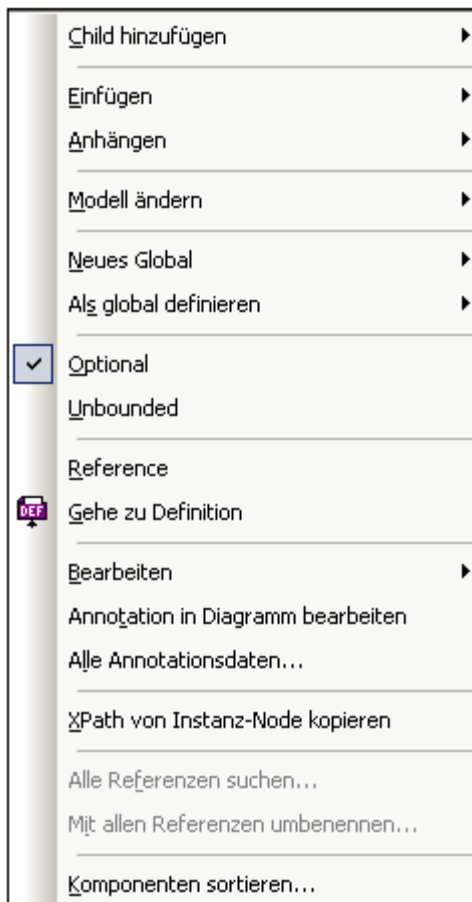
Bitte beachten Sie:

- Durch Klicken auf das Symbol "Vordefinierte Details hinzufügen"  können Sie Eigenschaften-Beschreibungen, die Sie im [Dialogfeld Schema-Ansicht konfigurieren](#) definiert haben, ein oder ausblenden.
- Durch Auswahl der Symbole zur Anzeige von Attributen bzw. Constraints im Diagramm anzeigen  und  können Sie Attribute und Identity Constraints wahlweise entweder im Diagramm des Content Models selbst oder in einem Fenster unterhalb der Content Model-Ansicht anzeigen lassen.
- 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. Durch Doppelklicken auf den Namen eines Base Type können Sie das Content Model dieses Base Type aufrufen.

Andere Editiermöglichkeiten in der Content Model-Ansicht

Die Editierbefehle für die Content Model-Ansicht können über das Kontextmenü (*siehe Screenshot unten*) aufgerufen werden, das Sie durch einen Rechtsklick in die Content Model-Ansicht öffnen können. Im Folgenden wird beschrieben, welche Funktionen Ihnen zur Verfügung stehen.



Hinzufügen von Child-Kompositoren/Komponenten und Einfügen/Anhängen von Kompositoren/Komponenten

1. Rechtsklicken Sie auf den Kompositor bzw. die Komponente. Daraufhin wird das Kontextmenü (in dem nur die zulässigen Befehle aktiv sind) geöffnet.
2. Wählen Sie den gewünschten Befehl aus.

Ändern eines Kompositors

1. Rechtsklicken Sie auf den gewünschten Kompositor.
2. Wählen Sie die Kontextmenüoption **Modell ändern** und wählen Sie im Untermenü den Kompositor aus, den Sie verwenden möchten. (Der aktuell ausgewählte Kompositor ist durch ein Häkchen markiert.) Wenn ein Kompositor an dieser Stelle nicht verwendet

werden darf, ist er ausgegraut.

Erstellen globaler Komponenten

- Um eine globale Komponente zu erstellen, rechtsklicken Sie an eine beliebige Stelle in der Content Model-Ansicht, wählen Sie den Befehl **Neues Global** und aus dem Untermenü die gewünschte Komponente aus.
- Um ein lokales Element zu einem globalen Element oder complexType zu machen, rechtsklicken Sie auf das lokale Element, wählen Sie **Als global definieren** und wählen Sie im Untermenü entweder **Element** oder **Complex type** aus. Wenn eine dieser Komponenten an dieser Stelle nicht erstellt werden kann, ist sie ausgegraut.

Ändern der "Occurrence"-Definition

Sie können zwischen einem "minimum occurrence" und "maximum occurrence" Wert eines Kompositors zwischen 0 und 1 (für `minOccurs`) bzw. 1 und `unbounded` (für `maxOccurs`) wechseln. Gehen Sie dazu vor wie folgt:

1. Rechtsklicken Sie auf den Kompositor oder die Komponente, für die der Occurrence-Wert geändert werden muss.
2. Wählen Sie die Kontextmenüoption **Optional**, um den `minOccurs`-Wert auf 1 zu setzen. Wählen Sie die Menüoption **Unbounded**, um den `maxOccurs` Wert auf `unbounded` einzustellen. Deaktivieren Sie die Option **Unbounded**, um `maxOccurs` auf 1 zu setzen. Die jeweils ausgewählte Option ist links neben dem Menüeintrag durch ein Häkchen gekennzeichnet.

Wechseln zwischen einer lokalen und einer globalen Definition

Wenn ein globales Element vorhanden ist, das denselben Namen wie ein lokales Element hat, können Sie die globale Definition referenzieren oder die lokale Definition verwenden. Gehen Sie dabei vor wie folgt:

1. Rechtsklicken Sie auf das Element.
2. Wählen Sie die Kontextmenüoption **Reference**. Wenn das globale Element referenziert wird, wird dieser Menüeintrag mit einem Häkchen versehen. Wenn die lokale Definition verwendet wird, erhält der Menüeintrag **Reference** kein Häkchen.

Zu einer Definition springen

Sie können von einem Content Model zur Definition jeder globalen Komponente springen, die in diesem Content Model enthalten ist. Gehen Sie dazu vor wie folgt:

1. Rechtsklicken Sie auf die globale Komponente. Dabei kann es sich um das gelbe Rechteck eines Base Type, ein Element, das ein globales Element referenziert oder eine Model Group handeln.
2. Wählen Sie die Kontextmenüoption **Gehe zu Definition**. Daraufhin wird die Content Model-Ansicht der globalen Komponente geöffnet.

Alternativ dazu doppelklicken Sie auf den Namen des Base Type oder drücken Sie die Strg-Taste und doppelklicken Sie auf das referenzierende Element oder die Model Group.

Bearbeiten von Elementnamen und -typen

1. Rechtsklicken Sie auf das Element.
2. Wählen Sie die Kontextmenüoption **Bearbeiten | Name** und ändern Sie den Namen oder wählen Sie **Bearbeiten | Typ** und ändern Sie den Typ.

Alternativ dazu können Sie auch auf den Elementnamen doppelklicken und die Änderung eingeben.

Erstellen und Bearbeiten von Dokumentation für einen Kompositor oder eine Komponente

Sie können zu einzelnen Kompositoren und Komponenten als Hilfe für Schema-Editoren Dokumentation hinzufügen. Gehen Sie dazu vor wie folgt:

1. Rechtsklicken Sie auf den Kompositor bzw. die Komponente.



2. Wählen Sie die Kontextmenüoption **Annotation in Diagramm bearbeiten**. Daraufhin wird der Raum unterhalb des Kompositors/der Komponente markiert und Sie können beschreibenden Text zum Kompositor bzw. der Komponente eingeben. In der Textansicht wurden dadurch die Elemente `annotation` und `annotation/documentation` erstellt. Das Element `documentation` enthält den von Ihnen eingegebenen beschreibenden Text.

Alternativ dazu können Sie auf den Kompositor/die Komponente rechtsklicken und die Menüoption "Alle Annotationsdaten" wählen. Im Dialogfeld "Annotation", das daraufhin angezeigt wird, können Sie ein Dokumentationselement anhängen oder einfügen und Inhalt dafür eingeben.

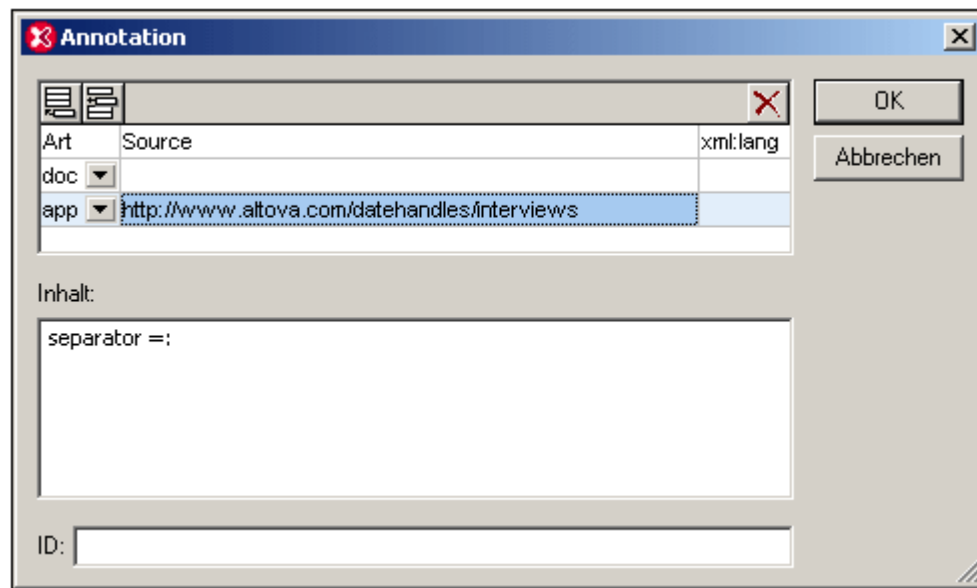
Um bereits vorhandenen Dokumentationstext zu bearbeiten, können Sie eine der oben beschriebenen Methoden verwenden, schneller ist es jedoch, wenn Sie im Diagramm auf die Annotation doppelklicken und diese direkt bearbeiten.

Erstellen und Bearbeiten von Applikationsinformationen für einen Kompositor oder eine Komponente

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.



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.

XPath von Instanz-Node kopieren

Der Kontextmenübefehl **XPath von Instanz-Node kopieren** ist für Elemente und Attribute, die innerhalb eines globalen Elements oder eines complexType definiert wurden, aktiv. Er kopiert einen XPath-Ausdruck, der den ausgewählten Node identifiziert, in die Zwischenablage.

Alle Referenzen suchen und Mit allen Referenzen umbenennen

Die beiden Kontextmenübefehle **Alle Referenzen suchen** und **Mit allen Referenzen umbenennen** sind für globale Elemente aktiviert. Sie dienen zum Suchen bzw. Umbenennen der ausgewählten globalen Komponente im aktiven Dokument und optional dazu in allen mit dem aktiven Dokument in Zusammenhang stehenden Schema-Dateien. Nähere Informationen dazu siehe [Suchen und Umbenennen von globalen Komponenten](#).

Sortieren von Deklarationen und Referenzen

Der Befehl **Sortieren** ist aktiv, wenn ein Node (Element oder Attribut) mehrere Deklarationen oder Referenzen hat. Die Sortierung funktioniert auf dieselbe Art und Weise wie die im Abschnitt [Schema-Übersicht](#) beschriebene Sortierung.

Verwendung von Tastenkürzeln in der Content Model-Ansicht

Sie können Elemente in der Content Model-Ansicht mit den Kürzeln **Strg + C** und **Strg + V** kopieren und einfügen.

So kopieren Sie Elemente und fügen diese ein:

1. Wählen Sie die zu kopierenden Elemente aus. Klicken Sie auf ein Element, um es auszuwählen. Um mehr als ein Element auszuwählen, klicken Sie auf das erste

- Element und halten Sie die Umschalttaste gedrückt.
2. Drücken Sie **Strg + C**. Die Elemente werden in die Zwischenablage kopiert.
3. Wählen Sie den Kompositor aus, in den Sie die Elemente kopieren möchten.
4. Drücken Sie **Strg + V**. Die Elemente werden als Child-Elemente des Kompositors eingefügt.

So kopieren Sie Elemente und fügen Sie in umgekehrter Reihenfolge ein:

1. Klicken Sie auf das unterste Element, das Sie kopieren möchten und wählen Sie weitere Elemente mit Hilfe der Umschalttaste und des nach-oben-Pfeils aus.
2. Drücken Sie **Strg + C**. Die Elemente werden in die Zwischenablage kopiert.
3. Wählen Sie den Kompositor aus, in den Sie die Elemente kopieren möchten.
4. Drücken Sie **Strg + V**. Die Elemente werden so eingefügt, dass das Element, welches zuvor das oberste war, als unterstes eingefügt wird.

Informationen zu XML-Schema-Annotationen

XML-Schema-Annotationen werden im `annotation`-Element gespeichert. Es gibt zwei Arten von Annotationen. Beide sind Elemente des `annotation`-Elements:

- Kompositor- oder Komponenten-Dokumentation: Enthält Informationen, die für Schema-Editoren nützlich sein könnten. Diese Informationen befinden sich im `documentation` Child-Element von `annotation`.
- Applikationsinformationen: Hier können ein Skript oder Daten eingefügt werden, die von einer verarbeitenden Applikation gelesen werden können; diese Informationen befinden sich im `appinfo`-Child-Element von `annotation`.

Unterhalb finden Sie den Text eines `annotation`-Elements. Er beruht auf dem Beispiel in der obigen Beschreibung zum Erstellen von Dokumentations- und Applikationsinformationen.

```
<xs:element name="session_date" type="xs:dateTime" nillable="true">
  <xs:annotation>
    <xs:documentation>Datum und Zeit des
Interviews</xs:documentation>
    <xs:appinfo
source="http://www.altova.com/datehandlers/interviews">separator =
: </xs:appinfo>
  </xs:annotation>
</xs:element>
```

Kommentare und Processing Instructions

Kommentare und Processing Instructions (Verarbeitungsanweisungen) innerhalb von `simpleTypes` und `complexType`s in XML-Schema-Dokumenten werden gesammelt und an das Ende des einschließenden Objekts verschoben. In solchen Fällen wird daher empfohlen, Annotationen anstelle von Kommentaren zu verwenden.

Konfigurieren der Content Model-Ansicht

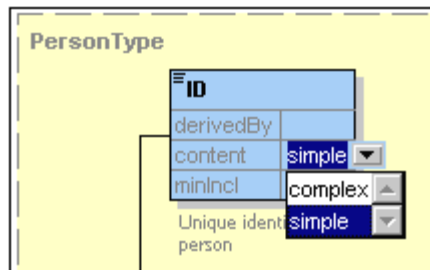
Sie können die Content Model-Ansicht für das gesamte Schema im Dialogfeld "Schema-Ansicht konfigurieren" (**Schema-Design | Ansicht konfigurieren**) konfigurieren. Nähere Informationen zu Konfigurationsoptionen finden Sie in der Benutzerreferenz im Abschnitt [Ansicht konfigurieren](#). Beachten Sie, dass die hier definierten Einstellungen für das gesamte Schema, die Erzeugung der Schema-Dokumentation und die Druckausgabe gelten.

Ändern von Komponenteneigenschaften direkt im Content Model

Wenn die Content Model-Ansicht so konfiguriert ist, dass Komponenten mit zusätzlichen beschreibenden Eigenschaftszeilen im Komponentenkästchen angezeigt werden, können Sie diese Informationen bearbeiten und dadurch die Eigenschaften der Komponente ändern. Sie können diese Eigenschaftszeilen durch Klicken auf das Symbolleistensymbol "Vordefinierte

Details hinzufügen"  ein- und ausblenden. Sie können also zwischen einer Ansicht, die die definierten Eigenschaften enthält und einer Ansicht, die diese nicht enthält hin- und herschalten. So bearbeiten Sie Komponenteneigenschaften:

- Doppelklicken Sie auf das gewünschte Informationsfeld der jeweiligen Komponente und beginnen Sie mit der Eingabe oder Bearbeitung der Daten. Wenn eine vordefinierte Option zur Auswahl steht, erscheint eine Dropdown-Liste, aus der Sie den gewünschten Eintrag auswählen können. Geben Sie andernfalls den gewünschten Wert einfach ein.



- Bestätigen Sie durch Drücken der **Eingabetaste**. Die Details-Eingabehilfe wird aktualisiert, um Ihre Änderungen anzuzeigen.

Alternativ dazu können Sie die Eigenschaften einer Komponente in der Details-Eingabehilfe ändern. Die Änderungen werden in den Platzhalterfeldern übernommen - falls diese Anzeige aktiviert ist.

Dokumentieren des Content Models

Sie können [detaillierte Dokumentation](#) zu Ihrem Schema im HTML- und MS Word-Format erstellen. Für jede globale Komponente wird detaillierte Dokumentation generiert. Die Liste der globalen Komponenten wird im Inhaltsverzeichnis angezeigt, über das Sie Verknüpfungen zu den Content Models der einzelnen Komponenten herstellen können. Außerdem werden verwandte Elemente (wie z.B. Child-Elemente, complexTypes usw.) durch Hyperlinks referenziert, sodass Sie von einem Element zum anderen navigieren können. Um Schema-Dokumentation zu generieren, wählen Sie den Menübefehl **Schema-Design | Schema-Dokumentation generieren**.

Attribute und Identity Constraints

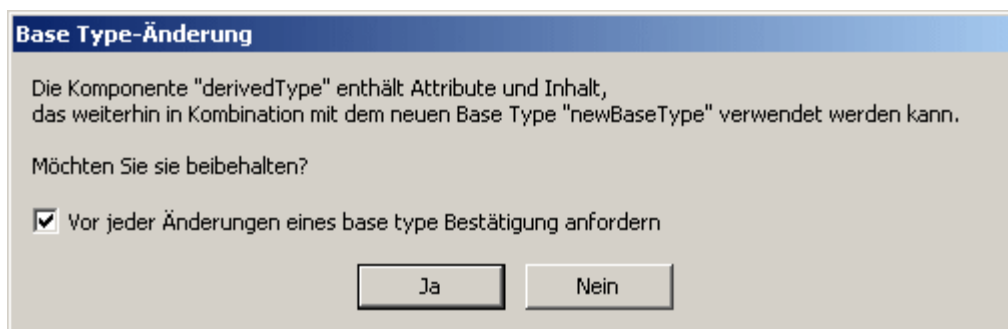
Attribute und Identity Constraints können in einem Fenster unterhalb der Content Model-Ansicht oder in Kästchen im Content Model selbst angezeigt werden. Durch Klicken auf die Symbole  können Sie zwischen diesen beiden Ansichten wechseln. Eine Beschreibung zum Einfügen und Bearbeiten von Attributen und Identity Constraints finden Sie unter [Definieren von Attributen für eine Komponente](#) und [Definieren von Identity Constraints für eine Komponente](#).

4.3.3 Modifizieren des Base Types

Wenn der Base Type eines abgeleiteten Types in der Schema-Ansicht geändert wird, können Inhalt, Attribute, Facets und Beispielwerte, die innerhalb des abgeleiteten Types 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.

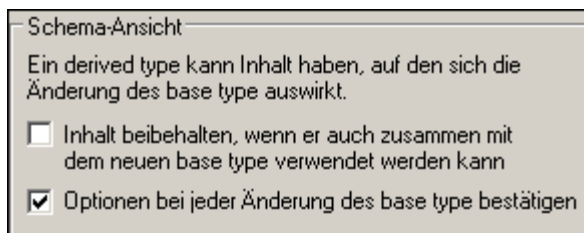
Wenn der Base Type eines abgeleiteten Types, der noch Inhalt, Attribute, Facets oder Beispielwerte enthält, geändert wird, erscheint das Dialogfeld "Base Type-Änderung".



Wenn das Kontrollkästchen der Bestätigungsaufforderung deaktiviert wird, erscheint eine Meldung (siehe Abbildung unten) mit dem Hinweis, dass die Bestätigung auf dem Register "Ansicht" im Dialogfeld "Optionen" ([Extras | Optionen | Ansicht](#)) wieder eingeschaltet werden kann.



Im Bereich "Schema-Ansicht" (siehe Abbildung unten) auf dem Register "Ansicht" im Dialogfeld "Optionen" ([Extras | Optionen | Ansicht](#)) können Sie festlegen, ob Inhalte beibehalten sollen und ob für jede Modifikation des Base Types 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.

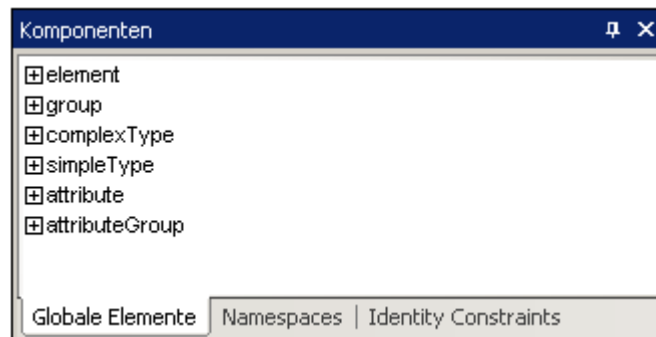
4.3.4 Eingabehilfen in der Schema-Ansicht

Es gibt drei Eingabehilfen in der Schema-Ansicht: den Komponenten-Navigator, die Details-Eingabehilfe und die Facets-Eingabehilfe. Im Folgenden sind diese näher beschrieben.

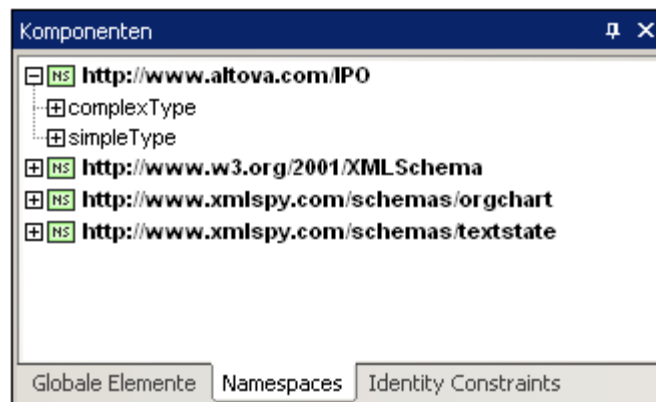
Komponenten-Navigator

Der Komponenten-Navigator ist eine Eingabehilfe in der **Schema-Ansicht**. Er 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 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 im Komponenten-Navigator angezeigt.
- Ihnen eine Übersicht über die im Schemadokument definierte Identity Constraints zu verschaffen. Eine Beschreibung von Identity Constraints finden Sie im Abschnitt [Identity Constraints in der Schema-Übersicht](#).



Auf dem Register "Globale Elemente" (*siehe oben*) werden die globalen Komponenten nach ihrem Komponententyp in einer Baumstruktur gegliedert. Auf dem Register Namespace (*siehe unten*) sind die Komponenten zuerst nach Namespace und anschließend nach Komponententyp geordnet. Beachten Sie, dass ein Komponententyp nur dann in einer Baumstruktur angezeigt wird, wenn mindestens eine Komponente dieses Typs im Schema vorhanden ist.

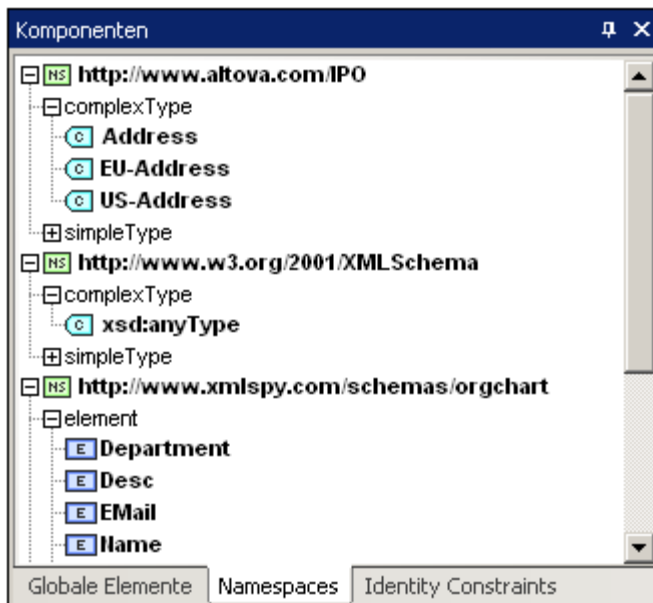


In der Baumstrukturanzeige werden globale Komponenten in die folgenden sechs Gruppen eingeordnet:

- Element-Deklarationen (Elemente)
- Model Groups (Groups)
- ComplexTypes
- SimpleTypes
- Attribut-Deklarationen (Attribute)
- Attribut-Gruppen

Wenn Sie eine Komponenten-Typ-Gruppe in der Baumstruktur erweitern, werden alle Komponenten in dieser Gruppe angezeigt (*siehe Abbildung unten*). Auf diese Art können Sie

schnell zur gewünschten Komponente navigieren.



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 oder eine Attribut-Gruppe handelt), so wird die Komponente in der Schema-Übersicht (im Hauptfenster) markiert.

Bitte beachten Sie: Wenn es sich bei der Komponente um ein inkludiertes oder importiertes Schema handelt, 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.

Details-Eingabehilfe

Die Details-Eingabehilfe steht in der **Schema-Ansicht** zur Verfügung. Sie enthält editierbare Informationen über den im Hauptfenster ausgewählten Kompositor/die ausgewählte Komponente. Wenn Sie eine Schemadatei, die Datenbank-Extensions enthält, bearbeiten, wird unter Umständen ein zusätzliches Register mit Informationen über die DB Extensions angezeigt.

The 'Details' dialog box shows the configuration for a SimpleType element named 'PhoneExt'. The 'type' field is set to 'xsd:integer'. Other fields include 'isRef' (unchecked), 'minOcc' (1), 'maxOcc' (1), 'content' (simple), 'derivedBy', 'default', 'fixed', 'nillable', 'block', 'form', and 'id'.

name	PhoneExt
isRef	☐
minOcc	1
maxOcc	1
type	xsd:integer
content	simple
derivedBy	
default	
fixed	
nillable	
block	
form	
id	

Details SimpleType

Um die Eigenschaften des derzeit ausgewählten Kompositors oder der ausgewählten Komponente 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 Dropdown-Liste zur Verfügung steht, wählen sie den gewünschten Wert aus; dieser Wert wird in dem Feld übernommen.

Änderungen, die Sie über die Details-Eingabehilfe vornehmen, werden sofort im Content Model-Diagramm angezeigt.

Facets-Eingabehilfe

Die Facets-Eingabehilfe steht in der **Schema-Ansicht** zur Verfügung und gestattet die Eingabe der Werte von Facets, Patterns, Beispielwerten und Enumerations.

The 'Facets' dialog box shows a list of values: 1000, 900, 500, 200, 100, and 0. The 'Facets' tab is selected.

1000
900
500
200
100
0

Facets Patterns Enumerations Samples

Um Facets, Patterns, Beispielwerte oder Enumerations in der Facets-Eingabehilfe zu ändern, gehen Sie vor wie folgt:

1. Wählen Sie das entsprechende Register aus (Facets, Patterns, Beispielwerte oder Enumerations)
2. Wählen Sie einen Wert aus der Dropdown-Liste aus, falls vorhanden. Doppelklicken Sie andernfalls auf eine Zeile und bearbeiten Sie den Text manuell bzw. geben Sie ihn manuell ein.

Anmerkung:

- Sie können die Patterns und Enumerations einer Komponente mit Hilfe der Tastenkürzel zum Ausschneiden, Kopieren und Einfügen (Strg+X, Strg+C bzw. Strg+V) in eine andere Komponente kopieren. Wählen sie in der Facets-Eingabehilfe ein oder mehrere Patterns oder Enumerations aus, um diese auszuschneiden oder zu kopieren, klicken Sie anschließend in die Facets-Eingabehilfe der Zielkomponente und fügen Sie sie ein.
- Beispielwerte können beim Generieren einer XML-Datei anhand des Schemas verwendet werden (mit dem Menübefehl **DTD/Schema | XML-Beispieldatei generieren**). Wenn ein Beispielwert ungültig ist, wird er mit einer Warnung markiert und der Beispielwert wird auf dem Register "Beispiele" orange angezeigt. Sie können die Symbolleisten-Schaltfläche "Validierungswarnungen"  auch deaktivieren. Beachten Sie außerdem Folgendes: Die XSD-Datei selbst ist bei ungültigen Beispielen nicht betroffen. Sie bleibt gültig, wenn die Datei in anderer Hinsicht gültig ist.

4.3.5 Identity Constraints

Identity Constraints können für eine globale Komponente durch zwei Ansätze definiert werden:

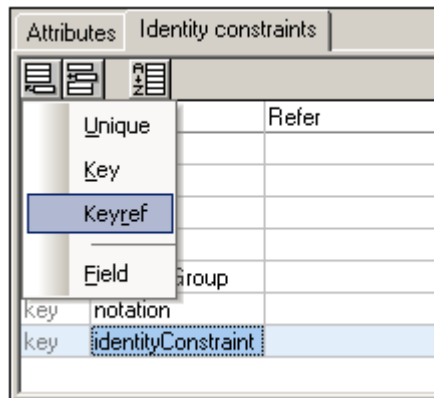
- In der [Schemaübersicht](#): Wählen Sie eine globale Komponente aus und definieren Sie auf dem Register "Identity Constraints" (am unteren Rand der Ansicht und unterhalb des Hauptfensters) Constraints.
- In der [Content Model-Ansicht](#) einer globalen Komponente. In der Content Model-Ansicht sehen Sie - anders als bei Bearbeitung von Identity Constraints in der Schemaübersicht - eine grafische Darstellung der Identity Constraints und es steht Ihnen eine Drag-and-Drop-Funktion zur Verfügung.

In diesem Abschnitt werden diese beiden Methoden zum Erstellen und Bearbeiten von Identity Constraints beschrieben. Auf dem Register "Identity Constraints" der Eingabehilfe "Komponenten" finden Sie eine [Übersicht über alle Identity Constraints](#) im Schema. Eine nähere Beschreibung dazu finden Sie [weiter unten](#).

Identity Constraints in der Schema-Übersicht

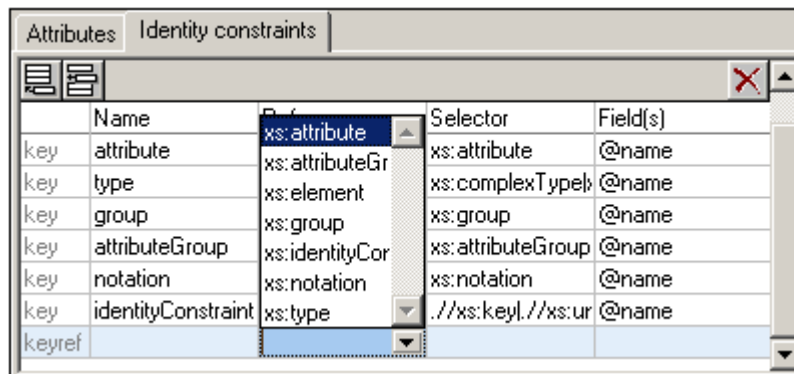
Um in der Schema-Übersicht Identity Constraints für eine globale Komponente zu definieren, gehen Sie folgendermaßen vor:

1. Wählen Sie die globale Komponente in der [Schemaübersicht](#) in der Liste der globalen Komponenten aus.
2. Wählen Sie im Attribute-/Identity Constraints-Fenster das Register "Identity Constraints" aus.
3. Klicken Sie auf das Symbol "Anhängen"  oder "Einfügen"  in der linken oberen Ecke des Registers.
4. Wählen Sie im Popup-Fenster (*Abbildung unten*) die Art des gewünschten Identity Constraint aus: *Unique*, *Key* oder *Keyref*.



In der Liste der Identity Constraints wird ein Eintrag erstellt.

- Geben Sie in diesen Eintrag die Eigenschaften des Identity Constraint ein. Geben Sie dem Identity Constraint einen Namen (siehe Abbildung unten) und definieren Sie anschließend für die Elemente `selector` und `field` XPath-Ausdrücke. Für jeden Schritt in diesen XPath-Ausdrücken erscheint ein Popup-Fenster mit den verfügbaren Nodes (siehe Abbildung unten). Für das `refer`-Element der Identity Constraint-Art `keyref` zeigt eine Dropdown-Liste die verfügbaren `key`-Elemente an (siehe Abbildung unten).



Wenn Sie zu einem Identity Constraint mehrere Feldelemente definieren möchten, wählen Sie den Identity Constraint aus, klicken Sie auf das **Anhängen** oder **Einfügen**-Symbol (siehe Schritt 3 oben) und wählen Sie **Field** aus. Ein leerer Field-Eintrag wird zum ausgewählten Identity Constraint hinzugefügt. Geben Sie in das Field-Eingabefeld den gewünschten XPath-Ausdruck ein.

Anmerkung: Jedem dieser Elemente (also dem Identity Constraint, seinem Selektor und/oder Feld(ern)) kann eine ID zugewiesen werden. Wählen Sie dazu das Element aus und geben Sie in der Eingabehilfe "Details" die ID in die Zeile `id` ein.

Identity Constraints in der Content Model-Ansicht

Um Identity Constraints in der [Content Model-Ansicht](#) anzuzeigen, gehen Sie folgendermaßen vor:

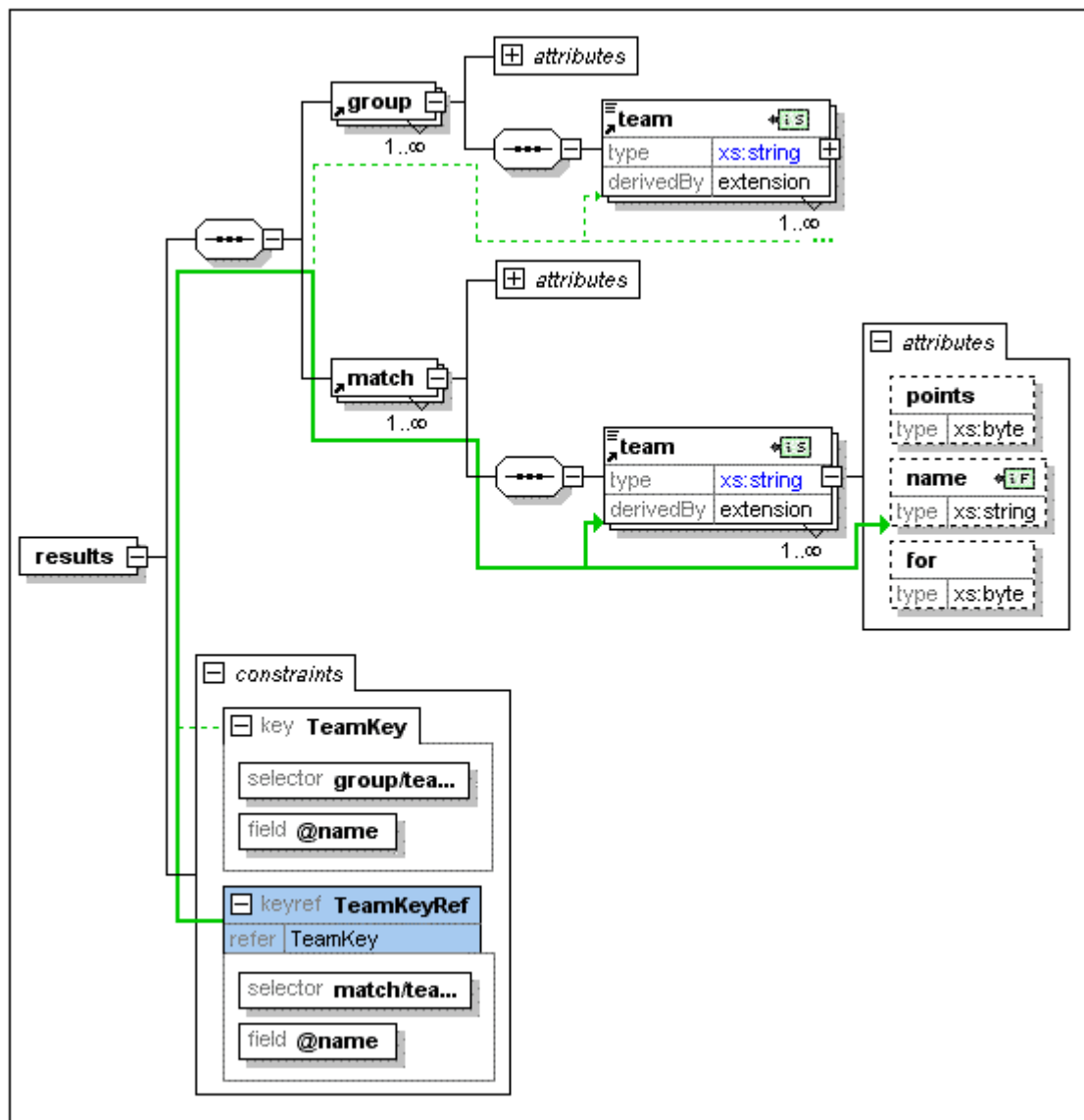
- Wählen Sie in der Schemaübersicht die globale Komponente aus, für die Sie den Identity Constraint erstellen möchten und wechseln Sie in die [Content Model-Ansicht](#) dieser Komponente.
- Stellen Sie sicher, dass die Anzeige von Identity Constraints im Dialogfeld "Schema-Ansicht konfigurieren" (**Schema-Design | Ansicht konfigurieren**) aktiviert ist. Damit können Sie den Kasten Identity Constraints im Design ein- und ausblenden (

siehe Abbildung unten). Sie können diesen Kasten auch über die Schema-Design-Symbolleistenschaltfläche "Constraints anzeigen"  ein- und ausblenden.

3. Stellen Sie sicher, dass das Symbol "Identity Constraints visualisieren"  in der Symbolleiste "Schema-Design" aktiviert ist. Dadurch wird gewährleistet, dass Beziehungslinien und ID Constraint-Informationen in der Content Model-Ansicht angezeigt werden (*siehe Abbildung unten*).

Anmerkung: Über die Schaltfläche "Identity Constraints visualisieren"  werden in der Schema-Ansicht die Funktionen zum Bearbeiten und Validieren von ID Constraints aktiviert. Wenn ein Node mit einem XPath-Ausdruck in einem ID Constraint nicht gefunden wird, wird unter Umständen eine Fehlermeldung oder eine Warnmeldung angezeigt, um Sie auf den fehlerhaften XPath-Ausdruck hinzuweisen; das XML-Schema-Dokument selbst ist aufgrund des fehlerhaften XPath-Ausdrucks nicht ungültig. Um die Gültigkeit des Dokuments ohne den XPath-Ausdruck erneut zu überprüfen, wechseln Sie in die Text- oder die Grid-Ansicht und validieren Sie das Dokument. Sie können die Validierung in der Schema-Ansicht auch deaktivieren. Deaktivieren Sie dazu die Schaltfläche "Identity Constraints visualisieren" .

In der Abbildung unten sehen Sie die grafische Anzeige von Identity Constraints. Im Identity Constraints-Kasten sehen Sie zwei Identity Constraints: `TeamKey` und `TeamKeyRef`. Die Eigenschaften von `TeamKeyRef` sind mit einer durchgezogenen grünen Linie markiert (da `TeamKeyRef` ausgewählt ist). Die Eigenschaften von `TeamKey` (nicht ausgewählt) werden durch eine gepunktete grüne Linie angezeigt. Für die einzelnen Identity Constraints wird der durch den XPath Ausdruck für `selector` und `field` ausgewählte Node durch das Symbol  bzw.  angezeigt. Bei reduzierten Nodes (z.B. hier beim Element `field` des `TeamKey` Constraint) endet die Beziehungslinie in einem Auslassungszeichen (*siehe Abbildung unten*).



Um in der [Content Model-Ansicht](#) einen Identity Constraint zu erstellen, gehen Sie folgendermaßen vor:

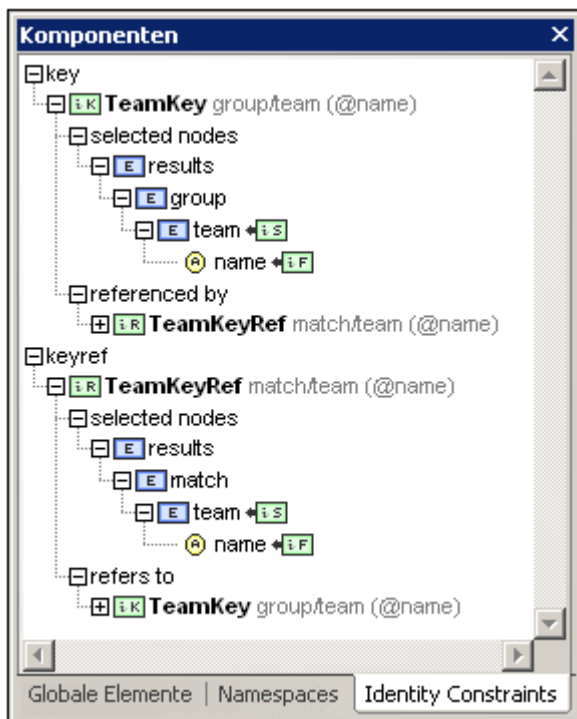
1. Klicken Sie im Kasten "Constraints" mit der rechten Maustaste auf den Eintrag "Constraints" oder auf den Namen eines vorhandenen Identity Constraint.
2. Wählen Sie **Einfügen** oder **Anhängen** und wählen Sie anschließend die Art von Identity Constraint aus, die Sie (oben in der Liste der vorhandenen Identity Constraints) einfügen oder (unten in der Liste der vorhandenen Constraints) anhängen möchten. Der Identity Constraint wird mit einem leeren `selector` Element und einem leeren `field` Element erstellt.
3. Der XPath-Ausdruck kann auf zwei Arten in die Elementkästen `selector` und `field` eingegeben werden: (i) entweder durch Eingabe oder (ii) durch Ziehen des Target Node in den Elementkasten `selector` oder `field` und Loslassen der Maustaste, wenn der Kasten markiert erscheint; der XPath-Ausdruck wird automatisch erstellt.
4. Um ein zusätzliches `field` Element zu einem Identity Constraint hinzuzufügen, klicken Sie entweder mit der rechten Maustaste auf den Namen des Constraint und wählen Sie im Kontextmenü den Befehl **Child hinzufügen | Feld** aus oder klicken Sie mit der rechten Maustaste auf das `selector` oder `field` Element und wählen Sie im

- Kontextmenü den Befehl **Einfügen | Feld** oder **Anhängen | Feld**.
- Um ein ID-Attribut zu einem Element in einem Constraint einzugeben, wählen Sie das Element aus und geben Sie in der Eingabehilfe "Details" in die Zeile `id` einen Wert ein.

Anmerkung: Identity Constraint-Informationen können über die Schaltfläche "ID Constraints-Informationen aktivieren"  der Symbolleiste "Schema-Design" ein- und ausgeblendet werden. Der gesamte Constraints-Kasten kann über das Dialogfeld "Schema-Anzeige konfigurieren" (**Schema-Design | Ansicht konfigurieren**) oder über die Symbolleiste "Schema-Design" (Schaltfläche "Constraints anzeigen" ) ein- und ausgeblendet werden.

Identity Constraints-Übersicht

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 erweiterbaren und reduzierbaren Struktur angezeigt.



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](#) 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](#) aus.
- Selektorziel/Feldziel in Diagramm anzeigen:** Wählt in der [Content Model-Ansicht](#) die Schemakomponente aus, die das Ziel 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 Identity Constraint in der [Schemaübersicht](#) aus.
- *Alle erweitern/reduzieren:* Erweitert bzw. reduziert die Struktur.

4.3.6 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.

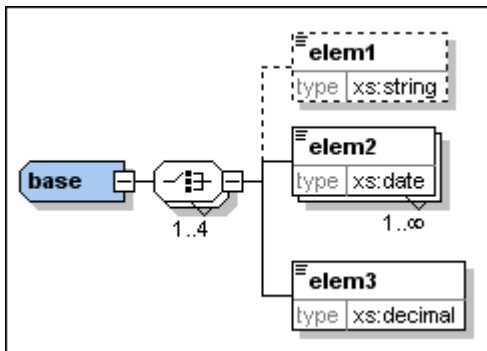
Mit Hilfe von Smart Restrictions werden die beiden Content Models in der grafischen Ansicht des abgeleiteten Content Model kombiniert und in Zusammenhang gesetzt. Im abgeleiteten complexType sehen Sie alle Partikel des Basis-complexType und in welcher Beziehung sie zum abgeleiteten Typ stehen. Außerdem 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" .

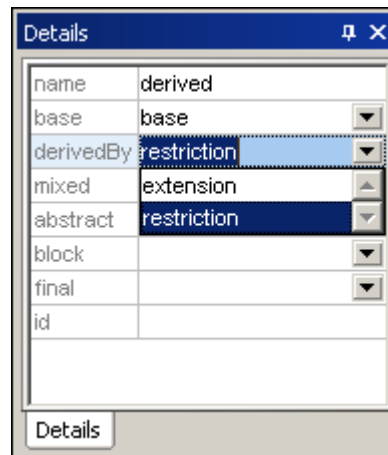
Im folgenden Beispiel sehen Sie, wie sich Smart Restrictions auswirken.

Der folgende complexType ist der Basis-Typ, der in diesem Beispiel verwendet wird:

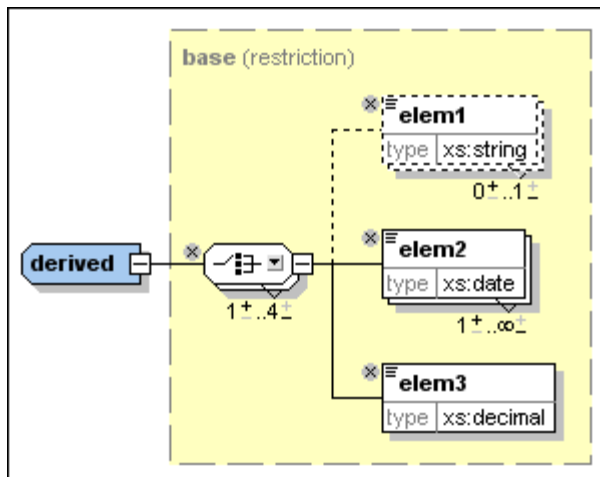


Der complexType "derived" wird folgendermaßen vom "Base" Type abgeleitet:

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**.

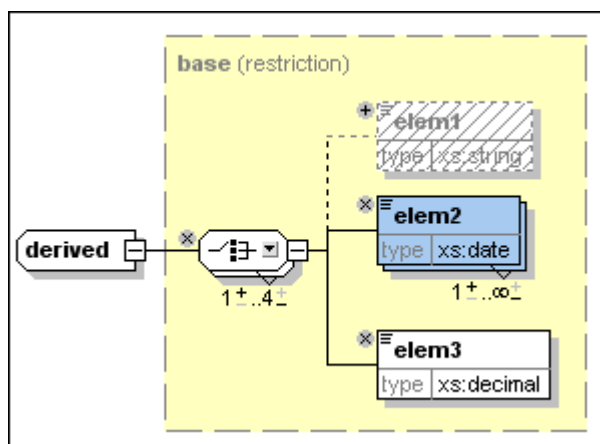


Wenn die Option "Smart Restrictions" aktiviert ist, so sieht der neue abgeleitete Typ folgendermaßen aus:

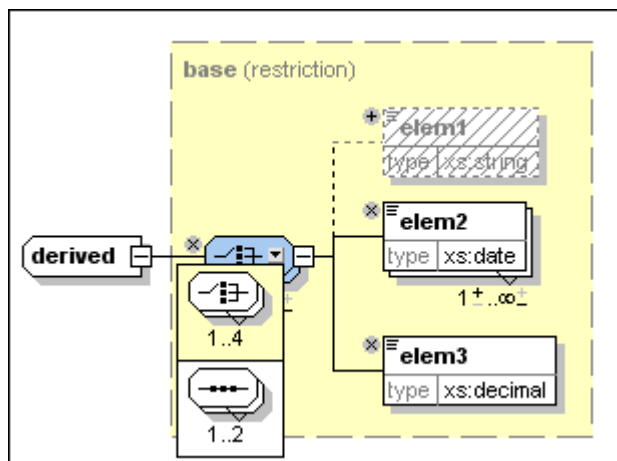


Beachten Sie die folgenden Steuerelemente, mit denen Sie den abgeleiteten Typ in diesem Beispiel durch Restriction einschränken können:

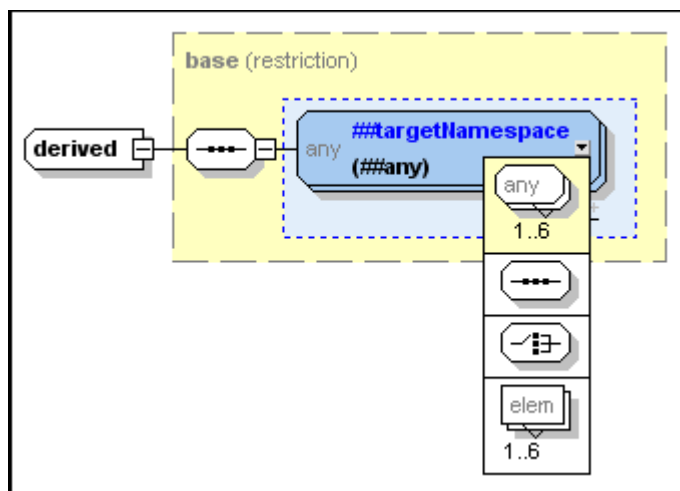
- Mit Hilfe des Symbols  können Sie Elemente aus dem abgeleiteten Typ entfernen, die sich im Basis-Typ befinden. Hier wurde elem1 gelöscht. Um es erneut hinzuzufügen, klicken Sie auf dieses Symbol .



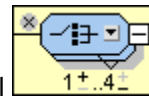
- Klicken Sie auf den Abwärtspfeil ▾ neben dem "Choice"-Kompositor, um die folgende Liste aufzurufen, über die Sie die Model Group "Choice" in eine Model Group "Sequence" ändern können:



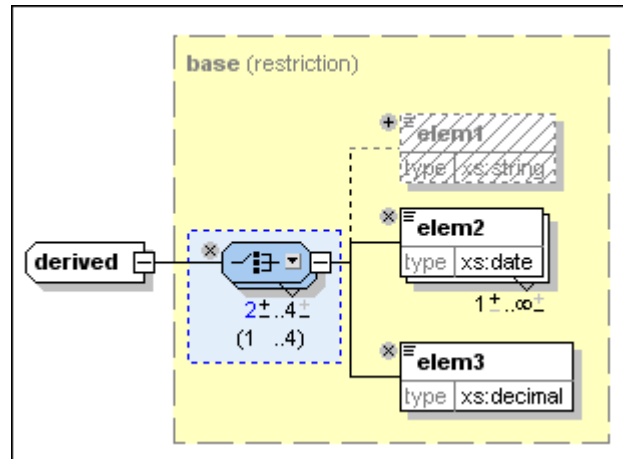
Sie können auch Wildcards auf dieselbe Art ändern - siehe Beispiel:



Ein ausführliche Liste darüber, welche Partikel durch welche anderen Partikel ersetzt werden können, finden Sie in der [XML-Schema-Spezifikation](#).

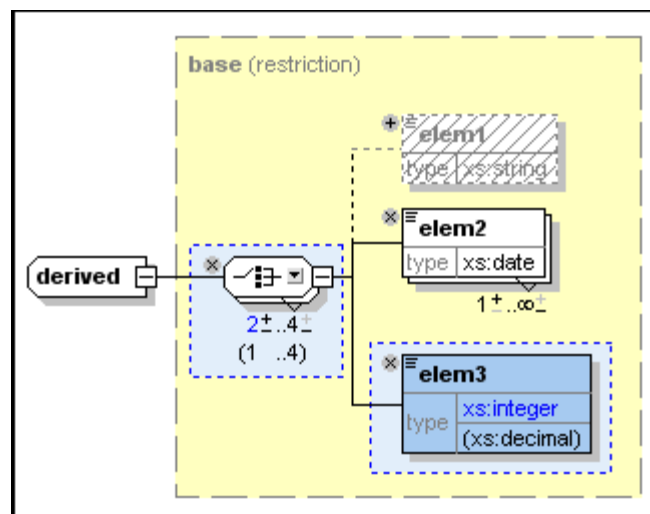


- Klicken Sie auf das folgende Symbol, um die Anzahl der "Occurrences" der Model Group zu ändern. Durch Klicken auf das +-Zeichen über der 1 können Sie die Mindestanzahl der Occurrences erhöhen, durch Klicken auf das Minus-Zeichen unterhalb von "4" können Sie die Maximalanzahl der Occurrences verringern. 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.

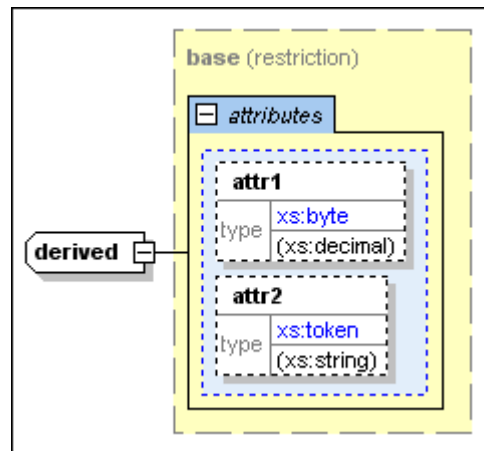


Hier sehen Sie, dass die Minimum Occurrence für dieses Element in 2 geändert wurde. Beachten Sie, dass die Model Group nun einen blauen Hintergrund aufweist, was bedeutet, dass sie nicht mehr mit der Model Group im Base complexType identisch ist. Außerdem wird der Occurrence-Bereich der Model Group im Basiselement nun in Klammern angezeigt.

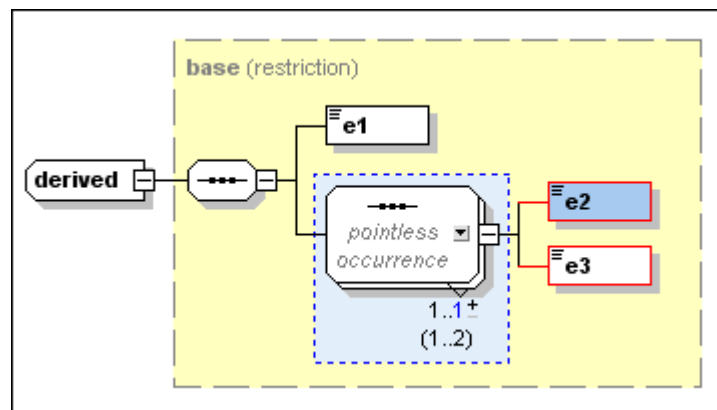
- Sie können die Datentypen von Attributen oder Elementen ändern, wenn es sich beim neuen Datentyp um eine gültige Restriction des in der [XML Schema-Spezifikation](#) definierten Basis-Typs handelt. So können Sie z.B. den Datentyp von elem3 im Datentyp "derived" von Decimal in Integer ändern. Das Element 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.



- 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":



Bitte beachten Sie: 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](#).

4.3.7 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

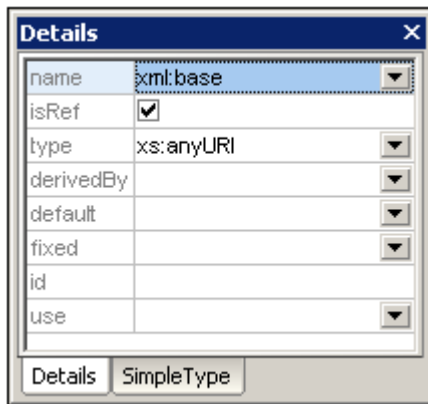
Erstellen Sie in der Schemaübersicht eine globale Import-Deklaration für den XML Namespaces Namespace. Klicken Sie dazu im oberen Bereich des Schemaübersichtsfensters auf die

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.



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.

4.3.8 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 angesehenen 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.

4.4 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 **Datei | XML-Arbeitsdatei zuweisen** können Sie jederzeit ein SPS zuweisen.

Dieser Abschnitt enthält:

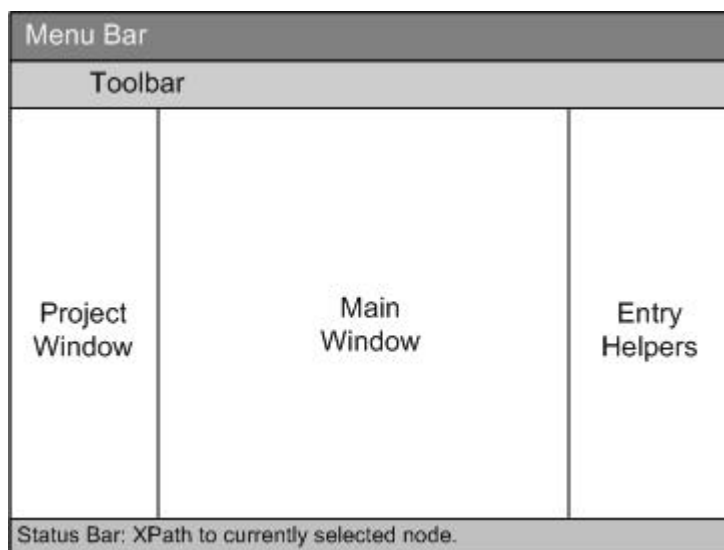
- eine Übersicht über die Benutzeroberfläche
- eine Beschreibung der Symbolleistensymbole der Authentic-Ansicht
- eine Beschreibung der Ansichtsmodi im Authentic-Ansicht-Hauptfenster
- eine Beschreibung der Eingabehilfen und deren Verwendung
- eine Beschreibung der verschiedenen in der Authentic-Ansicht 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.

4.4.1 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).



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.

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.

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](#).

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](#).

Statusleiste

In der Statusleiste wird der XPath zum aktuell ausgewählten Node angezeigt.

Kontextmenüs

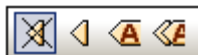
Dies sind Menüs, die angezeigt werden, wenn Sie im Hauptfenster auf die rechte Maustaste klicken. Es stehen Ihnen kontextsensitive Editierbefehle zur Verfügung, mit Hilfe derer Sie die Struktur und den Inhalt des ausgewählten Nodes bearbeiten können. Diese Befehle gestatten das Einfügen, Anhängen oder Löschen eines Nodes, das Hinzufügen von Entities und das Ausschneiden und Einfügen von Inhalt.

4.4.2 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.



Gemischte Markup-Symbole einblenden. Im StyleVision Power Stylesheet können Sie für jedes XML-Element oder Attribut einzeln festlegen, wie es im "Gemischten Markup"-Modus 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 einfügen



Zeile kopieren (d.h. der Inhalt einer Zelle wird dupliziert)



Zeile nach oben



Zeile nach unten



Zeile löschen

Bitte beachten Sie: 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](#) 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](#) 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](#).

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 Authentic-Ansicht Symbole 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 in der DB; 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.

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 HTML und RTF aktiv, während diejenigen für die PDF- und DocX (Word 2007+)-Ausgabe deaktiviert sind.

4.4.3 Authentic-Ansicht Hauptfenster

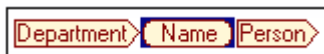
In der Authentic-Ansicht gibt es vier Ansichtsmodi: Große Markup-Symbole; Kleine Markup-Symbole; Gemischte Markup-Symbole; und Alle Markup-Symbole ausblenden. Damit können Sie die Markup-Informationen für Dokumente mit unterschiedlicher Genauigkeit anzeigen. Um zwischen diesen Ansichtsmodi zu wechseln, verwenden Sie die Befehle im Menü "Authentic" oder die Symbole der Symbolleiste (siehe vorhergehender Abschnitt, [Authentic-Ansicht Symbolleistenschaltflächen](#)).

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:

Nanonull, Inc.	
Location: US	
Street: 119 Oakstreet, Suite 4876 City: Vereno State & Zip: DC 29213	Phone: +1 (321) 555 5155 0 Fax: +1 (321) 555 5155 4 E-mail: office@nanonull.com
Vereno Office Summary: 4 departments, 15 employees. The company was established in Vereno in 1995 as a privately held software company. Since 1996, Nanonull has been actively involved in developing nanoelectronic software technologies. It released the first version of its acclaimed NanoSoft Development Suite in February 1999. Also in 1999, Nanonull increased its capital base with investment from a consortium of private investment firms. The company has been expanding rapidly ever since.	

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).

2006-04-01; **Boston**, **USA**

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.

Office Summary: 4 departments, 15 employees.

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.

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.

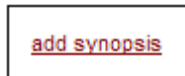


Bei den anderen Dateneingabe-Hilfen wird durch Ihre Auswahl der entsprechende XML-Wert erzeugt, der im StyleVision Power Stylesheet definiert wurde. In einer Auswahlliste könnte also beispielsweise "approved" (das in der Dropdown-Liste der Auswahlliste verfügbar wäre) ausgewählt werden, und dieser Eintrag 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

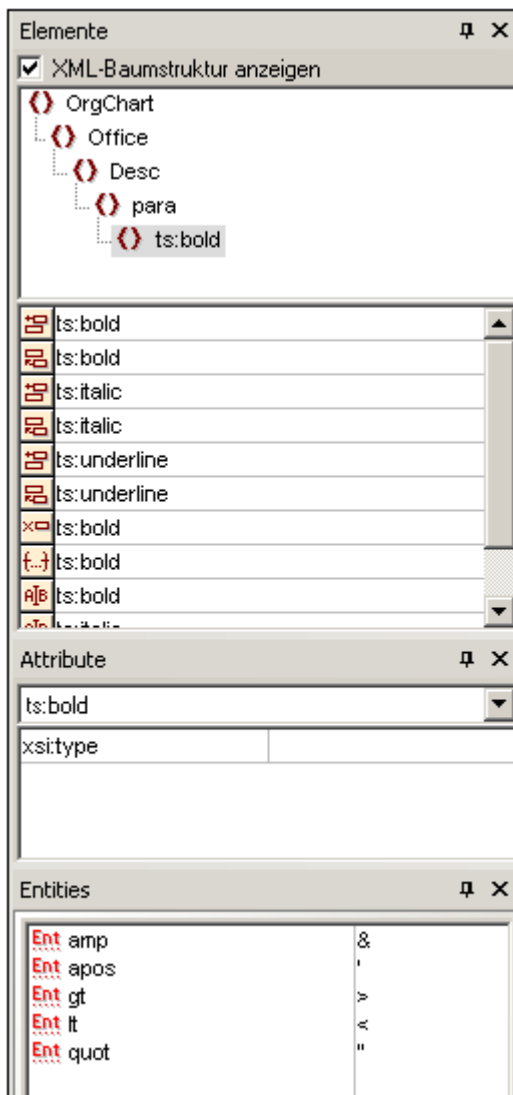
Wenn ein Element oder Attribut gemäß dem referenzierten Schema **optional** ist, wird eine Eingabeaufforderung vom Typ "add [Element/Attribut]" angezeigt.



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.

4.4.4 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:

- 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 das 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 Originalements 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 Originalements enthält, dann wird das instantiierte Child-Element des Originalements 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 Originalements 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 (wenn ein Bereich ausgewählt ist)

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.



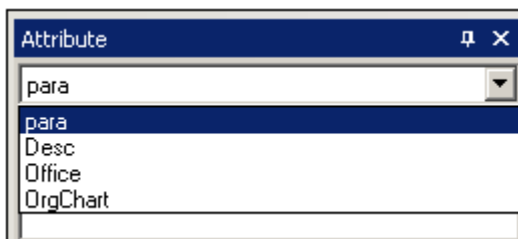
Element löschen (wenn ein Einfügepunkt ausgewählt ist)

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.

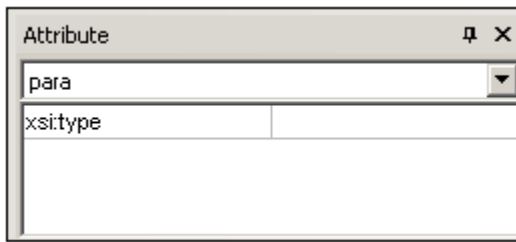
Attribut-Eingabehilfe

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`)



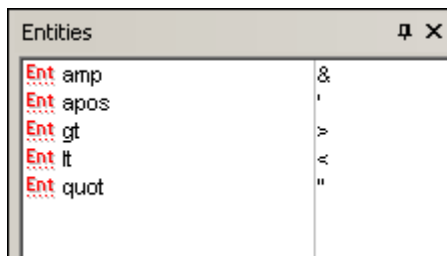
Um den Wert für ein Attribut einzugeben, klicken Sie in das Wertefeld des Attributs und geben Sie den Wert ein. Daraufhin wird das Attribut mit dem entsprechenden Wert im XML-Dokument erstellt.

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 Auswahl eines der aufgelisteten Werte aus der Dropdown-Liste geändert werden. Bei den 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](#).

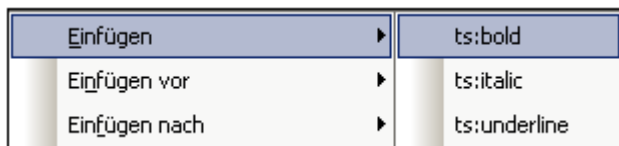
4.4.5 Authentic-Ansicht Kontextmenüs

Wenn Sie mit der rechten Maustaste auf einen ausgewählten Dokumenteninhalte oder Node klicken, wird ein Kontextmenü mit Befehlen, die für die Auswahl oder Cursorposition relevant

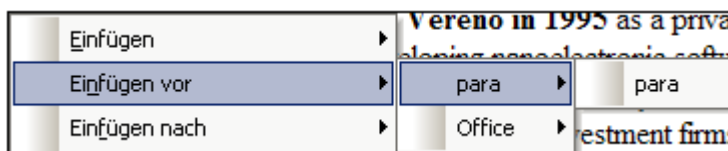
sind, angezeigt.

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](#) 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](#).

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 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 ausgewähltes Element in Ihre Auswahl im Hauptfenster übernommen. Nähere Informationen dazu finden Sie unter [Authentic-Ansicht Eingabehilfen](#).

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 sicherstellen, 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).

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.

4.5 Browser-Ansicht

Die Browser-Ansicht wird normalerweise zur Ansicht 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**.

Anmerkung zu Microsoft Internet Explorer und XSLT

Für die Browser-Ansicht wird Microsoft Internet Explorer 5.0 oder höher benötigt. 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. Nähere Informationen dazu erhalten Sie in unserem [Download Center](#). (Beachten Sie, dass die Unterstützung von XSL in IE 5 nicht zu 100 % kompatibel mit der offiziellen XSLT Recommendation ist. Wenn Sie daher in der Browser-Ansicht Probleme mit IE 5 haben, sollten Sie auf IE 6 upgraden.) Außerdem sollten Sie überprüfen, welche XSLT-Version Ihre Version des Internet Explorer unterstützt.

Funktionen der Browser-Ansicht

In der Browser-Ansicht stehen die folgenden Funktionen zur Verfügung. Sie können über das [Menü Browser](#), [Menü 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 des selben Dokuments anzeigen. Dieser Befehl befindet sich im Menü **Browser** und ist ein Ein/Aus-Befehl, mit dem Sie zwischen dem Browser-Fenster als separatem Fenster und als Registerkarte des Fensters wechseln können. Im Dialogfeld "Optionen" können Sie auf dem [Register "Ansicht"](#) festlegen, ob die Browser-Ansicht in einem separaten Fenster angezeigt werden soll.
- **Vorwärts und Zurück:** die allgemeinen Browser-Befehle zum Navigieren durch Seiten, die in der Browser-Ansicht gefunden wurden. Diese Befehle befinden sich im Menü **Browser**
- **Schriftgröße:** Kann über den **Browser** Menübefehl 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**.

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](#). In diesem Abschnitt werden einige wichtige XMLSpy-Einstellungen zur Erstellung von Dateien erläutert.
- XML-Dokumente können in der [Textansicht](#), der [Grid-Ansicht](#) und der [Authentic-Ansicht](#) 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.
- [Eingabehilfen](#) für XML-Dokument haben bestimmte Funktionen, die hier beschrieben werden.
- Wie [XML-Dokumente mit XSLT und XQuery verarbeitet werden](#). Es werden verschiedene XMLSpy Funktionen im Zusammenhang damit erläutert.
- Diverse [andere Funktionen](#) für die Arbeit mit XML-Dokumenten werden beschrieben.

Altova Website:  [XML Editor](#)

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.

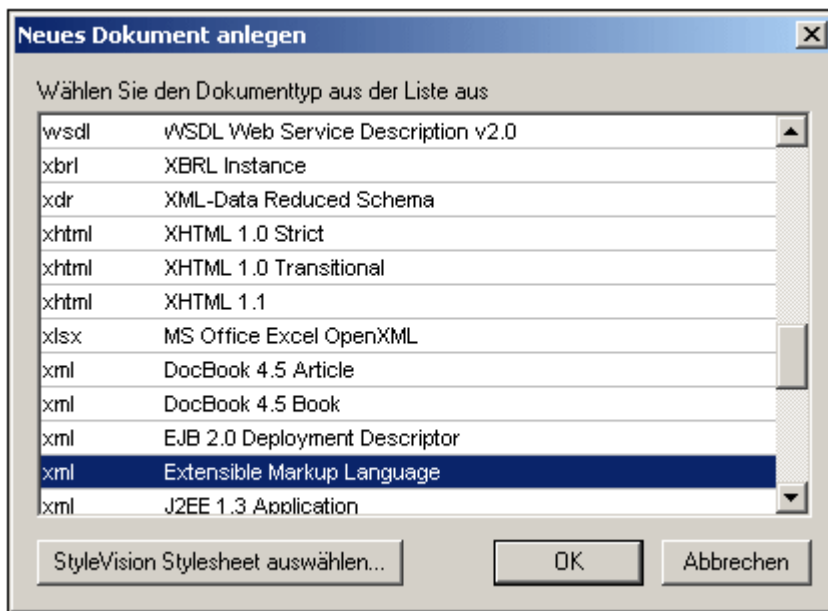
Wählen Sie im Dialogfeld "Optionen" auf dem Register **Dateiarten** die Dateart `.xml` aus und aktivieren Sie die gewünschte Bearbeitungsansicht (Text oder Grid). Beachten Sie dass: (i) die Schema-Ansicht nur für XML-Schemas verwendet werden können; und dass (ii) die Browser-Ansicht eine reine Ansicht ist, in der Sie keine Bearbeitungen vornehmen können.

Auf dem Register **Dateiarten** können Sie außerdem `<%PRODUCT%>` als Standardeditor für die ausgewählte Dateart 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" auf dem Register **Ansicht** die Option *...Dateien immer in der Authentic-Ansicht öffnen*. Wenn diese Option nicht aktiviert ist, wird die Datei in der Standardansicht geöffnet, die auf dem Register **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

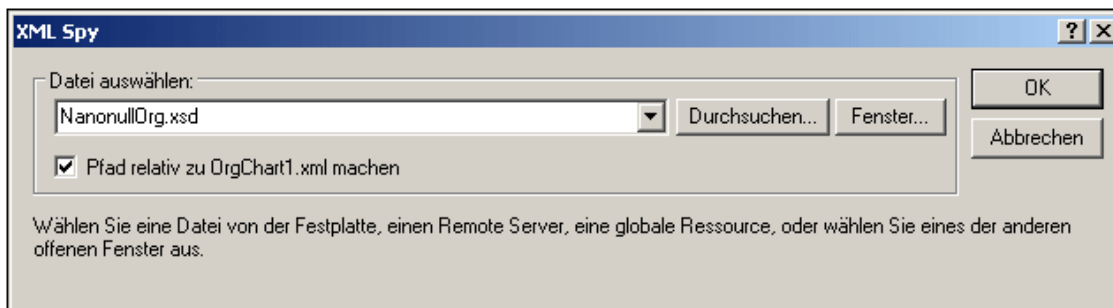
Wenn einem vorhandenen XML-Dokument ein Schema zugewiesen ist, kann es beim Öffnen und/oder Speichern validiert werden. Die Einstellung dafür wird im Dialogfeld "Optionen" auf dem Register **Datei** vorgenommen (**Extras | Optionen**).

Die automatischen Validierungseinstellungen auf dem Register **Datei** können mit einer Einstellung auf dem Register "Dateiarten" kombiniert werden, sodass die automatische Validierung für bestimmte Dateiarnten deaktiviert wird. Über die Einstellungen auf den beiden Registern können Sie die automatische Validierung für bestimmte Dateiarnten definieren.

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](#). 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](#).

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 geparte 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.

5.3 Bearbeiten von XML in der Textansicht

XMLSpy bietet zusätzlich zu den allgemein verfügbaren Bearbeitungsfunktionen in der Textansicht einige spezielle XML-Textbearbeitungsfunktionen, die unter [Textansicht](#) im Abschnitt "Bearbeitungsansichten" beschrieben sind.

- [Ein- und Auskommentieren von Text](#)
- [Versehen von XML-Zeichen mit Escape und Entfernen von Escape](#)
- [Einfügen von Dateipfaden](#)
- [Einfügen von XML-Fragmenten mittels XInclude](#)
- [Kopieren von XPath und XPointer-Ausdrücken in die Zwischenablage](#)

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.

```
<      &lt;
```



```
>      &gt;
&      &amp;
'      &apos;
"      &quot;
```

Beispiel:

`<a></a>` kann mit dem Befehl **XML-Zeichen mit Escape versehen** als
`<a></a>` dargestellt werden.
`<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](#) 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](#).

Einfügen von XML-Fragmenten mittels XInclude

Mit dem Befehl [Bearbeiten | XInclude einfügen](#) können Sie mittels XInclude den Inhalt eines ganzen XML-Dokuments oder ein Dokumentfragment in das bearbeitete Dokument einfügen. Mit diesem Befehl können Sie einen Dateipfad schnell und korrekt eingeben. Nähere Informationen dazu finden Sie unter der [Beschreibung zum Befehl](#).

Kopieren von XPath- und XPointer-Ausdrücken in die Zwischenablage

Der XPath- und der XPointer-Ausdruck des ausgewählten Node können mit dem Befehl [Bearbeiten | XPath kopieren](#) bzw. [Bearbeiten | XPointer kopieren](#) in die Zwischenablage eingefügt werden. Auf diese Art erhalten Sie den korrekten XPath- bzw. XPointer-Ausdruck für einen bestimmten Node. Die kopierten Ausdrücke können anschließend an der 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 "Benutzerreferenz".

5.4 Bearbeiten von XML in der Grid-Ansicht

In der [Grid-Ansicht](#) 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.

XML

Company

= xmlns	http://my-company.com/namespace
= xmlns:xsi	http://www.w3.org/2001/XMLSchema-instance
= xsi:schema...	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	Ⓢ Last
1	false	MA	true	Alfred	Aldrich
2	true	Ph.D	false	Colin	Coleman
3	true	BA	false	Fred	Smith

Beachten Sie in der Abbildung oben, dass das Dokument als Hierarchie in Form eines Rasters angezeigt wird. Wenn ein Node Inhalt enthält, ist er in zwei Felder aufgeteilt: eines für den Namen und eines für den Inhalt. Node-Namen werden fett und Node-Inhalt in normaler Schrift angezeigt.

Als Tabelle anzeigen

Wenn es mehrere Instanzen eines sich wiederholenden Elements gibt, wird in der Standard-Grid-Ansicht jede Instanz, eine nach der anderen, zur Gänze angezeigt, wobei in der Dokumentreihenfolge von oben nach unten vorgegangen wird (*Abbildung unten*).

Person		
First		Fred
Last		Landis
Title		Program Manager
PhoneExt		951
E-Mail		f.landis@nanonull.com
Shares		2000
LeaveTotal		28
LeaveUsed		10
LeaveLeft		18

Person		
First		Michelle
Last		Butler
Title		Software Engineer
PhoneExt		654
E-Mail		m.butler@nanonull.com
Shares		1500
LeaveTotal		19
LeaveUsed		9
LeaveLeft		10

Eine solche Struktur bestehend aus mehreren Instanzen kann auch als Tabelle angezeigt werden (*Abbildung unten*), in der die Child Nodes die Spalten bilden und die Instanzen die Zeilen.

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. Dadurch werden Operationen wie das Sortieren von Daten nach dem Wert eines gemeinsamen Child-Node möglich. So können etwa in der Abbildung oben die sechs `Person` Elemente mittels einer einfachen Aktion auf der Benutzeroberfläche nach ihrem Child-Element `Last` sortiert werden. Eine solche Operation ist viel einfacher, als eine XSLT-Transformationen durchzuführen. Dies wäre die übliche Methode, um ein XML-Nodeset zu sortieren.

Bearbeiten der Dokumentstruktur

In der Grid-Ansicht kann das XML-Dokument grafisch bearbeitet werden. So können Sie z.B. einzelne Segmente der Dokumentstruktur erweitern und reduzieren, Nodes einfügen, anhängen und löschen, Nodes mit der Maus an andere Stellen verschieben und einen Dokumenttyp in einen anderen konvertieren.

Das Menü **XML** enthält Befehle zum Einfügen, Anhängen und Hinzufügen von leeren Child Nodes. So können Sie z.B. einen leeren Child Node hinzufügen, indem Sie ein Element auswählen und dann ein leeres Child-Element hinzufügen. Anschließend können Sie einen

Namen und Inhalt für den neu hinzugefügten Node definieren, indem Sie in das entsprechende Feld (Name oder Content) doppelklicken und den gewünschten String eingeben.

Mit Hilfe der Element- und Attributeingabehilfen können Sie Child Nodes einfügen, anhängen und hinzufügen, die an der ausgewählten Stelle zulässig sind. So können Sie z.B. einen Node im Hauptfenster auswählen. Die Element- und Attribut-Nodes, die an dieser Stelle gültig eingefügt, angehängt oder hinzugefügt werden können, werden in der Element- und der Attributeingabehilfe aufgelistet.

Die Befehle, die im Menü "XML" zur Verfügung stehen, und die Eingabehilfen, die auf einen ausgewählten Node angewendet werden können, stehen auch im Kontextmenü dieses Node zur Auswahl.

Bearbeiten von Inhalt

Um Inhalt zu bearbeiten, doppelklicken Sie in das Inhaltsfeld und geben Sie den Text des Inhalts ein. Entities können über die Eingabehilfe "Entities" eingefügt werden.

Mehr über die Grid-Ansicht

Eine ausführlichere Beschreibung der Grid-Ansicht finden Sie unter [Bearbeitungsansichten](#).

5.5 Bearbeiten von 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.

Location: US	
Street: 119 Oakstreet, Suite 4876	Phone: +1 (321) 555 5155 0
City: Vereno	Fax: +1 (321) 555 5155 4
State & Zip: DC 29213	E-mail: office@nanonull.com

Vereno Office Summary: 4 departments, 15 employees.

The company was established **in Vereno in 1995** as a privately held software company. Since 1996, Nanonull has been actively involved in developing nanoelectronic software technologies. It released the first version of its acclaimed *NanoSoft Development Suite* in February 1999. Also in 1999, Nanonull increased its capital base with investment from a consortium of private investment firms. The company has been expanding rapidly ever since.

Due to the fact that nanoelectronic software components are new and that sales are restricted to corporate customers, Nanonull and its product line have not received much media publicity in the company's early years. This has however changed in recent months as trade journals have realized the importance of this revolutionary technology.

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:

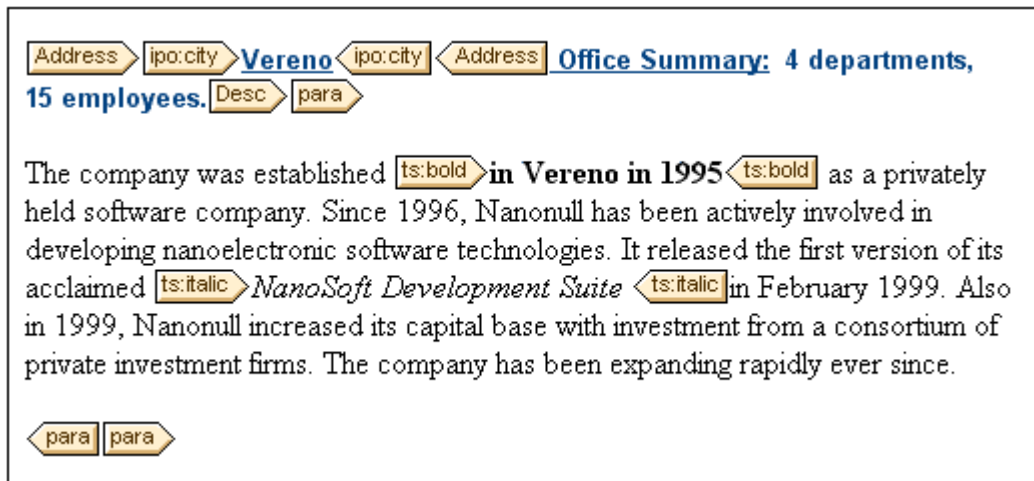
- Formatierung von Absätzen und anderen Blöcken
- Tabellenstrukturen
- Textformatierung wie z.B. Farbe und Schriftart
- Mit Hilfe von Auswahllisten (siehe Felder "State & Zip") kann der Benutzer einen Eintrag aus einer Gruppe von gültigen Auswahlmöglichkeiten auswählen. Diese Listeneinträge können wie 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)

SPSSs 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. SPSSs 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".

5.6 Eingabehilfen für XML-Dokumente

Es gibt drei Eingabehilfen für XML-Dokumente: die Element-Eingabehilfe, die Attribut-Eingabehilfe und die Entities-Eingabehilfe. Wenn ein Element über die Element-Eingabehilfe hinzugefügt wird, können Sie es je nach der [im Dialogfeld "Optionen" auf dem Register "Bearbeiten"](#) 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

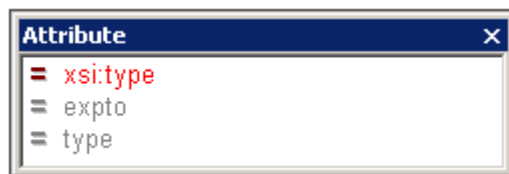
Beachten Sie die folgenden Punkte:

- *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.
- *Grid-Ansicht*: Elemente können hinter dem ausgewählten Element angehängt werden, davor eingefügt oder als Child-Element des ausgewählten Elements hinzugefügt werden. Es gibt daher drei Register, auf denen jeweils die Elemente angezeigt werden, die hinzugefügt werden können. Nicht verwendete Elemente werden schwarz, verwendete Elemente grau angezeigt.
- *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 weisen ein Ausrufezeichen "!" vor dem Namen des Attributs auf.



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.

- *Grid-Ansicht*: Wenn ein Element ausgewählt ist, werden die Attribute, die als Child-Attribute eingefügt werden können, auf dem Eingabehilferegister "Child hinzufügen" aufgelistet. Wenn ein Attribut ausgewählt ist, werden die verfügbaren Attribute auf den Registern "Anhängen" (hinter) und "Einfügen" (vor) aufgelistet. Nicht verwendete Attribute werden schwarz angezeigt, verwendete Attribute grau.
- *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.

5.7 Verarbeitung mit XSLT und XQuery

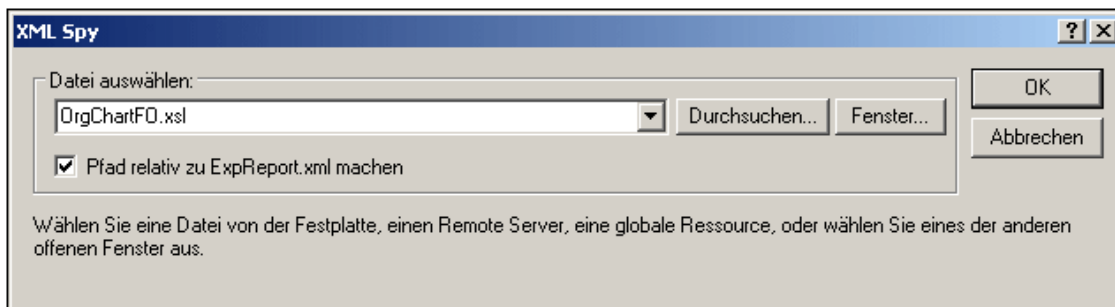
XML-Dokumente können mit XSLT- oder XQuery-Dokumenten verarbeitet werden um damit Ausgabedokumente zu erzeugen. XMLSpy hat einen integrierten XSLT 1.0-, XSLT 2.0- und XQuery 1.0-Prozessor. Über die Benutzeroberfläche stehen Ihnen die folgenden verarbeitungsbezogenen Funktionen zur Verfügung:

- [Zuweisen von XSLT Stylesheets](#)
- [Gehe zu XSLT](#)
- [XSLT-Parameter und XQuery-Variablen](#)
- [XSLT-Transformationen](#)
- [XQuery-Ausführungen](#)
- [Automatisieren von XML-Aufgaben mit AltovaXML](#)

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](#).

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 "Benutzerreferenz"](#).

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" auf dem Register [XSL](#) definiert ([Extras | Optionen](#)).
- 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](#)) auf dem [Register XSL](#) 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](#)) auf dem [Register XSL](#) 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 AltovaXML

AltovaXML ist eine kostenlose Applikation, die den XML Validator, und den XSLT 1.0-, XSLT 2.0- und XQuery 1.0-Prozessor enthält. Sie kann über die Befehlszeile, über eine COM-Schnittstelle, in Java-Programmen und in .NET-Applikationen aufgerufen werden, um XSL-Dokumente zu validieren und XML-Dokumente anhand von XSLT 1.0- und 2.0 Stylesheets zu transformieren und XQuery-Dokumente auszuführen.

XSLT-Transformationen können daher mittels AltovaXML automatisiert werden. So können Sie z.B. eine Batch-Datei erstellen, die AltovaXML aufruft, um eine Gruppe von Dokumenten zu transformieren. Nähere Informationen dazu finden Sie in der [Dokumentation zu AltovaXML](#).

5.8 Diagramme

Wenn ein XML-Dokument in der Text- oder Grid-Ansicht geöffnet ist, kann im Diagrammfenster (das eines der [Ausgabefenster](#) 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.

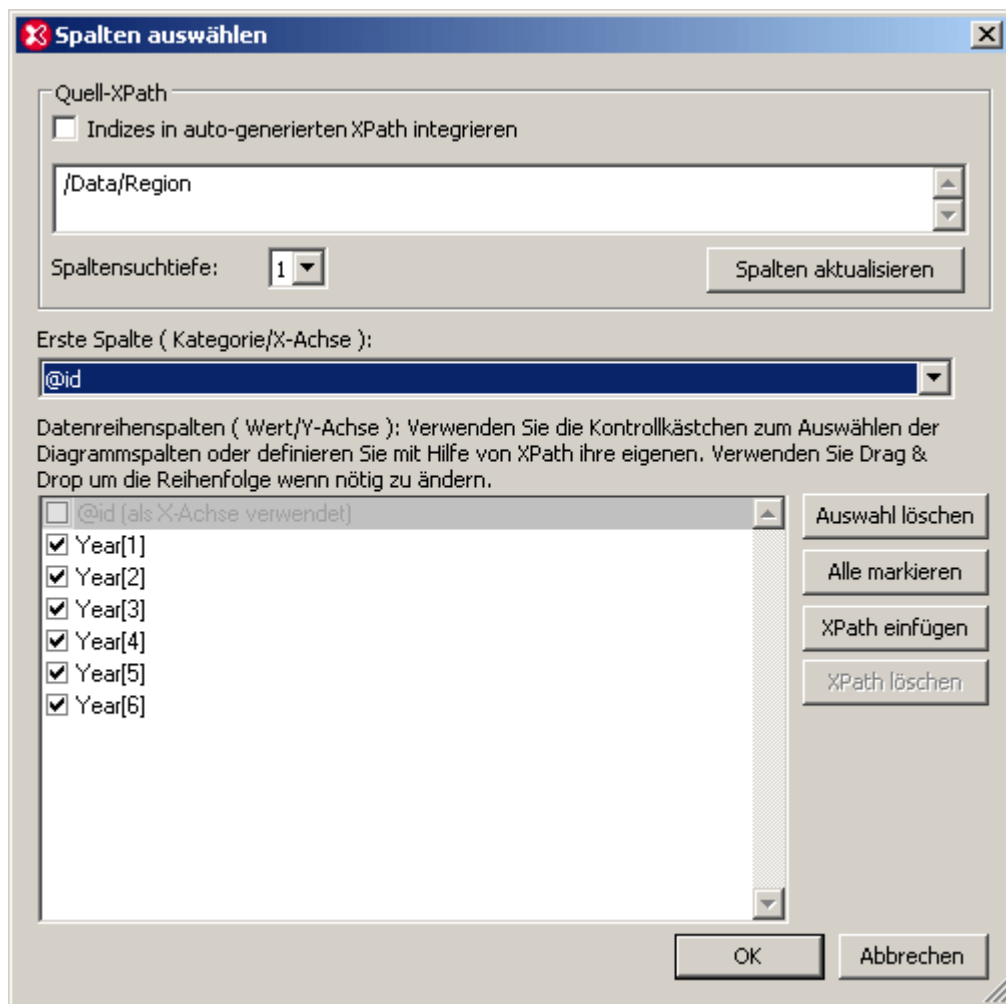
Die folgenden Diagrammtypen stehen zur Verfügung:

- 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

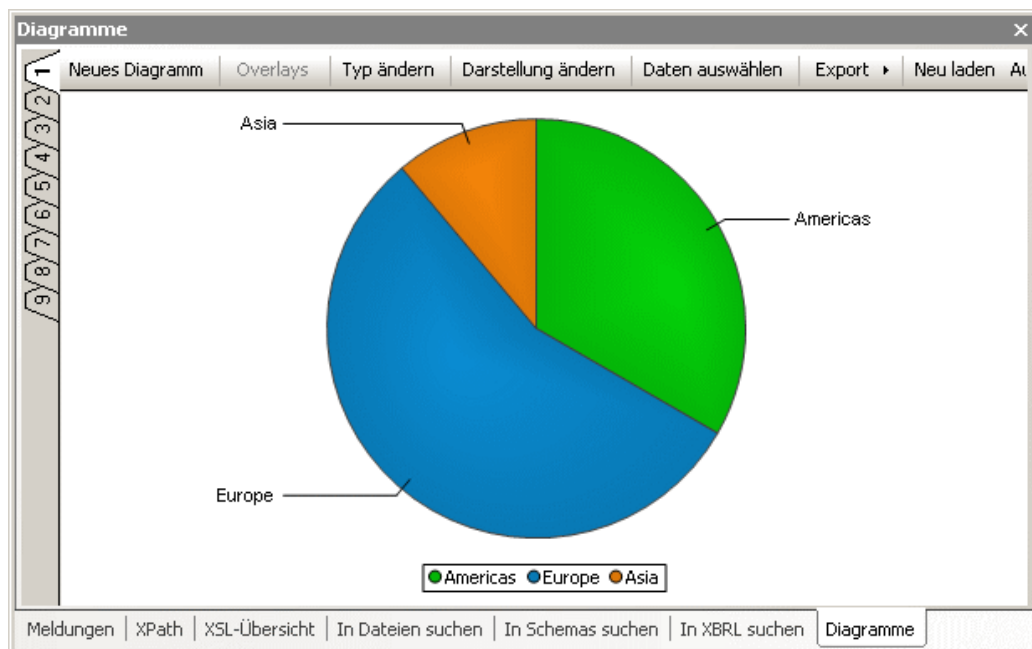
Übersicht: von der Erstellung bis zum Export

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 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- und die Y-Achse ausgewählt werden und der Quell-XPath geändert wird.



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 `C:\Dokumente und Einstellungen \<Benutzername>\Eigene Dateien\Altova\XMLSpy2012\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>

```

5.8.1 Diagrammbeispiel: Einfach

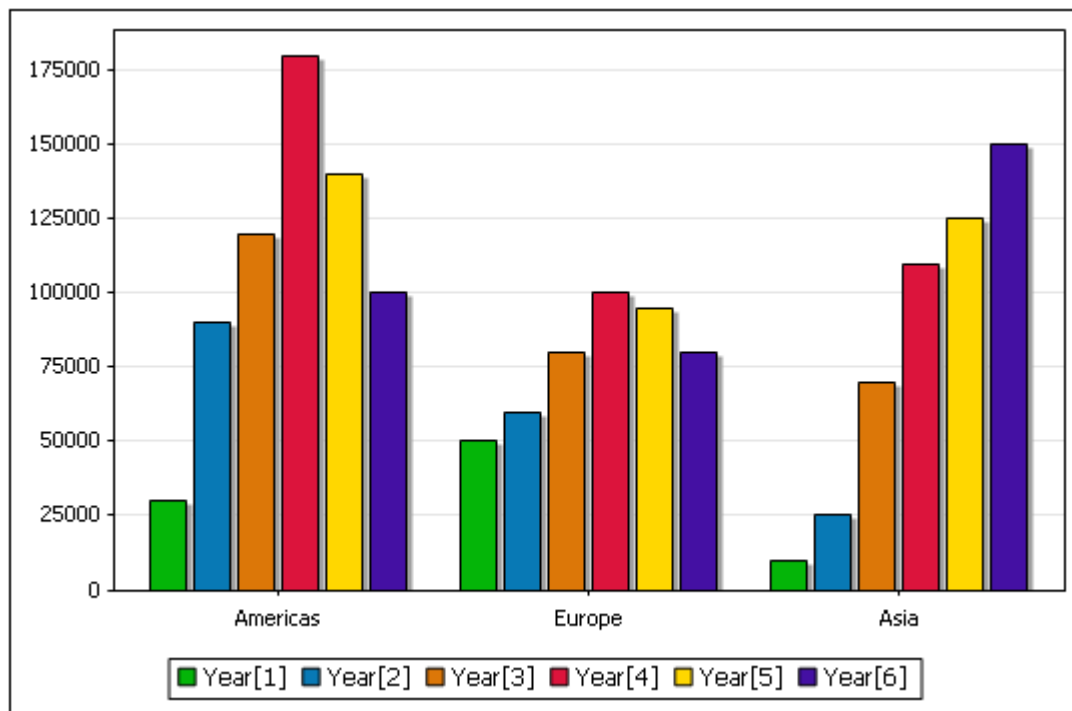
Angenommen, Sie haben das folgende XML-Dokument. (Es hat den Namen `YearlySales.xml` und befindet sich im Ordner `C:\Dokumente und Einstellungen\<Benutzername>\Eigene Dateien\Altova\XMLSpy2012\Examples\Tutorial.`)

```

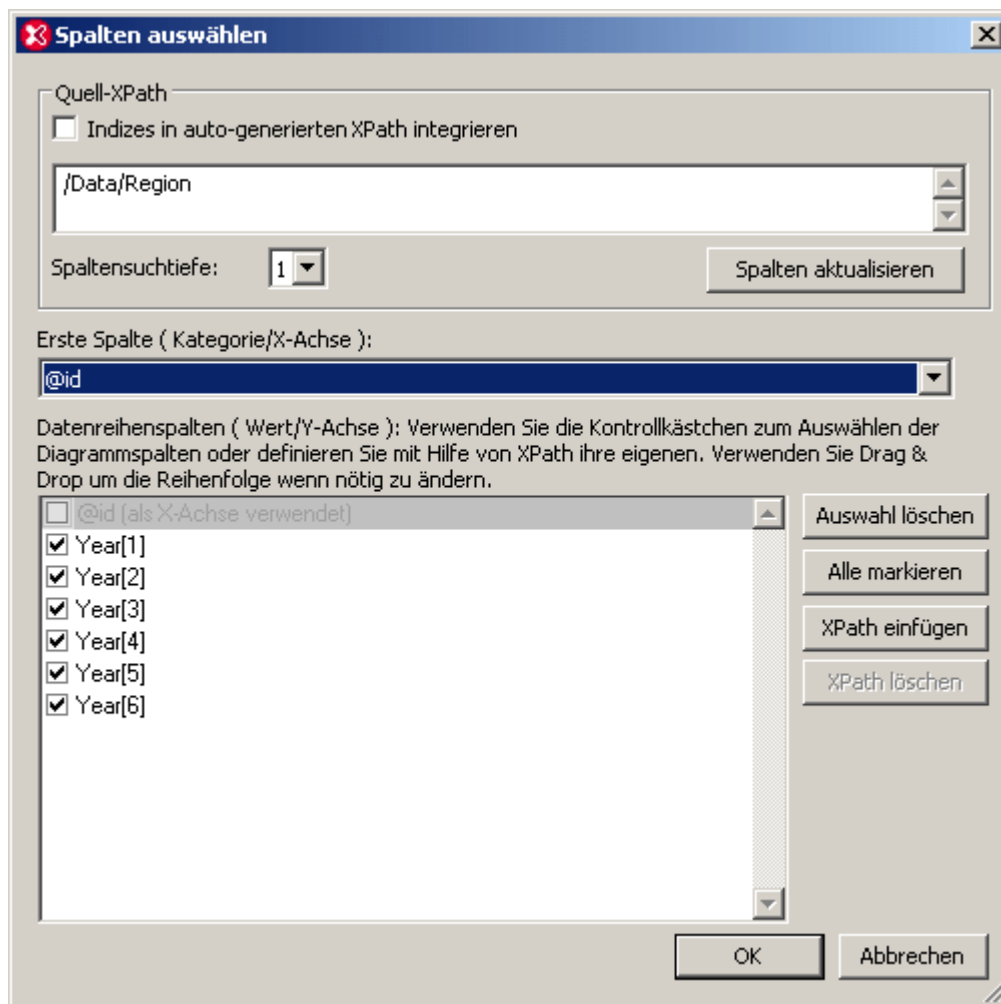
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
    <Region id="Americas">
        <Year id="2005">30000</Year>
        <Year id="2006">90000</Year>
        <Year id="2007">120000</Year>
        <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.



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**.

5.9 Zusätzliche Funktionen

Im Folgenden werden weitere Funktionen für die Arbeit mit XML-Dateien aufgelistet.

- [Kodierung](#)
- [Generieren von DTDs und XML-Schemas](#)
- [Suchen und Ersetzen](#)
- [XPath auswerten](#)
- [Importieren und Exportieren von Text](#)

Kodierung

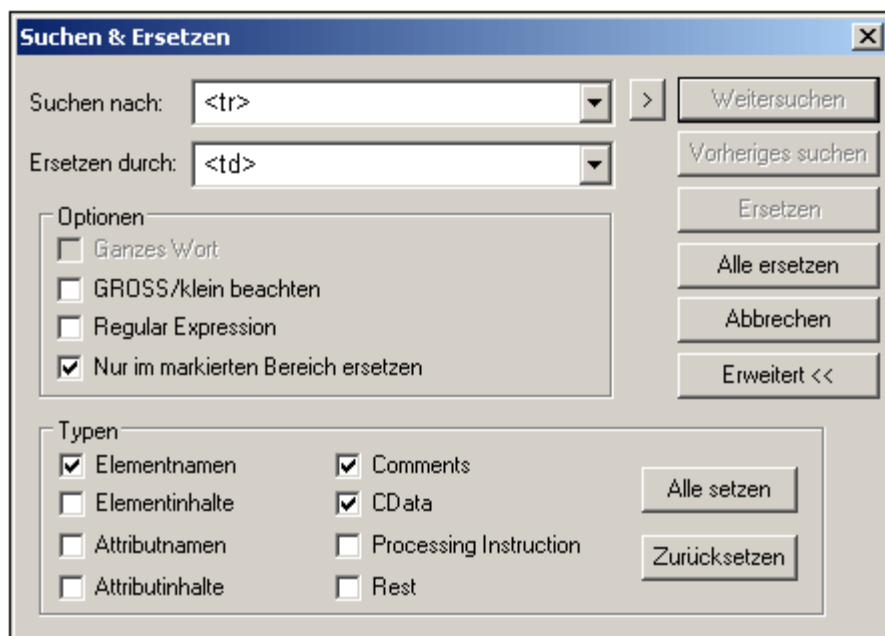
Die Kodierung von XML-Dateien (und anderen Dokumentarten) kann mit Hilfe des Menübefehls [Datei | Kodierung](#) gespeichert werden. Die Standardkodierung von XML- und Nicht-XML-Dateien kann auf dem Register [Optionen | Kodierung](#) gespeichert werden.

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](#). 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](#) und [Ersetzen](#) (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*).



In der Abbildung oben sehen Sie das Dialogfeld "Suchen & Ersetzen" in der Textansicht. Das Dialogfeld und seine Funktionalitäten sehen je nach aktiver Ansicht unterschiedlich aus.

XPath auswerten

Ein XPath-Ausdruck, den Sie in das Fenster "XPath" eingeben, kann gegen das aktive XML-Dokument ausgewertet werden. Die Ergebnisse der Auswertung werden im Fenster "XPath" 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" 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](#).

6 DTDs und XML Schemas

Altova Website:  [XML-Schema Editor](#)

Dieser Abschnitt enthält eine Übersicht über das Arbeiten mit [DTDs](#) und [XML-Schemas](#). Außerdem wird darin beschrieben, [wie man mit SchemaAgent arbeitet](#) und wie man die leistungsstarken [In Schemas suchen](#)-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](#) können URIs auf lokale Adressen umgeleitet werden, was die Verwendung auf mehreren Arbeitsplätzen erleichtert.

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](#).

Konvertieren von DTDs in XML-Schemas und umgekehrt

Mit dem Befehl [DTD/Schema | DTD/Schema konvertieren](#) kann eine DTD in ein XML-Schema konvertiert werden und umgekehrt. Eine Beschreibung dazu finden Sie im Abschnitt "Benutzerreferenz" unter der [Beschreibung zu diesem Befehl](#).

Erzeugen einer XML-Beispieldatei

Mit dem Menübefehl [DTD/Schema | XML-Beispieldatei erzeugen](#) wird anhand der DTD oder der 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](#) die Schema-Datei geöffnet und die Definition des ausgewählten XML-Node wird markiert.

- *Konvertieren einer DTD in ein XML-Schema:* Mit dem Befehl [DTD/Schema | DTD/Schema konvertieren](#) können DTDs in XML-Schemas konvertiert werden.
- *Generieren einer XML-Datei anhand einer DTD:* Mit dem Befehl [DTD/Schema | XML-Beispieldatei erzeugen](#) kann ein XML-Dokument generiert werden, das auf der aktiven DTD basiert.

6.2 XML-Schemas

XML-Schema-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" auf dem [Register "Dateiarten"](#) 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.

Bearbeiten in der Textansicht

In der Textansicht wird ein XML-Schema als XML-Dokument bearbeitet; die [Bearbeitungsfunktionen, die für XML-Dokumente zur Verfügung stehen](#), 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](#) 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](#), sind auch für XML-Schemas verfügbar. Wenn ein Element in der Grid-Ansicht ausgewählt ist, finden Sie in den [Eingabehilfen der Grid-Ansicht](#) 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 entsprechenden Textdokument. Eine Anleitung zum Erstellen und Bearbeiten von XML-Schema-Dokumenten in der Schema-Ansicht finden Sie im Abschnitt [Bearbeitungsansichten | Schema-Ansicht](#).

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](#) 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](#) kann ein XML-Dokument generiert werden, das auf dem aktiven XML-Schema basiert.

6.3 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:

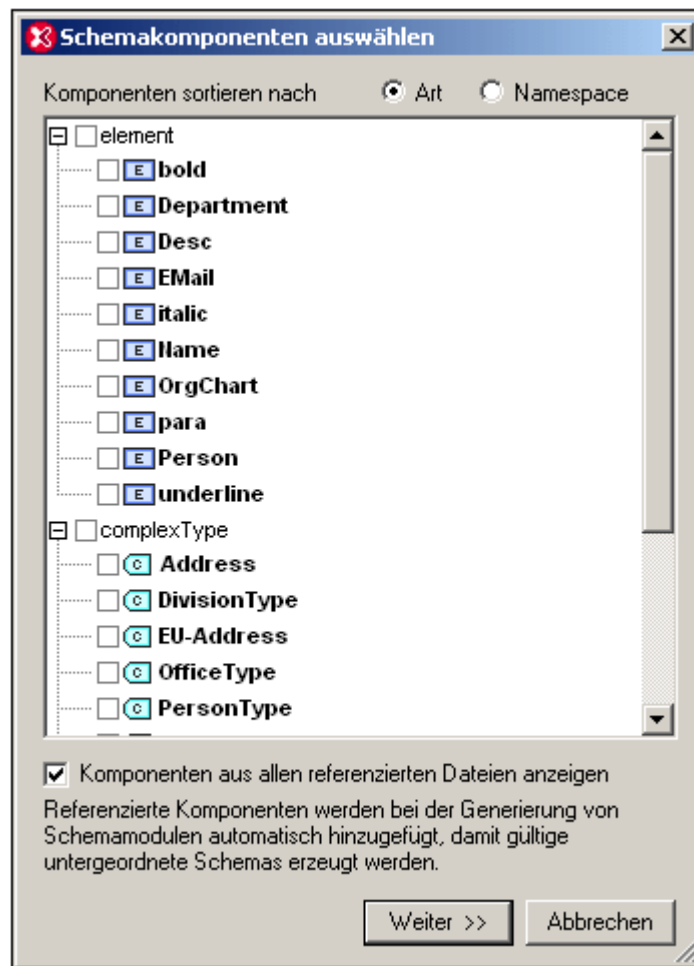
1. Erstellen Sie ein Schemamodul, das eine oder mehrere Komponenten des aktiven Schemas enthält. Eine Beschreibung dazu finden Sie [weiter unten](#).
2. Erstellen Sie je nach Bedarf weitere Schemamodule.
3. Inkludieren Sie das/die neu erstellten Schemamodul(e), um daraus das größere Schema zu bilden. Dies geschieht, indem Sie im [Fenster "Schemaübersicht"](#) 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](#).

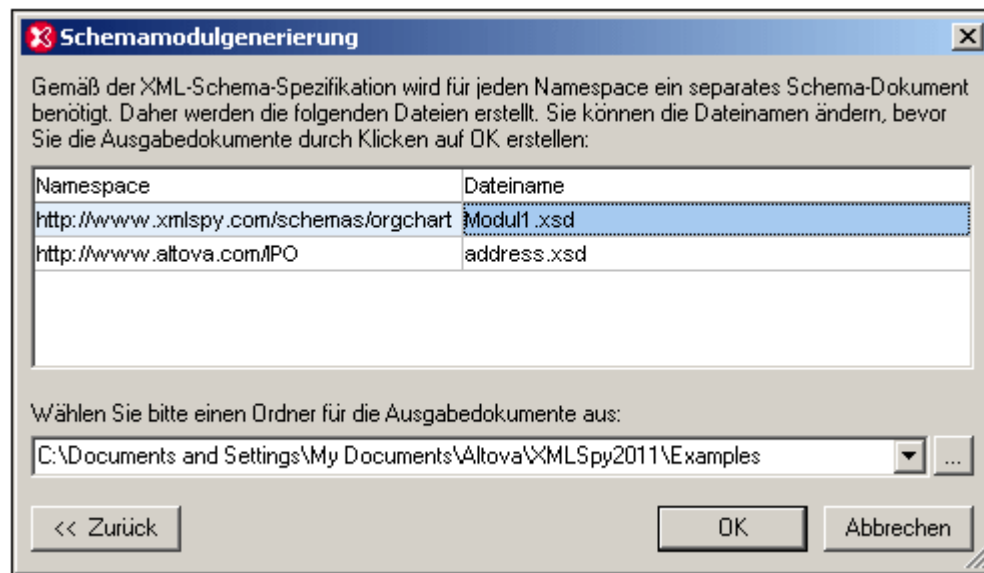
Erstellen von Schemamodulen

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.)



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.

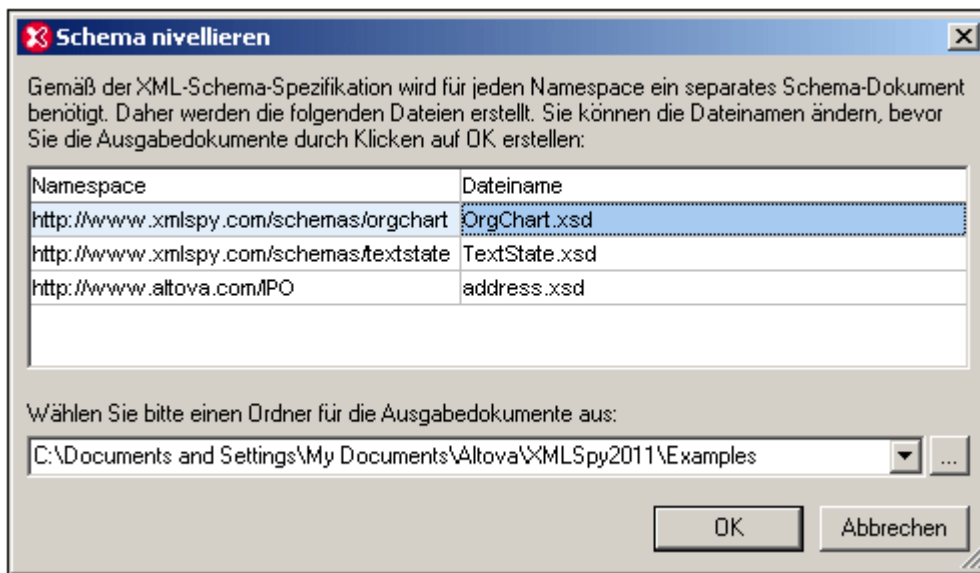


4. Wenn Sie auf **OK** klicken, wird die Schemamoduldatei mit dem Namespace, der dem der aktiven Datei entspricht, in der Schema-Ansicht geöffnet. Alle anderen Dateien im Paket werden erstellt, aber nicht in der Schema-Ansicht geöffnet.

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.



Wenn Sie auf **OK** klicken, wird die nivellierte Schema-Datei in der Schema-Ansicht geöffnet.

6.4 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 Mechanismus in XMLSpy funktioniert, wie unten beschrieben.

RootCatalog.xml

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 im Element `nextCatalog` in beliebig viele Katalogdateien definieren, die durchsucht werden sollen. Jede dieser Katalogdateien wird durchsucht und die URIs darin werden entsprechend den darin definierten Zuordnungen aufgelöst.

```
<?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"/>
  <nextCatalog catalog="CoreCatalog.xml"/>
  <!-- Include all catalogs under common schemas folder on the first directory
level -->
  <nextCatalog spy:recurseFrom="%AltovaCommonFolder%/Schemas" catalog="
catalog.xml" spy:depth="1"/>
  <!-- Include all catalogs under common XBRL folder on the first directory
level -->
  <nextCatalog spy:recurseFrom="%AltovaCommonFolder%/XBRL" catalog="
catalog.xml" spy:depth="1"/>
</catalog>
```

Beachten Sie, dass sich in der Liste oben in den Ordnern `Schemas` und `XBRL`, die durch die Variable `%AltovaCommonFolder%` definiert werden, Katalogdateien namens `catalog.xml` befinden. (Der Wert der `%AltovaCommonFolder%` Variablen wird in der Tabelle unten aufgelistet.)

Die Katalogdateien im Altova-Ordner "Common" mappen die vordefinierten Public und System Identifiers häufig verwendeter Schemas (wie z.B. SVG) und XBRL-Taxonomien auf URIs, die auf lokal gespeicherte Kopien der entsprechenden Schemas verweisen. Diese Schemas werden bei der Installation von XMLSpy im Altova Ordner "Common" installiert. Achten Sie darauf, in diesen Dateien keine Zuordnungen doppelt zu vergeben, da dies Fehler verursachen könnte.

CoreCatalog.xml, CustomCatalog.xml und Catalog.xml

Beachten Sie in der `RootCatalog.xml` Liste oben, dass `CoreCatalog.xml` und `CustomCatalog.xml` als Lookup aufgelistet sind:

- `CoreCatalog.xml` enthält bestimmte Altova-spezifische Zuweisungen zum Auffinden von Schemas im Altova Ordner "Common".
- `CustomCatalog.xml` ist eine Skelettdatei, in der Sie ihre eigenen Zuweisungen erstellen können. Sie können für jedes gewünschte Schema Zuweisungen zu `CustomCatalog.xml` hinzufügen, doch dieses wird nicht von den Katalogdateien im Altova Common-Ordner angegeben. Sie können dies mit Hilfe der unterstützten

- Elemente des OASIS-Katalogs (*siehe unten*) tun (*siehe unten*).
- Der Altova-Ordner "Common" enthält eine Reihe von `Catalog.xml` Dateien. Jede befindet sich in einem Ordner eines bestimmten Schemas oder einer XBRL-Taxonomie im Altova-Ordner "common" und jede weist URIs Public und/oder System Identifier zu, die auf lokal gespeicherte Kopien der entsprechenden Schemas verweisen.

Speicherpfad von Katalogdateien und Schemas

Die Dateien `RootCatalog.xml` und `CoreCatalog.xml` sind im XMLSpy Applikationsordner installiert. Die Datei `CustomCatalog.xml` befindet sich im Ordner `MyDocuments\Altova\XMLSpy`. Die `Catalog.xml` Dateien befinden sich in den jeweiligen Schemaordnern. Diese Schemaordner befinden sich im Ordner: `%AltovaCommonFolder%\Schemas` und `%AltovaCommonFolder%\XBRL`.

Shell-Umgebungsvariablen und Altova-Variablen

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:

<code>%AltovaCommonFolder%</code>	C:\Program Files\Altova\Common2012
<code>%DesktopFolder%</code>	Vollständiger Pfad zum Desktop-Ordner für den aktuellen Benutzer.
<code>%ProgramMenuFolder%</code>	Vollständiger Pfad zum Programmmenü-Ordner für den aktuellen Benutzer.
<code>%StartMenuFolder%</code>	Vollständiger Pfad zum Startmenü-Ordner für den aktuellen Benutzer.
<code>%StartupFolder%</code>	Vollständiger Pfad zum Startordner für den aktuellen Benutzer.
<code>%TemplateFolder%</code>	Vollständiger Pfad zum Vorlagen-Ordner für den aktuellen Benutzer.
<code>%AdminToolsFolder%</code>	Vollständiger Pfad zum Dateisystemverzeichnis, in dem die Verwaltungstools für den aktuellen Benutzer gespeichert sind.
<code>%AppDataFolder%</code>	Vollständiger Pfad zum Applikationsdatenordner für den aktuellen Benutzer.
<code>%CommonAppDataFolder%</code>	Vollständiger Pfad zum Dateiverzeichnis mit den Applikationsdaten für alle Benutzer.
<code>%FavoritesFolder%</code>	Vollständiger Pfad zum Ordner "Favoriten" für den aktuellen Benutzer.
<code>%PersonalFolder%</code>	Vollständiger Pfad zum persönlichen Ordner für den aktuellen Benutzer.
<code>%SendToFolder%</code>	Vollständiger Pfad zum Ordner "SendTo" für den aktuellen Benutzer.
<code>%FontsFolder%</code>	Vollständiger Pfad zum Systemschriftarten-Ordner.
<code>%ProgramFilesFolder%</code>	Vollständiger Pfad zum Programmdateiordner für den aktuellen Benutzer.
<code>%CommonFilesFolder%</code>	Vollständiger Pfad zum Ordner "Gemeinsame Dateien" für den aktuellen Benutzer.
<code>%WindowsFolder%</code>	Vollständiger Pfad zum Windows-Ordner für den aktuellen Benutzer.
<code>%SystemFolder%</code>	Vollständiger Pfad zum Systemordner für den aktuellen Benutzer.

%CommonAppData Folder%	Vollständiger Pfad zum Dateiverzeichnis mit den Applikationsdaten für alle Benutzer.
%LocalAppDataF older%	Vollständiger Pfad zum Dateisystemverzeichnis, das als Datenspeicher für lokale (nicht-Roaming) Applikationen dient.
%MyPicturesFol der%	Vollständiger Pfad zum Ordner "Meine Bilder".

Funktionsweise von Katalogen

Kataloge dienen normalerweise dazu, einen Aufruf von einer DTD auf eine lokale URI umzuleiten. Dies geschieht in der Katalogdatei durch Mappen von Public und System Identifiers auf die gewünschte lokale URI. Wenn also die DOCTYPE-Deklaration in einer XML-Datei gelesen wird, findet der Public oder System Identifier über das Katalogdatei-Mapping die gewünschte lokale Ressource.

Für gebräuchliche Schemas ist der PUBLIC Identifier normalerweise vordefiniert, sodass nur die URI in der Katalogdatei in die Adresse der richtigen lokalen Kopie geändert werden muss. Wenn das XML-Dokument geparkt 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>
```

Dieses Dokument wird gelesen und der Katalog wird nach dem PUBLIC Identifier 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 auf die URI `schemas/svg/svg11.dtd` umgeleitet wird; (dieser Pfad ist relativ zur Katalogdatei). Diese lokale Datei wird dann als DTD verwendet. Wenn im Katalog kein passender Treffer gefunden wird, wird die URL im XML-Dokument verwendet (im Beispiel oben: `http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd`).

Die von XMLSpy unterstützte Katalogteilmenge

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](#). Bitte beachten Sie: Jedes Element kann das Attribut `xml:base` erhalten, welches dazu dient, die Basis-URI dieses Elements zu definieren.

- `<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"/>`

In Fällen, in denen es wie z.B. bei den meisten Stylesheets keinen Public Identifier gibt, kann der System Identifier einer URL direkt über das `system` Element zugewiesen werden. Außerdem kann eine URI auch über das `uri` Element einer anderen URI zugewiesen werden. Mit Hilfe der Elemente `rewriteURI` und `rewritsSystem` kann der Anfangsteil einer URI bzw. eines System Identifier neu geschrieben werden. Auf diese Art kann der Beginn eines Dateipfads ersetzt werden, wodurch ein anderes Verzeichnis als Ziel definiert werden kann. Nähere Informationen zu diesen Elementen finden Sie in der [XML Catalogs Specification](#).

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.

```
<spy:fileExtHelper ext="myhtml" uri="schemas/xhtml/xhtml1-transitional.dtd"/>
```

Damit werden die intelligenten Bearbeitungsfunktionen (Autokomplettierung, Eingabehilfen usw.) von `.myhtml`-Dateien in XMLSpy gemäß der XHTML 1.0 Transitional DTD aktiviert.

XML-Schema und Kataloge

XML-Schemainformationen sind in XMLSpy integriert und die Gültigkeit von XML-Schema-Dokumenten wird anhand dieser internen Informationen überprüft. Daher sollte in einem XML-Schema-Dokument kein Verweis auf ein Schema für XML-Schema vorgenommen werden.

Die Datei `Catalog.xml` im Ordner `%AltovaCommonFolder%\Schemas\schemas` enthält Referenzen auf DTDs, die ältere XML-Schema-Spezifikationen implementieren. 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.

Weitere Informationen

Nähere Informationen zu Katalogen finden Sie in der [XML-Katalogspezifikation](#).

6.5 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 (Ordern auf dem Netzwerk) enthält.
- Einem SchemaAgent Client, dem Altova Produkt SchemaAgent 2012, 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 2012, der auf einem Netzwerk installiert werden kann, und auf den Sie über das Netzwerk zugreifen können und
- Altova SchemaAgent 2012. 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:

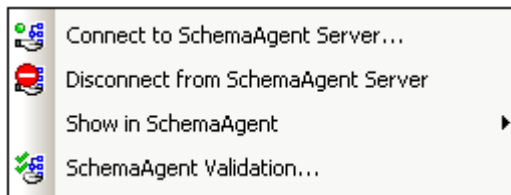
- Downloaden Sie SchemaAgent von der [Altova Website](#) und installieren Sie ihn auf Ihrem Netzwerk. Sie können nun über den LocalServer von SchemaAgent auf Schemas zugreifen. Informationen zum Konfigurieren von Suchpfaden in LocalServer finden Sie im SchemaAgent-Benutzerhandbuch. **Beachten Sie**, dass Sie für SchemaAgent eine Lizenz benötigen, die Sie nach Ablauf des Testzeitraums erwerben müssen, um das Produkt weiterverwenden zu können. Beachten Sie, dass die Altova MissionKit-Pakete das Produkt SchemaAgent sowie einen Lizenzschlüssel dazu enthalten. (Die SchemaAgent Server-Applikation ist jedoch nicht in den 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.



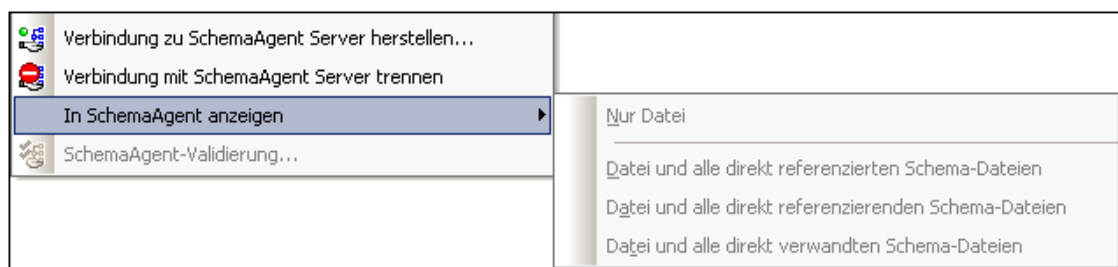
Über die Menübefehle stehen Ihnen allgemeine Verwaltungsfunktionen zur Verfügung. Die Eingabehilfen (und die Standardfunktion der Benutzeroberfläche wie z.B. Drag & Drop) dienen zum Bearbeiten von Schemas.

In diesem Abschnitt wird beschrieben, wie man die in der Schema-Ansicht zur Verfügung stehenden SchemaAgent-Funktionen verwendet.

6.5.1 Herstellen einer Verbindung mit SchemaAgent Server

Bitte beachten Sie: 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.



Bitte beachten Sie: Wenn SchemaAgent Client nicht installiert ist, kann keine Verbindung zu SchemaAgent Server hergestellt werden.

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 2012 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.

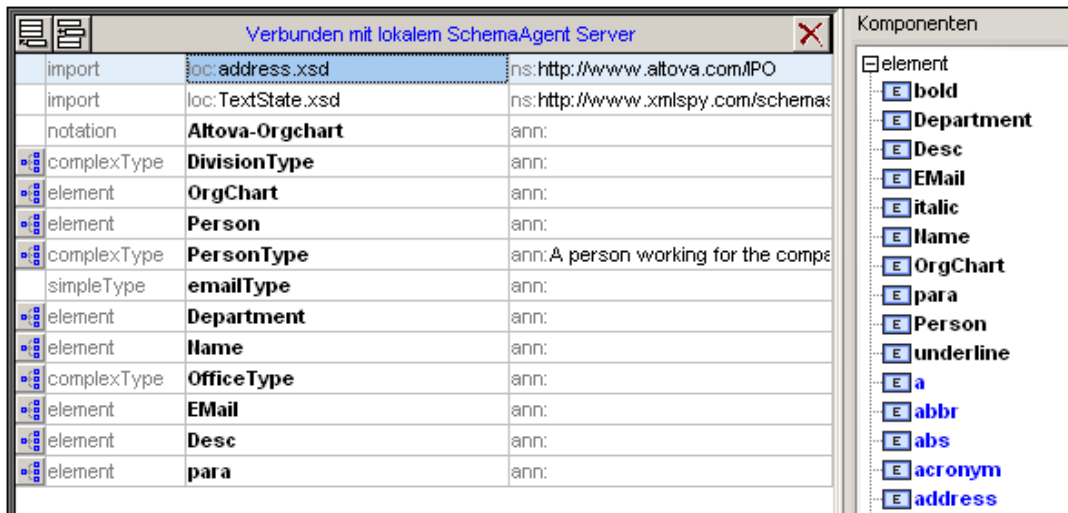
Um die Verbindung zu einem derartigen Server herzustellen, haben Sie folgende Möglichkeiten:

- 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:



The screenshot shows the SchemaAgent interface with a connection to a local server. The main window displays a list of XML elements and their associated schemas. The 'Komponenten' (Components) panel on the right lists the elements available in the schema.

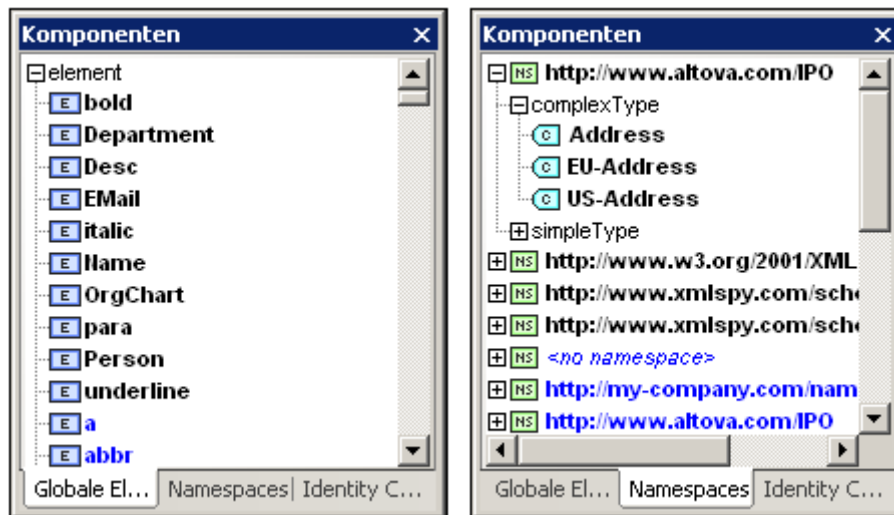
Import	Schema	Namespace
import	loc:address.xsd	ns:http://www.altova.com/IPO
import	loc:TextState.xsd	ns:http://www.xmlspy.com/schemas
notation	Altova-Orgchart	ann:
complexType	DivisionType	ann:
element	OrgChart	ann:
element	Person	ann:
complexType	PersonType	ann:A person working for the comp
simpleType	emailType	ann:
element	Department	ann:
element	HName	ann:
complexType	OfficeType	ann:
element	EMail	ann:
element	Desc	ann:
element	para	ann:

The 'Komponenten' panel on the right lists the following elements:

- element
- bold
- Department
- Desc
- EMail
- italic
- HName
- OrgChart
- para
- Person
- underline
- a
- abbr
- abs
- acronym
- address

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.

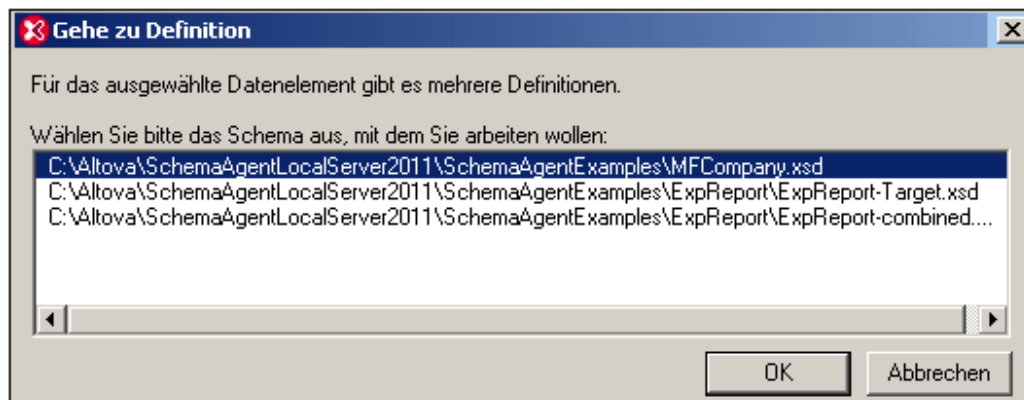


Schema-Konstrukte können in der Schema-Ansicht auf den entsprechenden Registern nach Typ (Globale Elemente), nach Namespace oder nach Identity Constraint angezeigt werden.

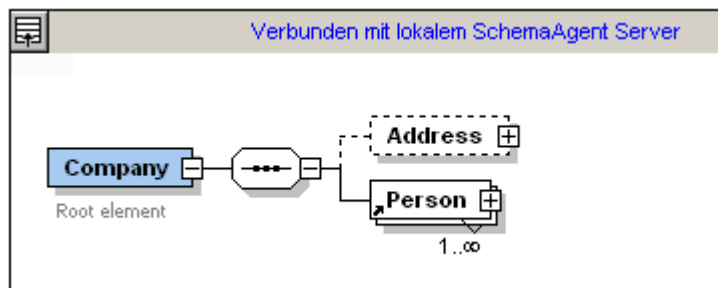
6.5.2 Öffnen von Schemas aus dem Suchpfad

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.



Bitte beachten Sie: 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.

6.5.3 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. `complexType`s und `simpleTypes` können aus einem Listenfeld ausgewählt werden, das automatisch geöffnet wird, wenn Sie ein neues Element/Attribut usw. definieren.
- Komponenten wie z.B. `complexType`s können bei der Erstellung/Aktualisierung dieser Konstruktart in der Details-Eingabehilfe ausgewählt werden.

Integrieren von Schema-Komponenten

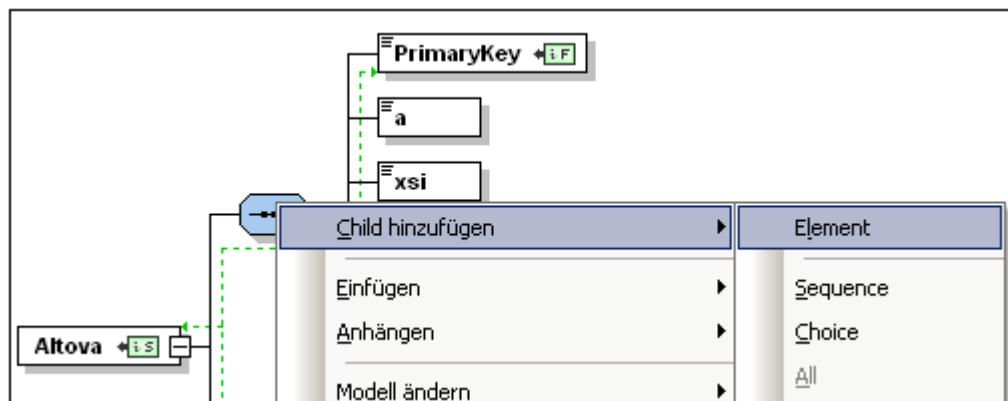
In diesem Beispiel wird die Datei `DB2schema.xsd` aus dem Ordner `.. \Tutorial` als aktives Schema verwendet; das Register `Globale Elemente` der Komponenten-Eingabehilfe ist aktiv.

So verwenden Sie Schema-Konstrukte aus SchemaAgent Server Schemas:

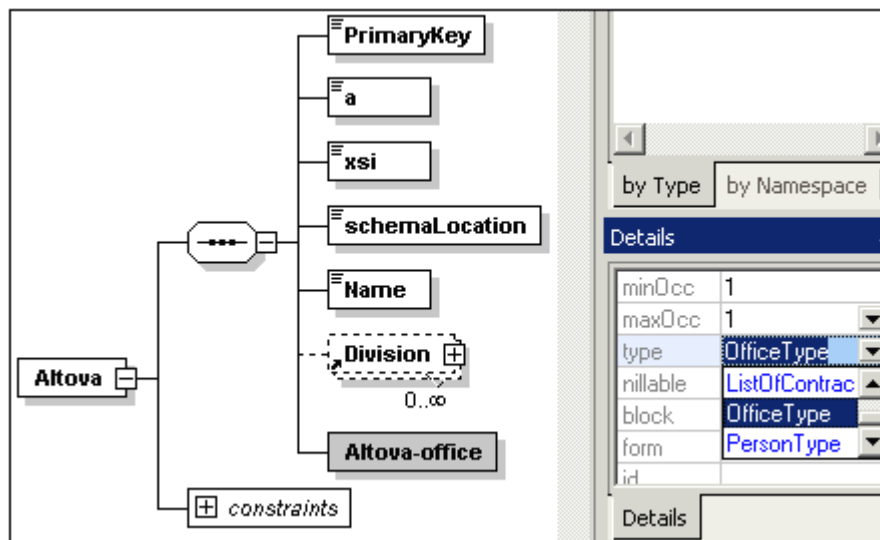
1. Stellen Sie sicher, dass eine Verbindung mit einem SchemaAgent Server hergestellt ist (siehe [Herstellen einer Verbindung mit SchemaAgent Server](#)).
2. Öffnen Sie die Datei DB2Schema.xsd und benennen Sie sie für dieses Beispiel in Altova-office um.

Verbunden mit lokalem SchemaAgent Server		
element	Altova	ann:
element	Division	ann:
element	Person	ann:
element	VIP	ann:

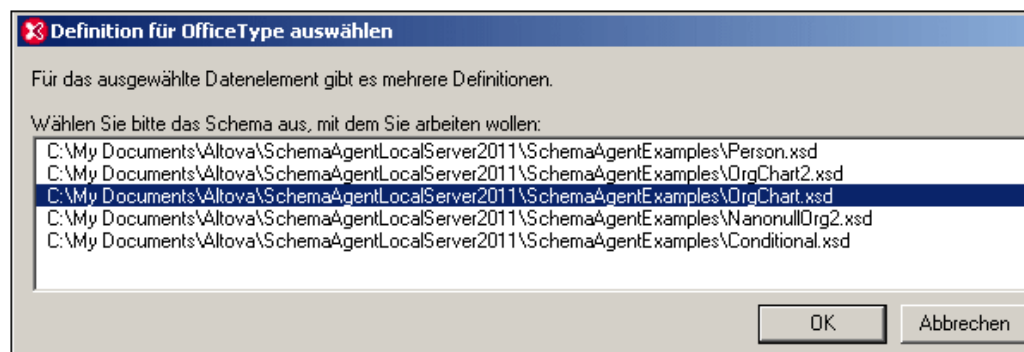
3. Klicken Sie in der Schema-Übersicht auf das Element `Altova`, um dessen Content Model aufzurufen.
4. Rechtsklicken Sie auf den `Altova` sequence-Kompositor und wählen Sie die Menüoption **Child hinzufügen | Element**. Beachten Sie, dass nun automatisch ein Listenfeld mit allen globalen Elementen aus dem Serverpfad geöffnet wird. Wenn Sie eines auswählen, wird dieses Element integriert.



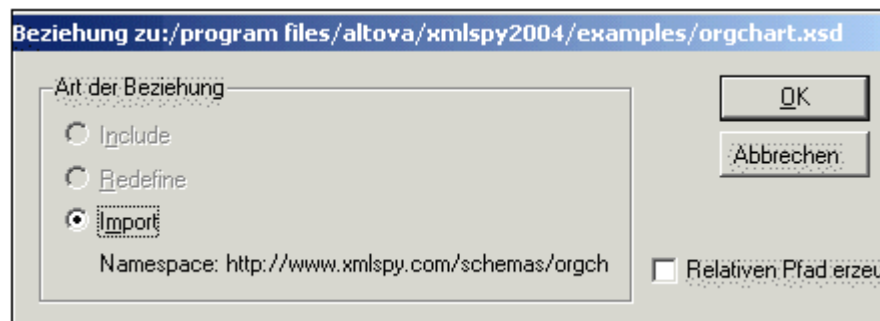
5. Geben Sie `Altova-office` als Namen für dieses neue Element ein und drücken Sie die Eingabetaste.
6. Klicken Sie in der Details-Eingabehilfe auf das Listenfeld `type` und wählen Sie den Eintrag `OfficeType` aus.



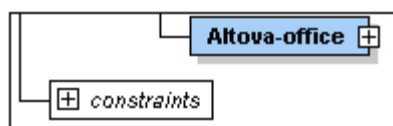
Daraufhin wird das Dialogfeld "Definition für OfficeType auswählen" geöffnet.



7. Wählen Sie Orgchart. xsd und klicken Sie auf OK.

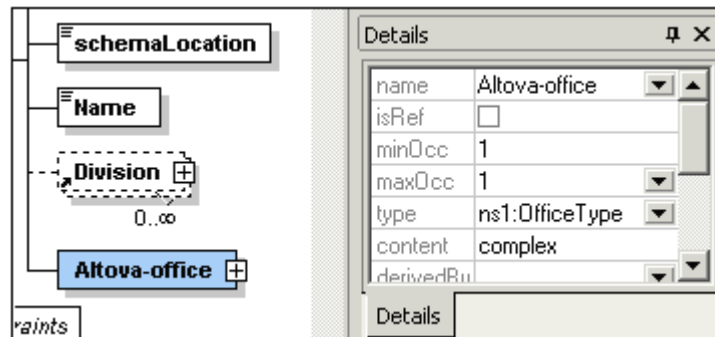


8. Klicken Sie im nächsten Dialogfeld auf OK. Der Befehl "Import" wurde automatisch ausgewählt. Das Element **Altova-office** weist nun ein Plus-Zeichen zum Erweitern auf.

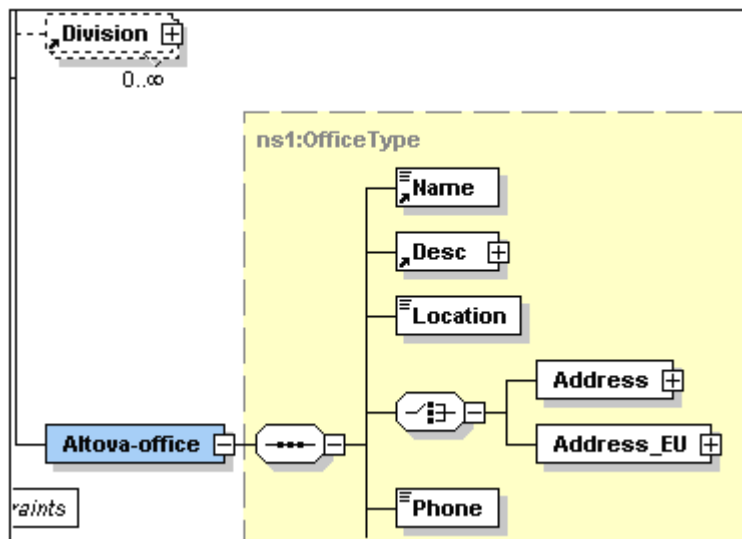


Bitte beachten Sie, dass sich der Eintrag **type** entry in der Details-Eingabehilfe geändert hat; er wird nun als **ns1:OfficeType** angezeigt, da die Schemadatei

Orgchart.xsd importiert wurde und die beiden Schemas unterschiedliche Target Namespaces haben müssen. Außerdem wurde zum Schema ein Import-Befehl hinzugefügt.



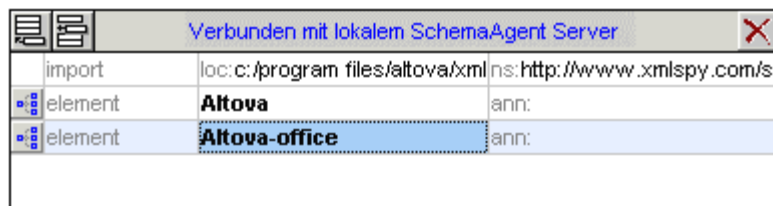
9. Klicken Sie auf das Erweiterungssymbol, um das Content Model von `OfficeType` anzuzeigen.



10. Drücken Sie F8, um das Schema zu validieren. Daraufhin sollte die Meldung, dass das Schema gültig ist, erscheinen.

Aufräumen des Schemas:

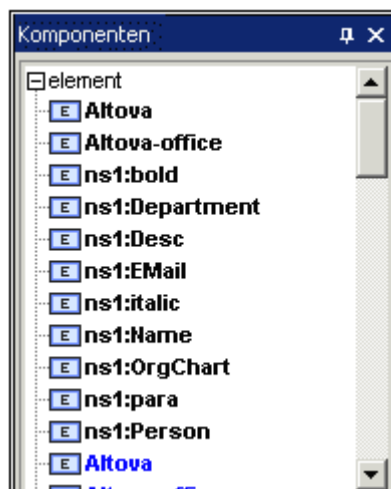
1. Löschen Sie das Element `Division` im Content Model.
2. Klicken Sie auf das Symbol zum Anzeigen der globalen Elemente, um in die Schema-Übersicht zu wechseln.
3. Löschen Sie die folgenden globalen Elemente: `Division`, `Person` und `VIP`.



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.

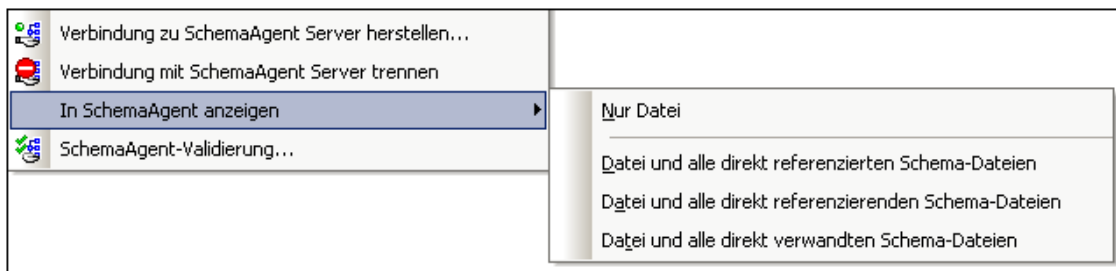


Bitte beachten Sie:

- 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 Eingabehilfen doppelt angezeigt (in Schwarz und Blau), wenn sich das Schema, mit dem Sie gerade arbeiten, auch unter dem SchemaAgent Serverpfad befindet.

6.5.4 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).



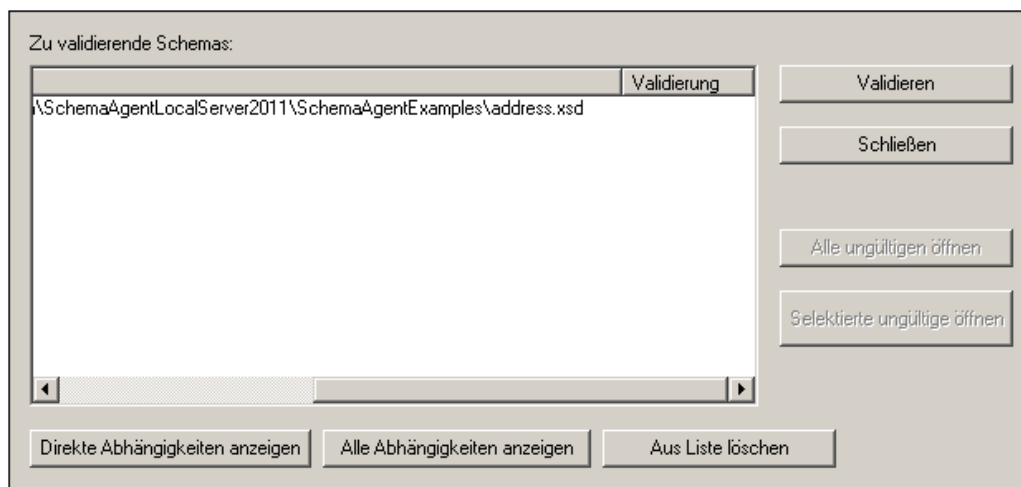
Sie haben die Option, entweder nur das aktive Schema in SchemaAgent zu öffnen (Befehl **Nur Datei**) oder das aktive Schema zusammen mit entweder (i) allen direkt referenzierten Schemas oder (ii) allen direkt referenzierenden Schema oder (iii) allen direkt verwandten Schemas zu öffnen.

6.5.5 SchemaAgent Validierung

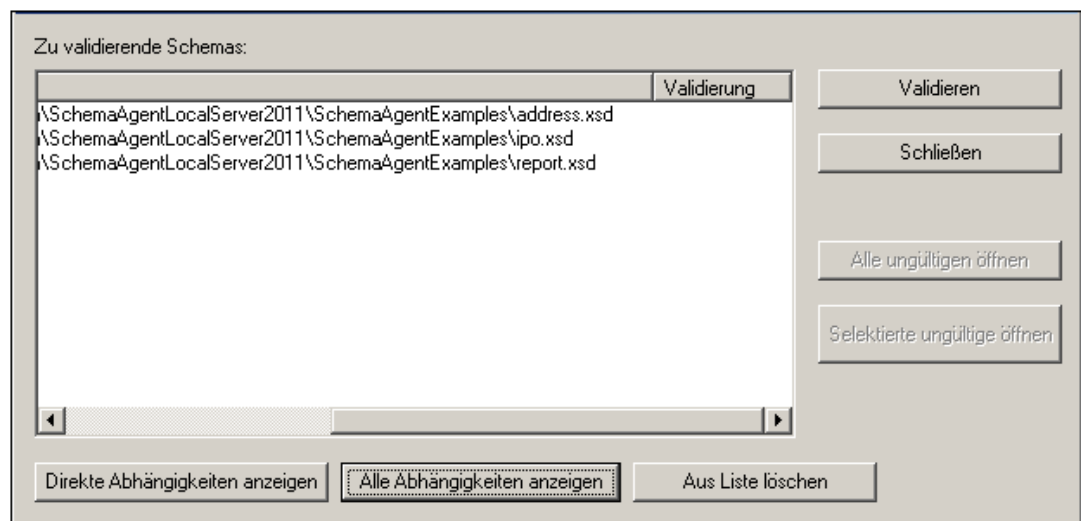
XMLSpy gestattet Ihnen in Verbindung mit SchemaAgent nicht nur das aktive Schema zu validieren, sondern auch Schemas, die mit dem aktiven Schema in Zusammenhang stehen. Wir bezeichnen dies als erweiterte Validierung. SchemaAgent unterscheidet bei der erweiterten Validierung zwischen: (i) direkt referenzierten Schemas und (ii) referenzierten Schemas.

Die Funktionsweise der erweiterten Validierung wird im Folgenden anhand eines Beispiels beschrieben. In diesem Beispiel wird als aktive Schemadatei in der Schema-Ansicht von XMLSpy `address.xsd` verwendet. Gehen Sie bei der erweiterten Validierung vor wie folgt:

1. Klicken Sie in der Symbolleiste auf das Symbol "Erweiterte Validierung"  oder wählen Sie den Menüeintrag **Schema-Design | SchemaAgent-Validierung**. Daraufhin wird das Dialogfeld "SchemaAgent-Validierung für..." geöffnet. Hier können Sie angeben, ob nur das aktive Schema oder auch ein oder mehrere damit verknüpfte Schemas validiert werden sollen.

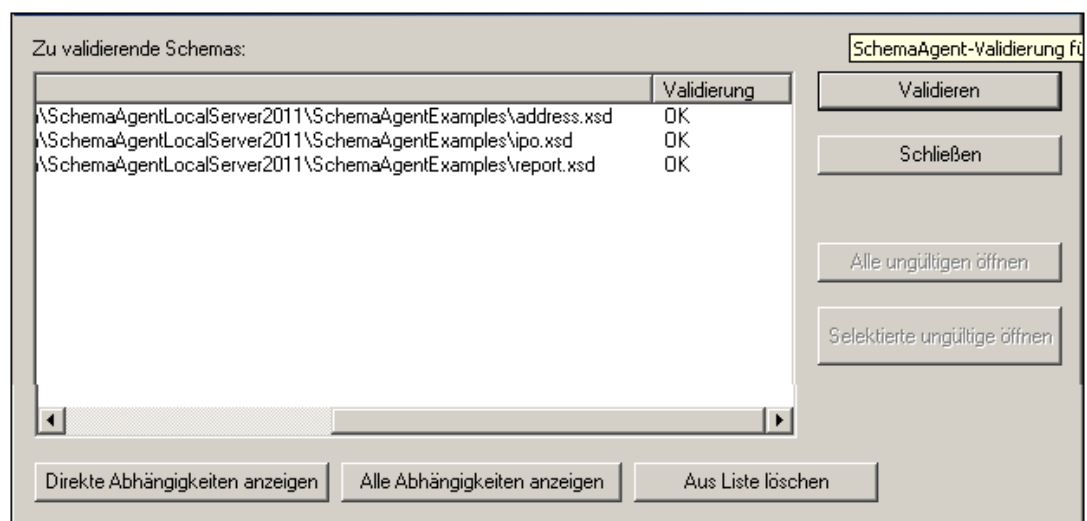


2. Um Schemas zur Liste hinzuzufügen, klicken Sie je nach Bedarf entweder auf "Direkte Abhängigkeiten anzeigen" oder auf "Alle Abhängigkeiten anzeigen". In diesem Beispiel haben wir die Option "Alle Abhängigkeiten anzeigen" gewählt, woraufhin alle referenzierten Dateien in die Liste übernommen werden.



Sie können zu diesem Zeitpunkt ein Schema aus der Liste entfernen ("Aus Liste löschen").

3. Klicken Sie auf die Schaltfläche "Validieren", um alle Schemas im Listenfeld zu validieren.



In der Spalte "Validieren" sehen Sie, ob die Validierung erfolgreich war oder fehlgeschlagen ist.

Sie können nun die Schemas, bei denen ein Validierungsfehler aufgetreten ist oder eine Gruppe von ausgewählten Schemas in XMLSpy öffnen.

6.6 In Schemas suchen

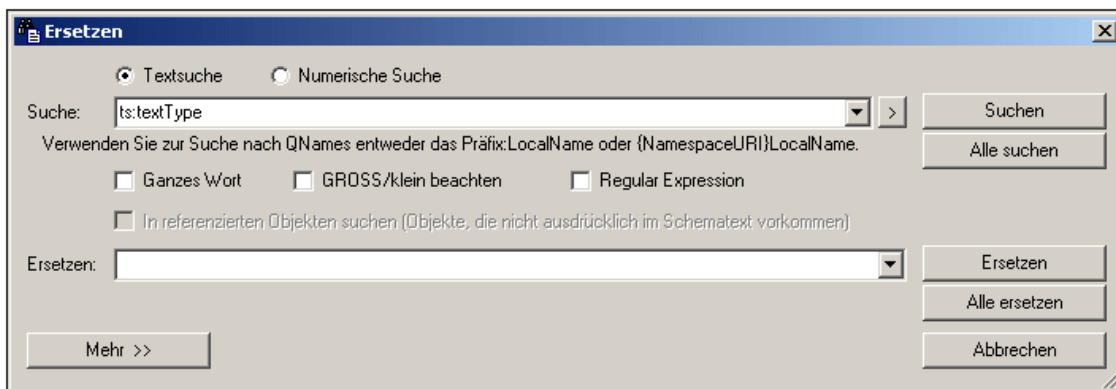
In der Schema-Ansicht können XML-Schemas mit Hilfe der XMLSpy-Funktion zum Suchen und Ersetzen in der Schema-Ansicht gesucht werden.

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:

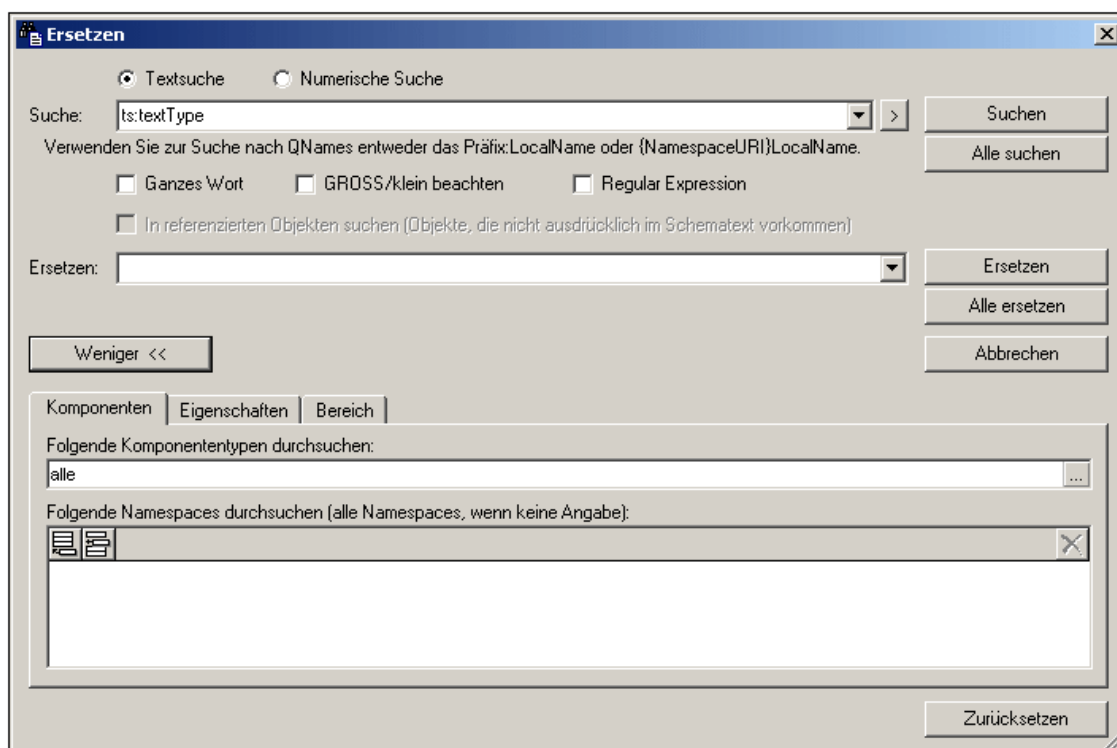
- über die Menübefehle **Bearbeiten | Suchen** und **Bearbeiten | Ersetzen**.
- über die Schaltflächen **Suchen** und **Ersetzen** im Fenster "In Schemas suchen".

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*).



Die Verwendungsweise ist die folgende:

- Geben Sie in die Textfelder "Suchen" und "Ersetzen" [den Such- und den Ersetzungsbegriff ein](#).
- Geben Sie auf dem Register "Komponenten" [die gesuchte Schemakomponente ein](#).
- [Definieren Sie die Eigenschaften der zu suchenden Komponenten](#); auf diese Art können Sie die Suche eingrenzen
- [Geben Sie den Bereich für die Suche](#) für das aktuelle Dokument oder Projekt ein oder geben Sie an, welcher Ordner durchsucht werden soll
- [Führen Sie den Befehl aus](#)
- Verwenden Sie das [Fenster "In Schemas suchen"](#), um schnell zu einer Komponente zu navigieren.

Mit der Schaltfläche **Zurücksetzen** am unteren Rand des Dialogfelds werden die folgenden Originealeinstellungen wiederhergestellt:

- Kein Suchbegriff, kein Ersetzungsbegriff
- Komponenten: `alle`
- Namespaces: keiner definiert
- Eigenschaftseinschränkungen: `überall`
- Zusätzliche Eigenschaftseinschränkungen: keine
- Bereich: aktuelle Datei

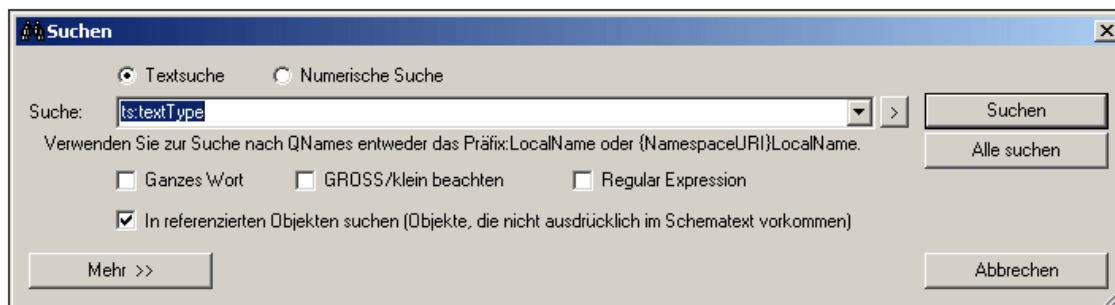
Anmerkung: Regular Expressions werden im Feld "Ersetzen" nicht unterstützt.

6.6.1 Suchbegriff

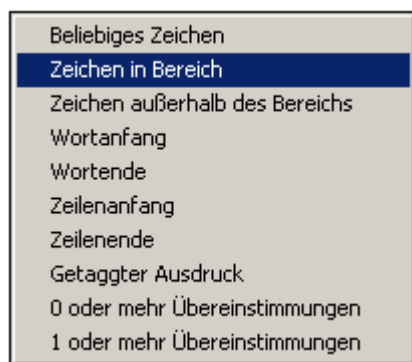
Der Suchbegriff kann als String (durch Auswahl des Optionsfelds **String**) oder als Zahl (durch Auswahl des Optionsfelds **Numerisch**) eingegeben werden.

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.

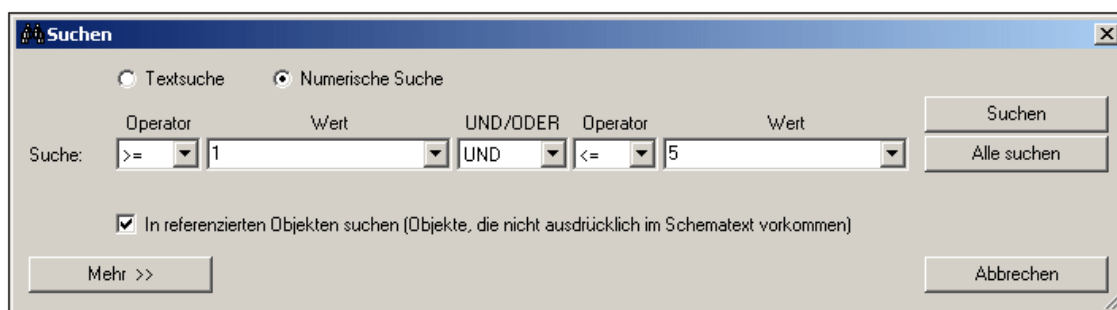


Sie können außerdem auswählen, ob ein Suchbegriff mit einem ganzen Wort im Dokument übereinstimmen muss und/oder ob die Groß- und Kleinschreibung im Dokument berücksichtigt werden soll. Verwenden Sie die Kontrollkästchen unterhalb des Texteingabefelds, um diese Optionen zu definieren.

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.

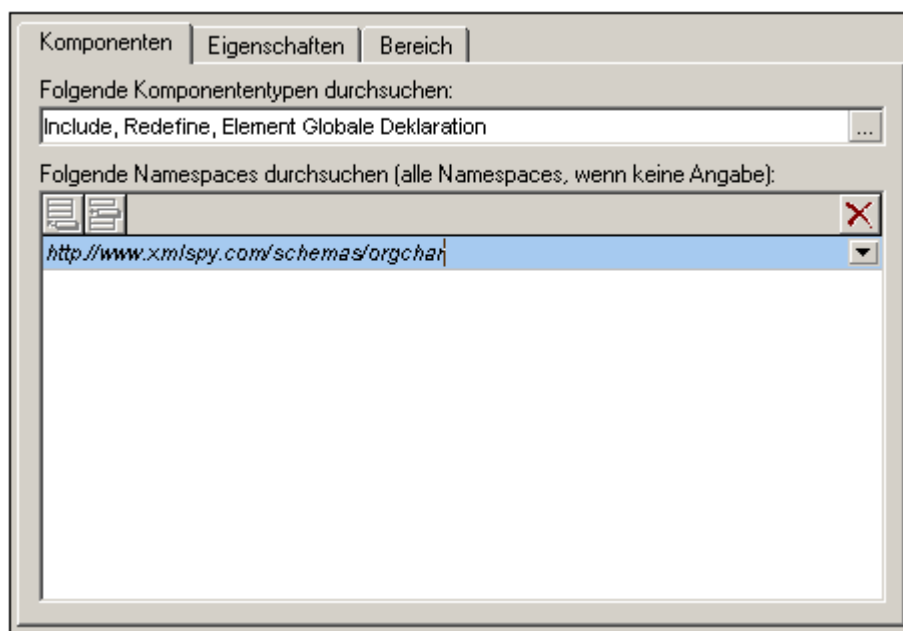
Numerische Suche

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.



6.6.2 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*).

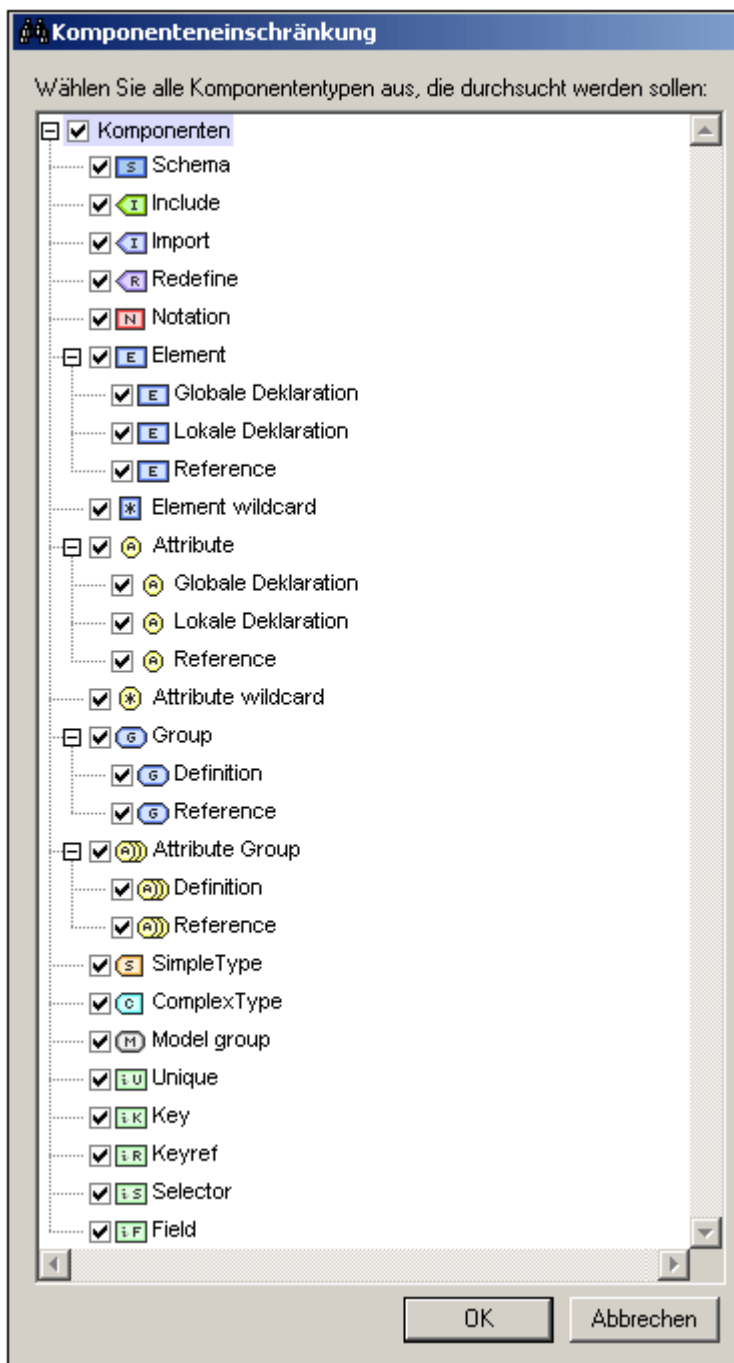


Das Register "Komponenten" besteht aus zwei Teilen: (i) einem Bereich zur Auswahl der zu durchsuchenden Komponententypen und (ii) einem Bereich zur Auswahl der zu durchsuchenden Target Namespaces.

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).



Anmerkung: Jedes Mal, wenn das Register "Komponenten" oder das Dialogfeld "Suchen/ Ersetzen" geöffnet wird, wird die vorherige Komponentenauswahl beibehalten.

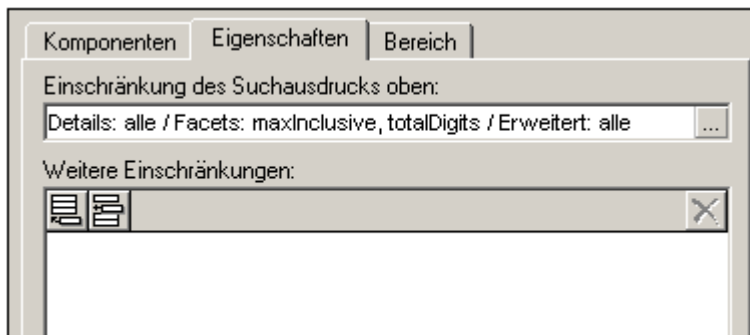
Namespace-Auswahl

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**.

6.6.3 Eigenschaften

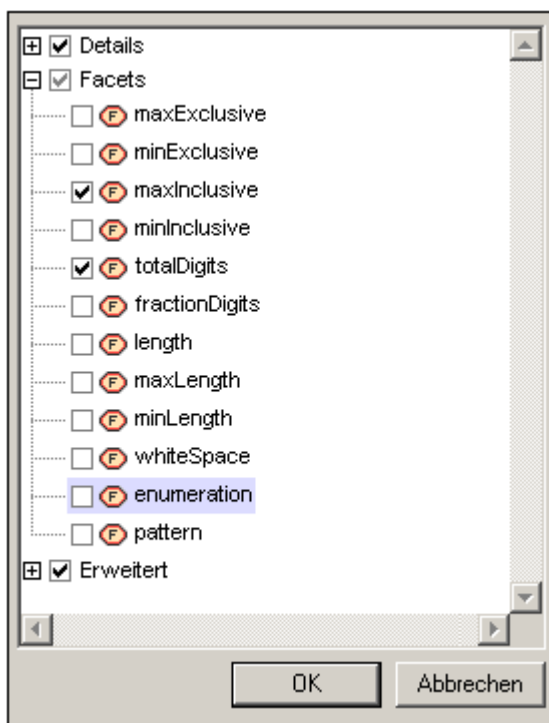
Die Suche kann mit Hilfe von Optionen vom Register "Eigenschaften" auf eine oder mehrere Komponenteneigenschaften (Details und Facets) eingeschränkt werden oder muss dem Inhalt von Eigenschaften entsprechen. Erweitern Sie das Dialogfeld "Suchen" oder "Ersetzen", indem Sie auf die Schaltfläche **Mehr** 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](#) 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.

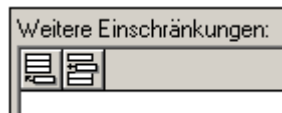


Anmerkung: Jedes Mal, wenn das Register "Eigenschaften" oder das Dialogfeld "Suchen/Ersetzen" geöffnet wird, wird die vorherige Eigenschaftsauswahl beibehalten.

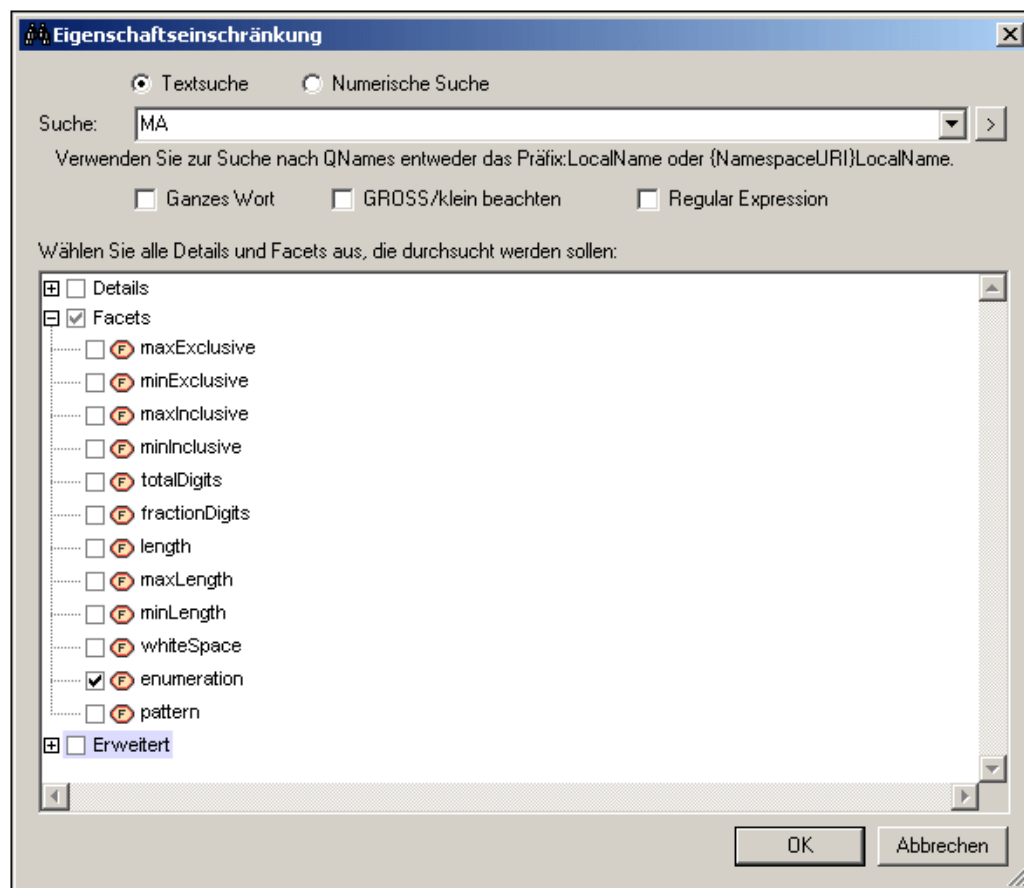
Zusätzliche Einschränkungen

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:

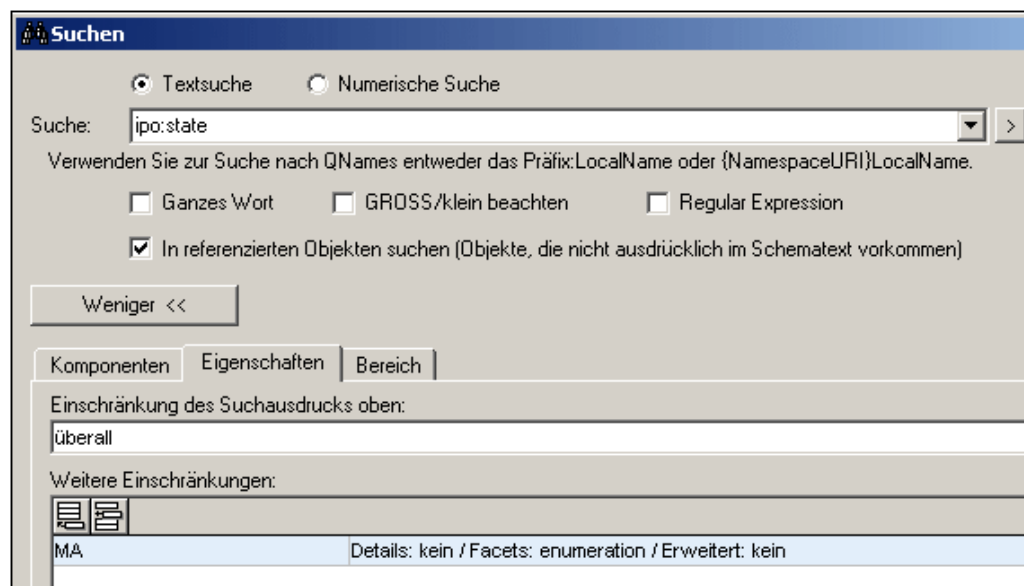
1. Klicken Sie im Fenster "Weitere Einschränkungen" auf die Schaltfläche **Hinzufügen** oder **Einfügen** (Abbildung unten).



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).



Beachten Sie in der Abbildung oben, dass der Suchbegriff `ipo:state` ist. Auf dem Register "Eigenschaften" gibt `anywhere` an, dass alle Eigenschaften durchsucht

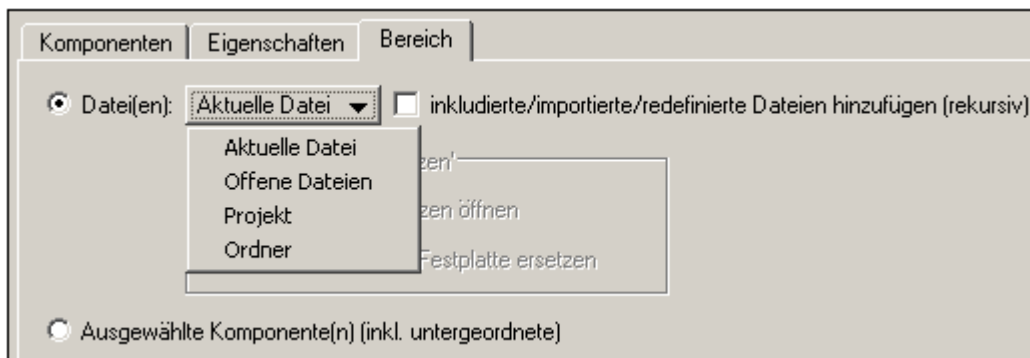
werden, doch durch die zusätzliche Einschränkung wird festgelegt, dass die Suche auf Enumerationen mit dem Wert `MA` eingeschränkt werden soll.

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.

6.6.4 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 | Dateiararten | 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 | Dateiararten | 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.

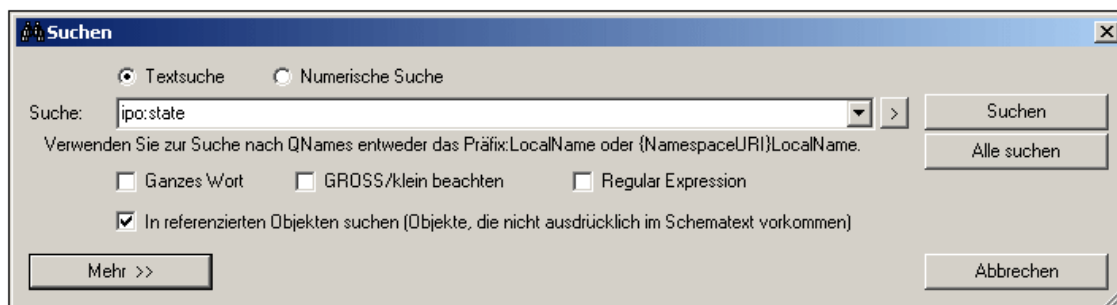
Im Dialogfeld "Ersetzen" können Sie wählen, ob der Ersetzungsbegriff in die Datei auf der Festplatte kopiert werden soll oder ob die Datei in XMLSpy geöffnet werden soll. Wählen Sie dazu im Dialogfeld die entsprechende Option aus.

6.6.5 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*).



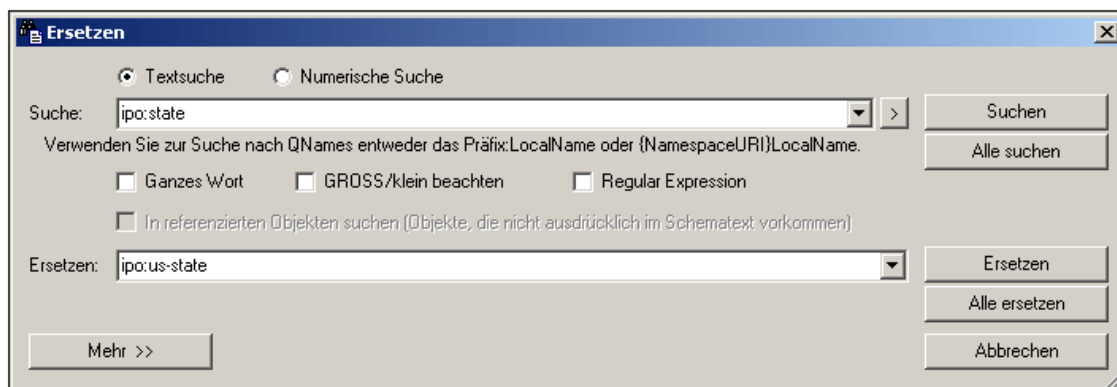
Bei Auswahl des Befehls **Suchen (Strg+F3)** wird das Dialogfeld **Suchen** geschlossen und die nächste Instanz des Suchbegriffs wird im definierten Bereich und unter Berücksichtigung der Suchkriterien gesucht. Die nächste Instanz wird relativ zur aktuell ausgewählten Komponente in der Schema-Ansicht 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.



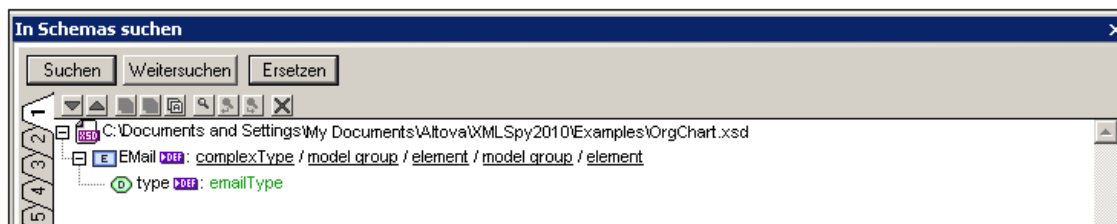
Mit dem Befehl **Alle suchen** wird das Dialogfeld "Ersetzen" geschlossen und alle Suchergebnisse werden im Fenster "In Schemas suchen" aufgelistet.

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.

6.6.6 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 **Suchen** 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:

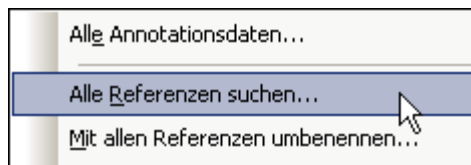
- Mit den Schaltflächen **Nächstes** und **Vorheriges** wird die im Verhältnis zur aktuell ausgewählten Meldung jeweils nächste bzw. vorherige ausgewählt
- Mit dem Befehlen zum **Kopieren** von Meldungen werden die ausgewählte Meldung, die ausgewählte Meldung und ihre untergeordneten Einträge bzw. alle 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.

6.6.7 Suchen und Umbenennen von globalen Komponenten

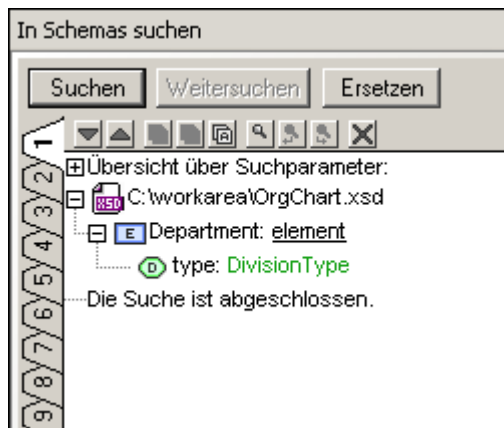
Benannte globale Komponente von XML-Schemas können in einer ausgewählten Datei und in allen Schema-Dateien, die mit der ausgewählten Datei in einer Beziehung stehen, gesucht und umbenannt werden. Benannte globale Komponenten sind alle globalen Komponenten mit Ausnahme von Include, Import, Redefine, Annotation, 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.



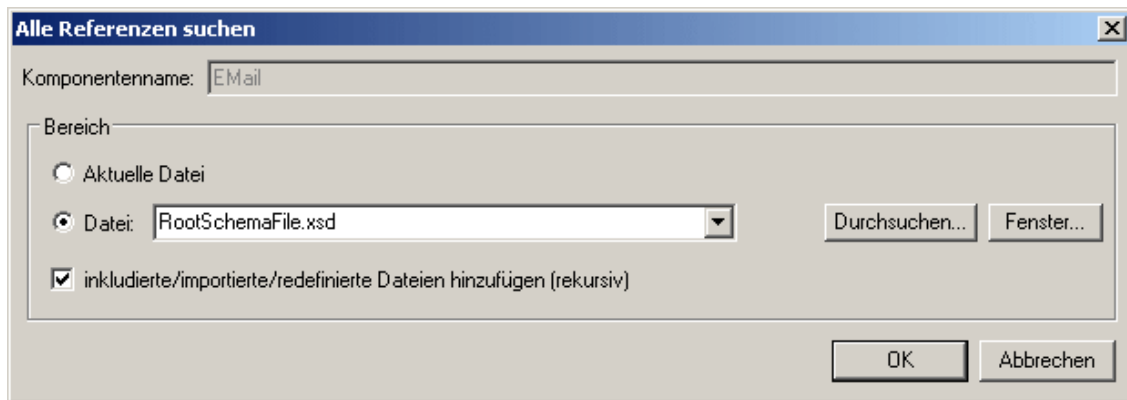
3. Wählen Sie im Dialogfeld, das erscheint, den Bereich der Suche (bzw. der Umbenennung). Geben Sie im Fall einer Umbenennung den neuen Namen der globalen Komponente ein.
4. Nachdem Sie auf **OK** geklickt haben, werden im Fenster "In Schemas suchen" die Suchergebnisse angezeigt (*Abbildung unten*).



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

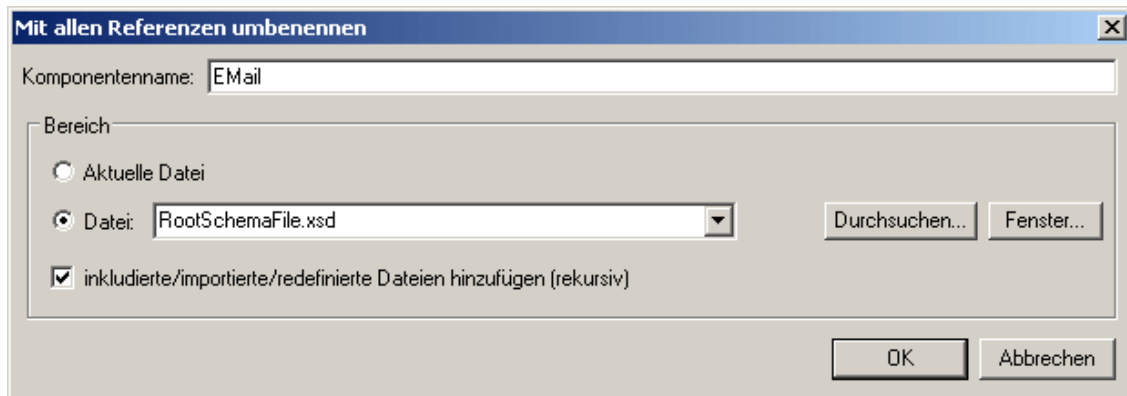
Um das Dialogfeld "Alle Referenzen suchen" (*Abbildung unten*) zu öffnen folgen Sie folgendem Ablauf: (i) Klicken Sie in der Schema-Übersicht mit der rechten Maustaste auf die globale Komponente, (ii) wählen Sie den Befehl **Alle Referenzen suchen** aus dem Kontextmenü, das erscheint.



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 *inkludierte/importierte/redefinierte Dateien hinzufügen (rekursiv)* in 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 *inkludierte/importierte/redefinierte Dateien hinzufügen (rekursiv)* in unteren Teil des Dialogfelds aktivieren.

7 XSLT und XQuery

XMLSpy bietet Funktionen zur Bearbeitung von XSLT- und XQuery-Dokumenten, verfügt über integrierte Prozessoren zum Transformieren mittels XSLT und zum Ausführen von XQuery-Dokumenten sowie leistungsstarke Debugger. Dieser Abschnitt bietet einen Überblick über die XSLT- und XQuery-Funktionalitäten in XMLSpy. Er ist in die folgenden Abschnitte gegliedert:

- [XSLT](#)
- [XQuery](#)
- XSLT und XQuery Debugger

Weitere Einzelheiten zu den Befehlen finden Sie im Abschnitt "Benutzerreferenz" unter der Beschreibung zum jeweiligen [Menübefehl](#). Nähere Informationen zu Bearbeitungsfunktionen finden Sie auch in den Abschnitten [Bearbeitungsansichten](#) und [XML](#).

7.1 XSLT

Altova Website:  [XSLT Editor](#)

Dieser Abschnitt über XSLT ist in die folgenden Unterabschnitte gegliedert:

- [Bearbeiten von XSLT-Dokumenten](#): Beschreibung der Bearbeitungsfunktionen für XSLT-Dokumente in XMLSpy
- [XSLT-Verarbeitung](#): 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](#): 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"](#) 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" angezeigt. Eine Beschreibung zur Verwendung des Fensters "XPath" finden Sie im Abschnitt [Einführung | XPath-Fenster](#)

XSLT

Weitere XSLT-Funktionen

Weitere detailliertere Beschreibungen zu den hier beschriebenen Features finden Sie in der Benutzerreferenz in den Beschreibungen zu den [entsprechenden Menübefehlen](#).

Altova XSLT-Prozessoren

Nähere Informationen zur Implementierung des Altova XSLT 1.0 und 2.0-Prozessors finden Sie [in den Anhängen mit Informationen zu den Prozessoren](#).

AltovaXML 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 kostenlose Altova-Produkt [AltovaXML](#), welches den Altova XSLT 1.0- und 2.0-Prozessor enthält, die auch in XMLSpy integriert sind. AltovaXML eignet sich ideal für XSLT-Transformationen oder Batch-Verarbeitung über die Befehlszeile.

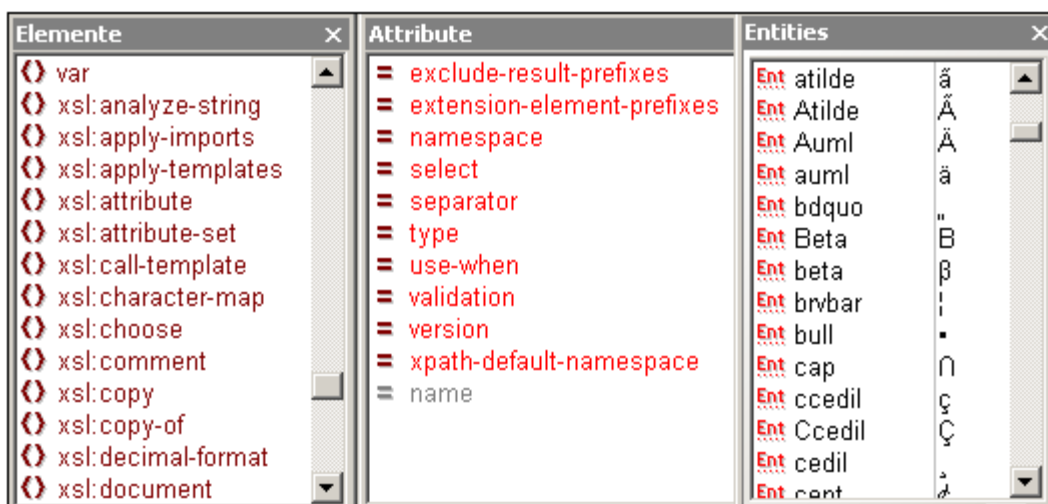
7.1.1 XSLT-Dokumente

XSLT 1.0- und 2.0-Dokumente können wie jedes andere XML-Dokument in der [Textansicht](#) und der [Grid-Ansicht](#) bearbeitet werden. Die Standardansicht, in der ein XSLT-Dokument geöffnet wird, kann im Dialogfeld "Optionen" auf dem Register "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 oder XSLT 2.0, 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 eingegebenen 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. der Wert eines `select` Attributs). Während der Eingabe stehen Ihnen XPath-Funktionen und XPath-Achsen in Popup-Fenstern zur Verfügung. Wenn außerdem im [Info-Fenster](#) eine XML-Datei zugewiesen wurde, stehen auch die Elemente und Attribute der XML-Datei im Popup-Fenster 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 oder XSLT 2.0-Dokument handelt (dies ist im `version` Attribut des `xsl:stylesheet` Elements definiert).

7.1.2 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" auf dem Register XSL](#) (*Abbildung unten*) definiert.

☒ Integrierter XSLT-Prozessor (wird immer für das XSLT Debuggen verwendet!)

☐ Microsoft® XML Parser (MSXML): ☐ v3.0 ☐ v4.0 ☐ v6.0 ☒ Version automatisch wählen

☐ Externes XSL-Transformationsprogramm:

Geben Sie die Befehlszeilen-Parameter für das externe XSL-Transformationsprogramm in folgender Form ein:

Program.exe %1 %2 %3

%1 wird durch den XML Eingabe-Dateinamen ersetzt, %2 durch den Ausgabe-Dateinamen und %3 (optional) durch den Namen der XSL Stylesheet-Datei. Sie können auch weitere notwendige Parameter anhängen.

☒ Ausgabe des externen Programms nach der Transformation anzeigen

☒ Ausgabefehler des externen Programms nach Transformation anzeigen

Standard-Dateierweiterung der Ausgabedatei: ☒ Ausgabefenster wiederverwenden

☒ Dateierweiterung aus `<xsl:output method="">` Attribut verwenden, falls verfügbar

Pfad zum XSL-FO Transformationsprozessor eingeben (bei Verwendung von FOP bitte Pfad zu fop.bat eingeben):

☒ Oben ausgewählten XSLT-Prozessor zur Durchführung des XSLT-Teils und anschließend XSL-FO-Prozessor für FO-Teil verwenden

☐ XSL-FO-Prozessor für XSLT- und FO-Teil der Transformation verwenden

Die Optionen, die zur Verfügung stehen, werden im Abschnitt [Benutzerreferenz](#) erläutert. Der Prozessor, der auf dem Register "XSL" definiert ist, wird für alle XSLT-Transformationen verwendet. 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"](#) (*Abbildung oben*) definiert.

XSLT 1.0 und 2.0 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- und den XSLT 2.0-Prozessor von Altova. Welcher Prozessor ausgewählt wird, hängt vom Wert des Versionsattributs (1.0 oder 2.0) 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](#). 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.

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](#). 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"](#) finden Sie im [Abschnitt "Benutzerreferenz"](#).

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](#).

Batch-Verarbeitung mit AltovaXML

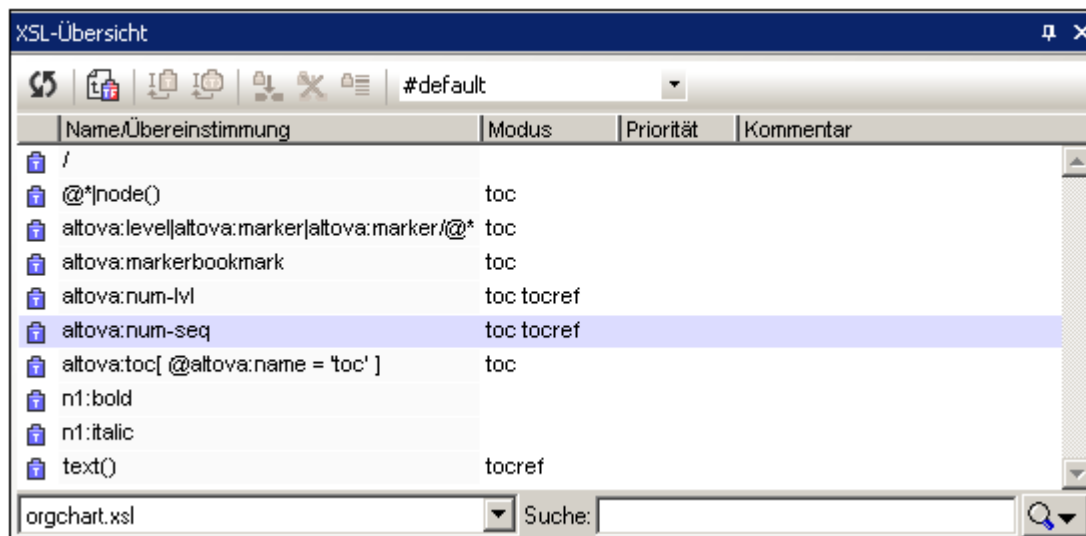
AltovaXML ist eine kostenlose eigenständige Applikation, die den XML-Validator, die XSLT-Prozessoren und den XQuery-Prozessor von Altova enthält. Sie kann über die Befehlszeile, eine COM-Schnittstelle, in Java-Programmen und in .NET-Applikationen verwendet werden, um XML-Dokumente zu validieren, XML-Dokumente mit XSLT 1.0 und 2.0 Stylesheets zu transformieren und XQuery-Dokumente auszuführen.

Mit Hilfe von AltovaXML können daher XSLT-Transformationen automatisiert werden. So können Sie z.B. eine Batch-Datei erstellen, die AltovaXML aufruft, um eine Gruppe von Dokumenten zu transformieren. Nähere Informationen dazu finden Sie in der [Dokumentation zu AltovaXML](#).

7.1.3 XSL-Übersicht

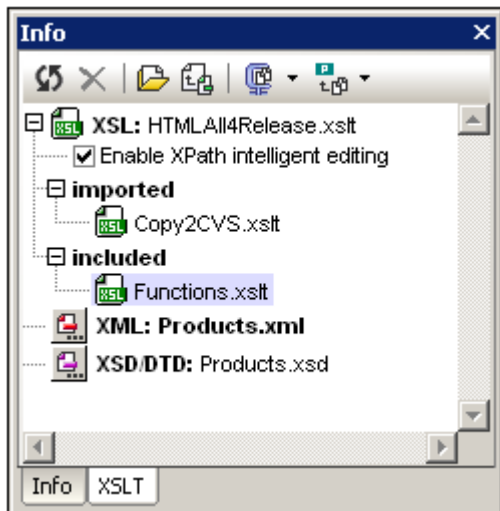
Wenn ein XSLT-Dokument in XMLSpy das aktive Dokument ist, werden Informationen über die Struktur des Dokuments im [Fenster "XSL-Übersicht"](#) und Informationen über die mit dem aktiven XSLT-Dokument in Zusammenhang stehenden Dateien auf dem [Register XSLT des Info-Fensters](#) (welches nur angezeigt wird, wenn es sich beim aktiven Dokument in XMLSpy um ein XSLT-Dokument handelt) angezeigt. Der 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"](#) (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

Auf dem [Register "XSLT" des Info-Fensters](#) (Abbildung unten) haben Sie folgende Möglichkeiten:

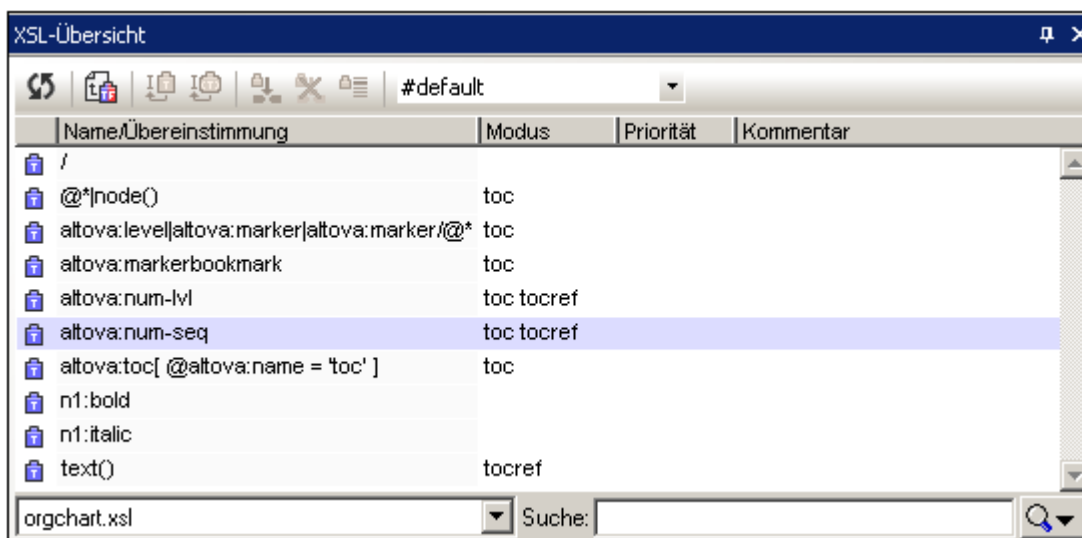


- Anzeigen von Informationen zu allen Dateien, die in Zusammenhang mit dem aktiven XSLT-Dokument stehen, - z.B. der Pfad der importierten und inkludierten Dateien
- Definieren einer XML-Datei für die Transformation mit dem aktiven XSLT-Dokument. Außerdem kann die Schema-Datei (XSD/DTD) zur Validierung 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"](#) und das [Register "XSLT" des Info-Fensters](#) werden in den Unterabschnitten dieses Abschnitts näher beschrieben.

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:

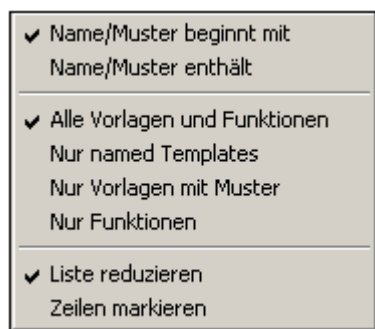
- der Name (wenn die Vorlagen ein `name` Attribut hat) und/oder der XPath-Ausdruck (wenn die Vorlage ein `match` Attribut hat). Wenn die Vorlage sowohl ein `name` als auch ein `match` Attribut hat, werden beide aufgelistet, wobei der Wert des Attributs `name` zuerst angeführt wird: `namevalue, matchvalue`.
- der Modus, falls definiert;
- 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.



- **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 bestimmten 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 **Neu laden** , 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 Eintrag** , 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.

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:

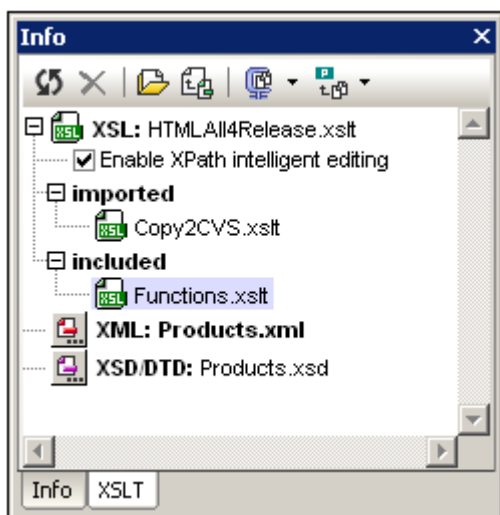
- **`xsl:call-template` einfügen:** Dieser Befehl wird aktiv, wenn im Fenster "XSL-Übersicht" ein Named Template ausgewählt wird. Der Befehl fügt ein `xsl:call-template`

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.

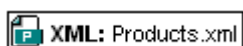
Das Fenster "Info"

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 eine XML-Datei auswählen, die als XML-Quelldokument herangezogen wird, an dem das XSLT-Dokument für die Transformation angewendet wird.

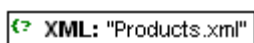


Auf dem Register "XSLT" des Info-Fensters werden die folgenden Dateien angezeigt:

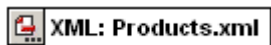
- **XSLT-Dateien:** Alle importierten und inkludierten XSLT-Dateien werden aufgelistet (siehe Abbildung oben). Der Pfad der einzelnen Dateien wird in einem Popup-Fenster angezeigt, wenn der Maus-Cursor über die Datei platziert wird. Wenn Sie auf eine importierte oder inkludierte Datei doppelklicken oder 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" (*XSL-Transformation von XSL-Dateien*) 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](#) 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:

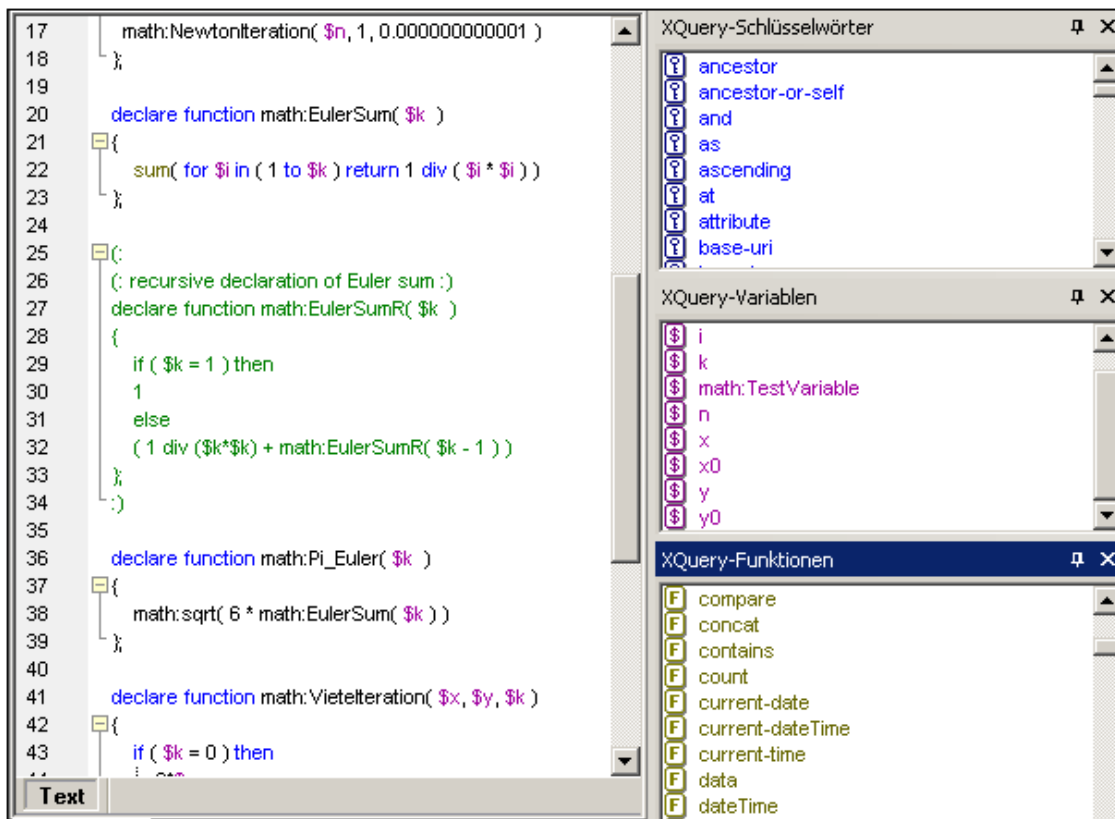


- *Info neu laden:* Aktualisiert das Info-Fenster, damit die im XSLT-Dokument vorgenommenen Änderungen berücksichtigt werden.
- *XML/XSD-Zuweisung löschen:* Löscht die vom Benutzer über die Schaltflächen XML bzw. XSD/DTD vorgenommene Zuweisung. Wählen Sie die zu löschende Datei aus und klicken Sie auf 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 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.

7.2 XQuery

Altova Website:  [XQuery Editor](#)

XQuery-Dokumente können in der Textansicht bearbeitet werden. Die Eingabehilfen, Syntaxfärbung und das intelligente Editieren unterscheiden sich dabei von denen, die für die Textansicht von XML-Dokumenten zur Verfügung stehen (*siehe Abbildung unten, Zeilennummerierung und Klappleisten stehen nur in der Enterprise und der Professional Edition zur Verfügung*). Dieser Textansichtsmodus wird XQuery-Modus genannt. Zusätzlich dazu können Sie Ihr XQuery-Dokument in der Textansicht validieren und den Code in einem XQuery-Dokument (mit einer optionalen XML-Datei, wenn nötig) mit Hilfe des Altova XQuery-Prozessors ausführen.



Hinweis: XQuery-Dateien können nur in der Textansicht bearbeitet werden. Für XQuery-Dateien stehen keine anderen Ansichten zur Verfügung.

XQuery Profiler und Debugger

XMLSpy enthält außerdem einen XQuery Profiler und [XQuery Debugger](#), mit denen Sie schneller korrekte XQuery-Dokumente erstellen können. Diese beiden Funktionen werden im Abschnitt [XSLT und XQuery](#) beschrieben.

Nähere Informationen dazu, wie der Altova-XQuery-Prozessor implementiert ist und XQuery-Dateien verarbeitet, finden Sie unter "[Implementierung des XQuery-Prozessors](#)".

AltovaXML 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 kostenlose Altova-Produkt AltovaXML](#) ausprobieren, das den in XMLSpy integrierten XQuery-Prozessor enthält. AltovaXML ist ideal für die Ausführung von XQuery oder die Batch-Verarbeitung über die Befehlszeile geeignet.

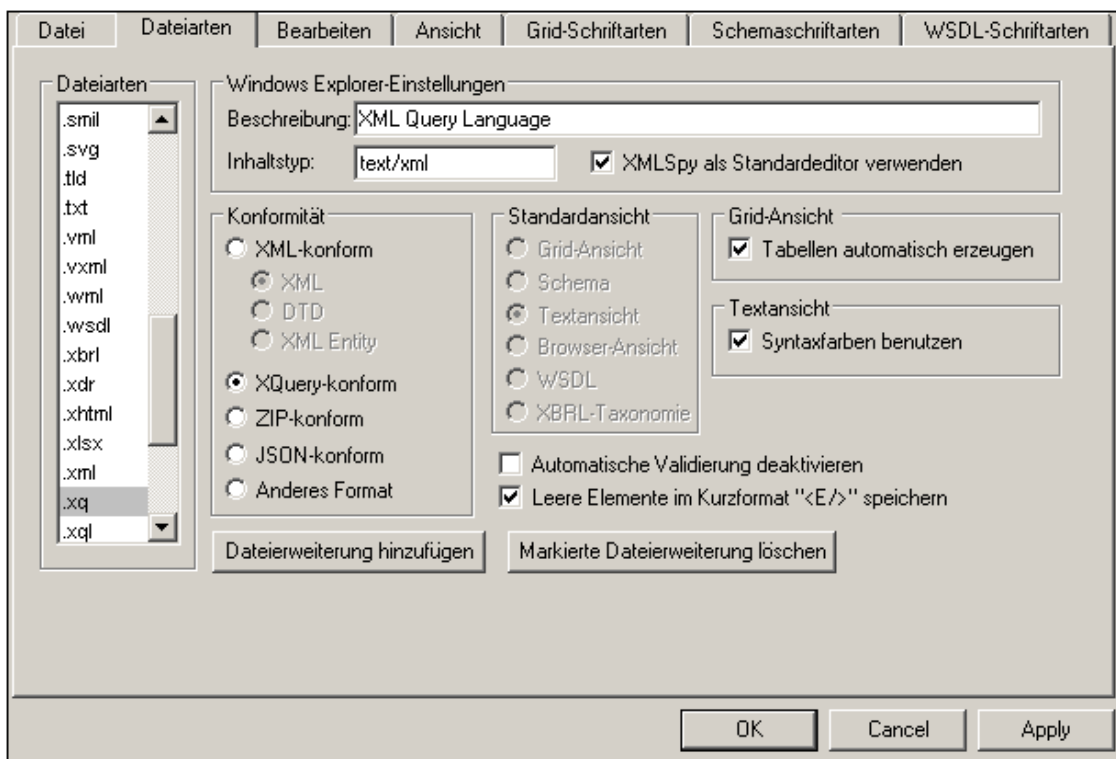
7.2.1 XQuery-Dokumente

Ein XQuery-Dokument wird automatisch im XQuery-Modus der Textansicht geöffnet, wenn es XQuery-konform ist. Dateien mit der Dateierweiterung `.xq`, `.xql`, und `.xquery` sind in XMLSpy als XQuery-konform vordefiniert.

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 *screenshot*).
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**.

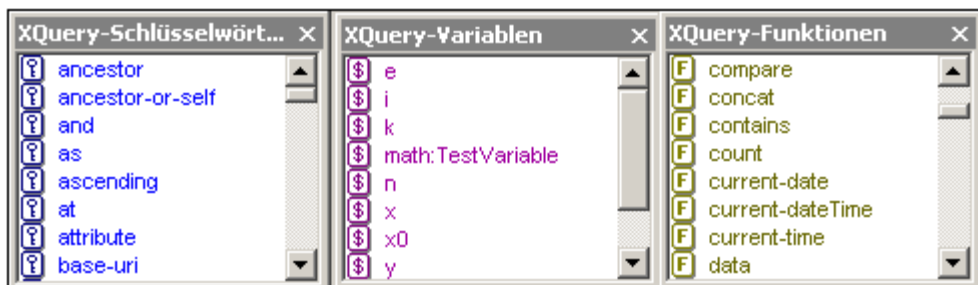


Außerdem sollten Sie in diesem Dialogfeld die folgenden Windows Explorer-Einstellungen vornehmen:

- **Beschreibung:** XML Query Language
- **Inhaltstyp:** Text/xml
- Wenn Sie XMLSpy als Standardeditor für XQuery-Dateien verwenden wollen, aktivieren Sie das Kontrollkästchen **XMLSpy als Standardeditor verwenden**.

7.2.2 XQuery Eingabehilfen

In der Textansicht stehen Ihnen Im XQuery-Modus drei Eingabehilfen zur Verfügung: XQuery-Schlüsselwörter (blau), XQuery-Variablen (violett) und XQuery-Funktionen (olivgrün).



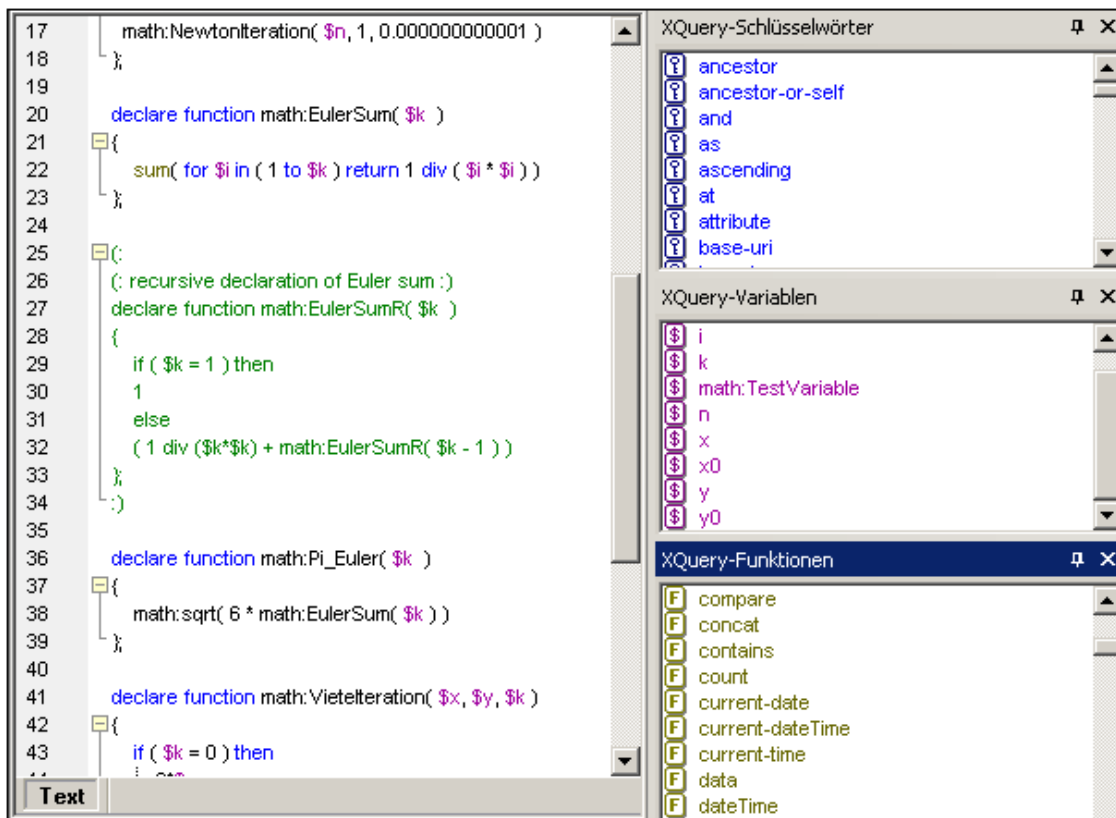
Beachten Sie bitte die folgenden Punkte:

- 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 vom Altova XQuery-Prozessor 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, Mausekursor 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. In XQuery stellen einige Zeichenfolgen sowohl ein Schlüsselwort als auch eine Funktion dar (`empty`, `unordered` und `except`). Diese Strings werden immer als Schlüsselwörter eingegeben (blau) - selbst wenn Sie die Funktion dieses Namens in der Funktionen-Eingabehilfe auswählen. Wenn eine Funktion blau angezeigt wird, ist sie durch die Klammern, die auf den Funktionsnamen folgen, als solche zu erkennen.

7.2.3 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**) auf dem Register "Textschriftarten" 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 grün :)
- XQuery-Schlüsselwörter sind blau: **Schlüsselwort**
- XQuery-Variablen, einschließlich des Dollar-Zeichens sind violett: **\$start**
- XQuery-Funktionen, **nicht** aber die einschließenden Klammern sind olivgrün: **Funktion()**
- 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 Schriftarteneinstellungen im Dialogfeld "Optionen" (**Extras | Optionen**) auf dem Register "Textschriftarten" ändern.

Hinweis: Im oben gezeigten Screenshot wird ein farbiges Klammerpaar für einen Comment schwarz und fett angezeigt. Dies geschieht aufgrund der Funktion zum Anzeigen von Kammerpaaren (siehe [Intelligentes Editieren von XQuery](#)).

7.2.4 Intelligentes Editieren von XQuery

Im XQuery-Modus der Textansicht stehen Ihnen die folgenden intelligenten Editierfeatures zur Verfügung.

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.

```
1  (: (: (Filename: seqQ5b.xq :))
2  (: Source: http://www.w3.org/TR/xquery-use-cases :):)|
```

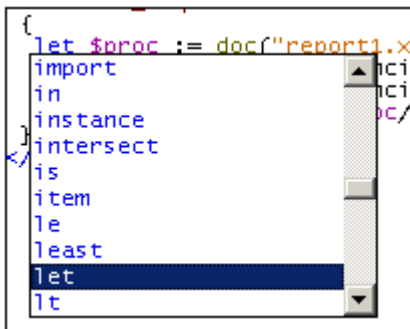
Die Klammerpaaranzeige wird aktiviert, indem Sie den Cursor entweder unmittelbar vor oder unmittelbar hinter eine (öffnende oder schließende) Klammer setzen. Diese Klammer wird zusammen mit ihrem Gegenstück (schwarz und fett) hervorgehoben. Beachten Sie die Position des Cursors im obigen Screenshot.

Die Klammerpaaranzeige ist für runde () , eckige [] und geschwungene Klammern { } aktiv. Eine Ausnahme bilden spitze Klammern <>, die für XML-Tags verwendet werden.

Hinweis: 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.

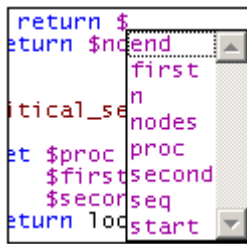


In der oben gezeigten Abbildung wurde der Cursor in das Schlüsselwort `let` gesetzt. Wenn Sie auf ein Schlüsselwort aus der Liste doppelklicken, wird das Schlüsselwort `let` durch dieses ersetzt.

Variablen

Variablenamen 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.)



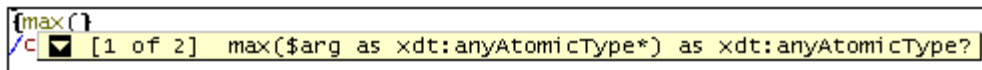
Um eine Variable (während der Eingabe) hinter dem \$-Zeichen einzufügen oder eine ausgewählte Variable zu ersetzen, doppelklicken Sie im Popup-Menü auf die gewünschte Variable.

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**.

Bitte beachten Sie: Die Bezeichnung kann nur für XQuery-Standardfunktionen angezeigt werden.



Der nach unten weisende Pfeil bedeutet, dass es mehrere Funktionen mit demselben Namen gibt, dass diese jedoch unterschiedliche Argumente oder Rückgabetypen haben. Klicken Sie auf die Bezeichnung 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.

7.2.5 Validierung und Ausführung von XQuery-Dokumenten

Validierung von XQuery-Dokumenten

Um ein XQuery-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 .

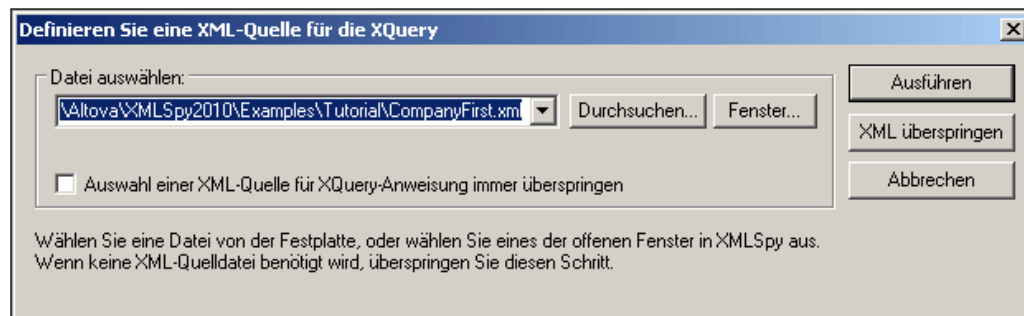
Das Dokument wird daraufhin auf die korrekte XQuery-Syntax überprüft.

Ausführen von XQuery-Dokumenten

XQuery-Dokumente werden in XMLSpy mit dem integrierten XQuery 1.0-Prozessor ausgeführt. Das Resultat wird in einem Fenster in XMLSpy angezeigt.

Normalerweise ist ein XQuery-Dokument nicht mit einem einzelnen XML-Dokument verknüpft, da XQuery-Ausdrücke mit Hilfe der `doc()`-Funktion eine beliebige Anzahl von XML-Dokumenten auswählen können. In XMLSpy können Sie jedoch vor dem Ausführen eines einzelnen XQuery-Dokuments ein XML-Quelldokument dafür auswählen. In diesem Fall ist der Dokument-Node des ausgewählten XML-Quelldokuments der Context-Node, der auf der Root-Ebene des XQuery-Dokuments als Ausgangspunkt angegeben ist. Pfade, die mit einem Schrägstrich beginnen, werden mit diesem Dokument-Node als Context-Element aufgelöst. Um ein XQuery-Dokument auszuführen, gehen Sie vor wie folgt:

1. Machen Sie das XQuery-Dokument zum aktiven Dokument.
2. Wählen Sie **XSL/XQuery | XQuery-Ausführung** oder klicken Sie in der Symbolleiste auf das Symbol . Daraufhin erscheint das Dialogfeld "Definieren Sie eine XML-Quelle für die XQuery".



3. Wählen Sie eine der folgenden Methoden:
 - Zum Auswählen einer XML-Datei verwenden Sie entweder die Schaltfläche **Durchsuchen** oder die Schaltfläche **Fenster** (damit werden Dateien, die in XMLSpy geöffnet sind und jene die sich in XMLSpy-Projekten befinden, aufgelistet). Wählen Sie eine XML-Quelldatei aus, wenn Sie deren Dokument-Node als Kontextelement für die Root-Ebene des XQuery-Dokuments zuweisen möchten. Klicken Sie auf **Ausführen**.
 - Um dieses Dialogfeld zu überspringen, klicken Sie auf **XML überspringen**.

Das Ergebnisdokument wird als temporäre Datei generiert, die im gewünschten Dateiformat und mit der gewünschten Erweiterung in einem beliebigen Ordner gespeichert werden kann.

XQuery-Variablen

Wenn Sie den Altova XQuery-Prozessor 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](#).

Nähere Informationen darüber, wie der Altova XQuery-Prozessor implementiert ist und XQuery-Dateien verarbeitet, finden Sie unter [Implementierung des XQuery-Prozessors](#).

7.2.6 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](#) 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](#) und stellen Sie (im Dialogfeld "Verbindungsassistent") [eine Verbindung zur DB her](#).
3. Fügen Sie im XQuery-Dokument [DB-spezifische XQuery-Erweiterungen ein](#), um die DB-Daten aufrufen zu können und für XQuery-Operationen verfügbar zu machen.
4. [Führen Sie das XQuery-Dokument](#) 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>
```

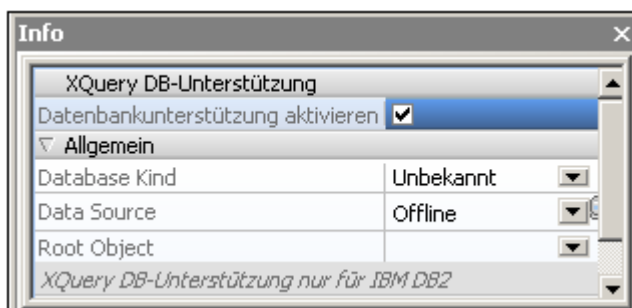
Wenn im Dokument der optionale Ausdruck `xquery version` verwendet wird, wird dennoch das Schlüsselwort `XQUERY` benötigt.

```
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>
```

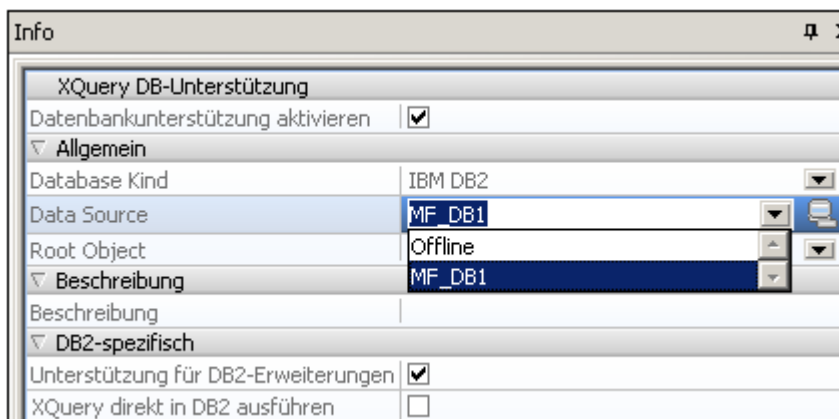
Anmerkung: Der integrierte XMLSpy XQuery-Prozessor liest das Schlüsselwort `XQUERY` als Indikator dafür, dass eine XML DB aufgerufen werden soll. Folglich führt der Versuch, ein XQuery-Dokument, das das Schlüsselwort `XQUERY` enthält, an einem XML-Dokument auszuführen, das nicht in einer XML DB enthalten 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 Datenquelle](#) 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.*

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.

7.3 XSLT und XQuery Debugger

Altova Website:  [XSLT Debugger](#), [XQuery Debugger](#)

Der XSLT und XQuery Debugger dient zum Testen und Debuggen von XSLT Stylesheets und XQuery-Dokumenten. Über die Oberfläche des XSLT und 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 und XQuery Debugger wird immer während einer **Debugger-Sitzung** geöffnet. Es gibt drei 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
- XQuery, wobei der integrierte Altova XQuery 1.0-Prozessor verwendet wird

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 und "2.0" für XSLT 2.0).). XQuery-Debugger-Sitzungen werden von XQuery-Dateien aus 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- oder ein XSLT 2.0-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)
XQuery	XML; (optional)	XQuery (mit dem integrierten Altova XQuery-Prozessor)
XML	XSLT 1.0, oder XSLT 2.0, oder XQuery; (erforderlich)	XSLT 1.0, XSLT 2.0 oder XQuery. XSLT 1.0 oder 2.0 je nach Wert des Attributs <code>version</code> des Elements <code>xsl:stylesheet</code> (oder <code>xsl:transform</code>) im XSLT Stylesheet.

Nähere Informationen zu den drei Altova-Prozessoren finden Sie in den folgenden Abschnitten:

- [Altova XSLT 1.0-Prozessor](#)
- [Altova XSLT 2.0-Prozessor](#)
- [Altova XQuery 1.0-Prozessor](#)

Automatisieren von XSLT- und XQuery-Aufgaben mit AltovaXML 2012

AltovaXML ist eine kostenlose Applikation, die den AltovaXML-Validator sowie einen XLST

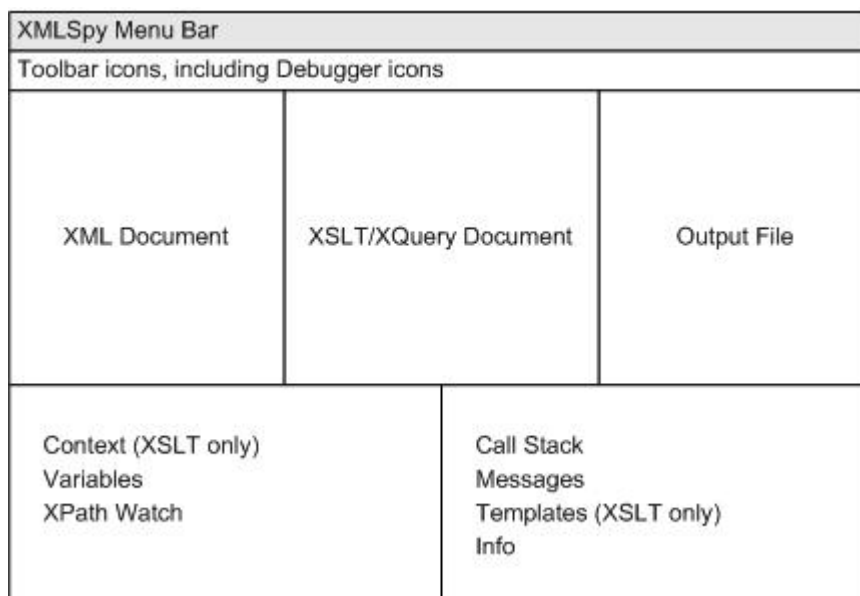
1.0-, XSLT 2.0- und XQuery 1.0-Prozessor enthält. Das Tool kann über die Befehlszeile, über eine COM-Schnittstelle, in Java-Programmen und in .NET-Applikationen zur Validierung von XML-Dokumenten, zum Transformieren von XML-Dokumenten unter Verwendung von XSLT 1.0 und 2.0 Stylesheets sowie zum Ausführen von XQuery-Dokumenten verwendet werden.

XSLT- und XQuery-Aufgaben können daher mittels AltovaXML automatisiert werden. So können Sie z.B. eine Batch-Datei erstellen, die AltovaXML aufruft, um damit eine Reihe von XML-Dokumenten zu transformieren oder eine Reihe von XQuery-Dokumenten auszuführen. Nähere Informationen dazu finden Sie in der [AltovaXML Dokumentation](#).

7.3.1 Funktionsweise und Benutzeroberfläche

Die grundlegende Vorgangsweise beim Debuggen von XSLT- und XQuery-Dateien mit XSLT und 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 oder XQuery) wird anhand der aktiven Datei ausgewählt (siehe [XSLT und XQuery Debugger](#)). Der XSLT und 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 und XQuery Debugger gewechselt werden darf. Sie können diese Option im [Dialogfeld "Debugger-Einstellungen"](#) 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 Watch, Call Stack, Messages, Info, usw.).
- Sie können den Debugger während einer Debugger-Sitzung anhalten (dies ist nicht dasselbe wie die Beendigung 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 und 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 und 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.)

Bitte beachten Sie: Die Debugger-Symbolleiste und -Symbole werden automatisch angezeigt, wenn eine Debugger-Sitzung gestartet wird.

7.3.2 Befehle und Symbolleistenflächen

Die Debugger-Befehle stehen im Menü **XSL/XQuery** und als Symbolleistenflächen 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 Debugger 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 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.



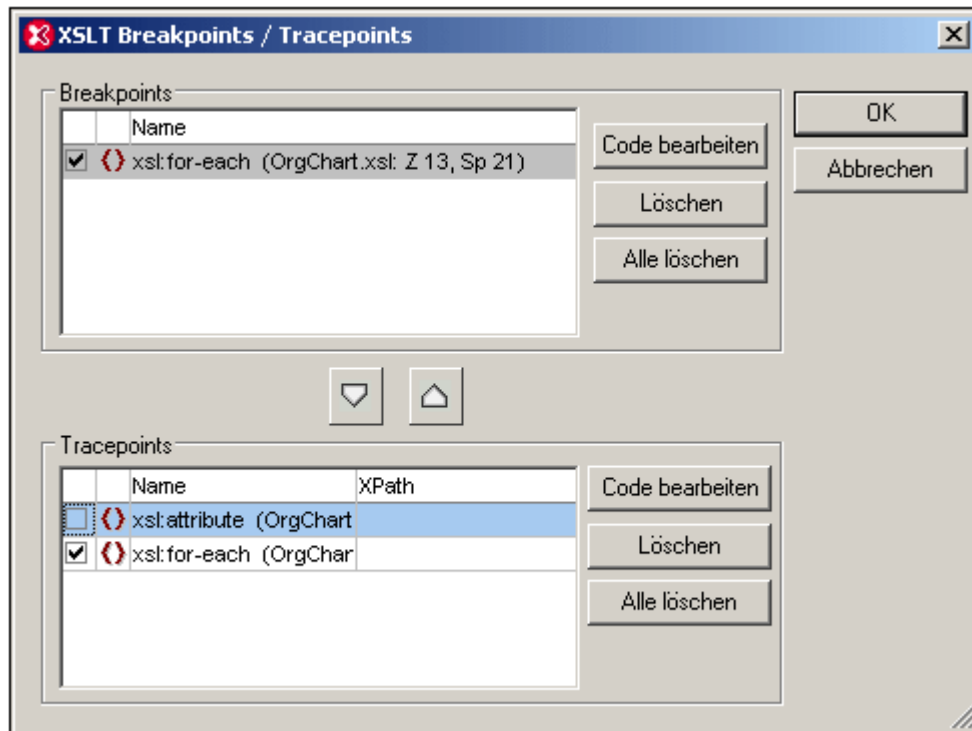
Debugger-Sitzung beenden

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](#) 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.



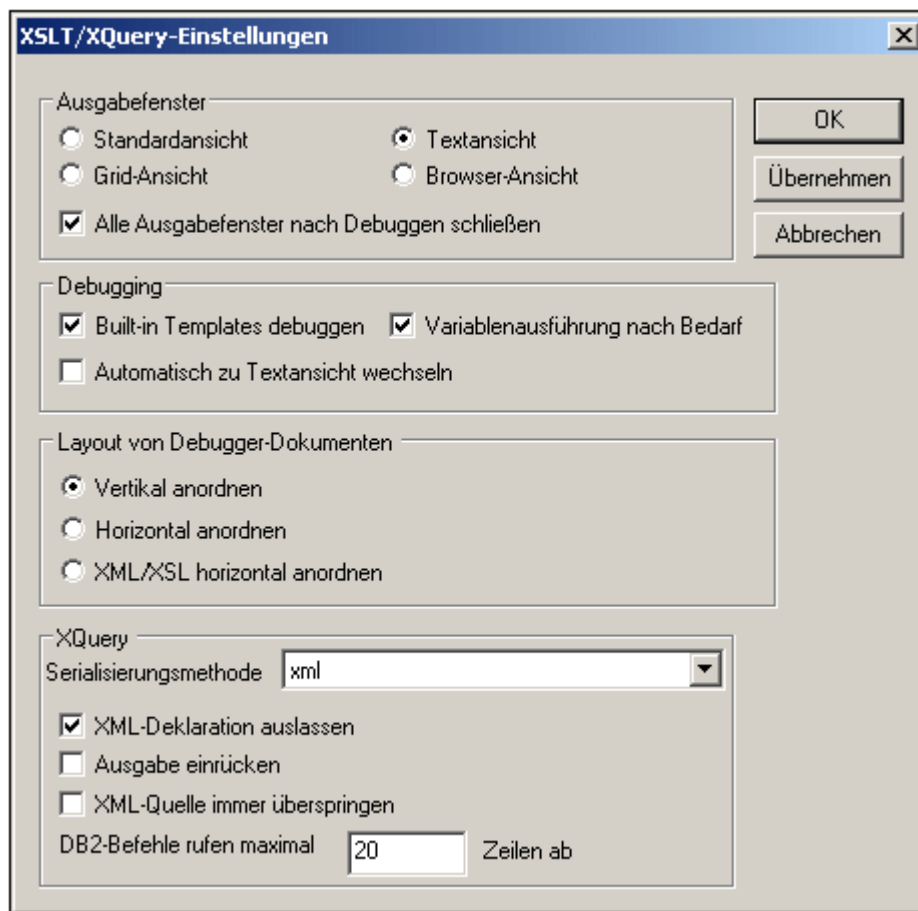
Die Kontrollkästchen zeigen an, ob ein Breakpoint/Tracepoint aktiviert (mit Häkchen versehen) oder deaktiviert ist. Sie können den markierten Breakpoint/Tracepoint durch Klicken auf die Schaltfläche **Löschen** entfernen. Bei Klicken auf die Schaltfläche **Alle löschen** werden alle

Breakpoints bzw. Tracepoints gelöscht. Bei Auswahl der Schaltfläche **Code bearbeiten** wird der entsprechende Breakpoint/Tracepoint in der Datei direkt angezeigt.

7.3.3 XSLT/XQuery Einstellungen

Im Dialogfeld XSLT und 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 |**

XSLT/XQuery-Einstellungen oder klicken Sie auf das Symbol  in der Symbolleiste. Die verschiedenen Einstellungen werden weiter unten beschrieben.



Ausgabefenster

Konfiguriert die Ansicht des Ausgabefensters (Standardansicht, Textansicht, Grid-Ansicht oder Browser-Ansicht). Die Standardansicht ist die Ansicht, die im Dialogfeld "Optionen" ([Extras | Optionen](#)) auf dem Register "Datei" für den Ausgabedateityp 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 auf dem Register "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.

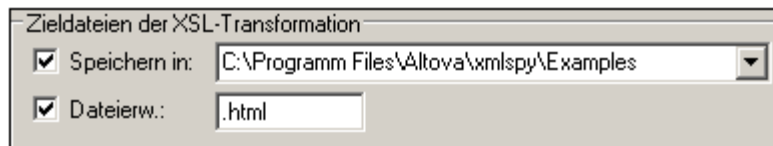
XQuery

Die Serialisierungsmethode bestimmt zwei Dinge: (i) das Dateiformat der generierten Ausgabedatei und (ii) die Regeln, die für das Schreiben der Ausgabe in die Ausgabedatei gelten. Zur Verfügung stehen die Optionen: HTML, Text, XHTML und XML. Je nach gewählter Option wird eine leere Datei des gewählten Dateityps erstellt, wenn Sie den Debugger innerhalb einer Debugger-Sitzung starten. Der Dateityp ist deshalb von Bedeutung, weil dadurch die aktuell aktiven XMLSpy-Funktionen für diesen Dateityp verfügbar werden (wie z.B. die Validierung für XML-Dateien).

Sie haben die Möglichkeit, die XML-Deklaration auszulassen und die Ausgabe einzurücken. Bei Aktivieren der Option "XML-Quelle immer überspringen" können Sie den optionalen Schritt zur Verknüpfung der XML-Datei immer überspringen, wenn Sie eine Debugger-Sitzung von einer XQuery-Datei aus starten. Bei DB2-Datenbanken können Sie die Maximalanzahl der abzurufenden Zeilen definieren.

Einstellung der Auswahl eines anderen Ausgabeformats

Im Dialogfeld "Projekteigenschaften" können Sie für XSLT-Transformationen Verknüpfungen für einen Ordner erstellen. Eine dieser Optionen ist die Auswahl des Zielordners für die XSLT-Transformation, für den Sie eine Dateierweiterung wählen können.



Der hier ausgewählte Dateityp bestimmt den Dateityp des Ausgabeformats für XML- oder XSLT-Dateien, die zu einem Projekt gehören, für das XSLT-Transformationseigenschaften definiert wurden.

7.3.4 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](#)) 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:

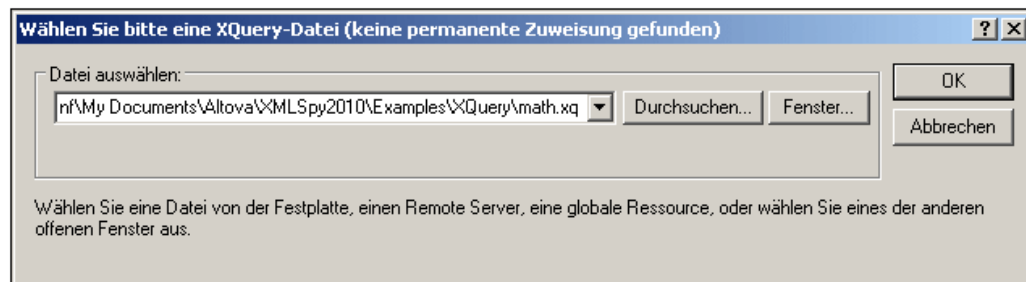
- In XML-Dateien: Öffnen Sie die Datei, wählen Sie **XSL/XQuery | XSL zuweisen** und wählen Sie die XSLT-Datei aus.
- In XSLT-Dateien: Öffnen Sie die Datei, wählen Sie den Befehl **XSL/XQuery | XML-Beispieldatei zuweisen...** und wählen Sie die XML-Datei aus.

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 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.



(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.

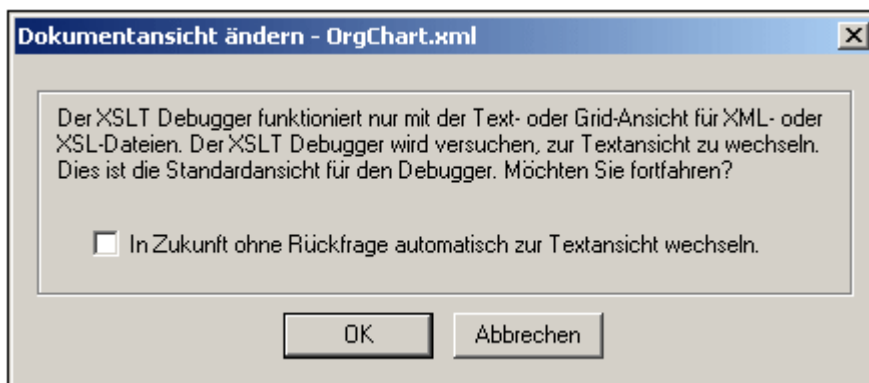
Alternative Methode zur Dateizuordnung

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 und 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.

7.3.5 Informationsfenster

Informationsfenster, die während der Debugger-Sitzung im XSLT und 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 und XQuery Debugger-Oberfläche (*siehe Abbildung unten*) angeordnet. Die Fenster und die darin angezeigten Informationen werden in diesem Abschnitt näher beschrieben.

XMLSpy Menu Bar		
Toolbar icons, including Debugger icons		
XML Document	XSLT/XQuery Document	Output File
Context (XSLT only) Variables XPath Watch	Call Stack Messages Templates (XSLT only) Info	

Standard-Layout der XSLT und XQuery Debugger-Oberfläche.

In der ersten Informationsfenstergruppe werden die folgenden Fenster in Form von Registerkarten in einem einzigen Fenster angezeigt:

- [Kontext](#) (nur bei XSLT Debugger-Sitzungen)
- [Variablen](#)
- [XPath-Watch](#)

In der zweiten Informationsfenstergruppe werden die folgenden Fenster in Form von Registerkarten in einem einzigen Fenster angezeigt:

- [Call Stack](#)
- [Messages](#)
- [Templates](#) (nur bei XSLT Debugger-Sitzungen)
- [Info](#)
- [Trace](#)

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-Watch" 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](#), [Call Stack](#), [Templates](#)).

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.

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.

Kontext		
Name	Typ	Wert / Attribute
OrgChart	Element	xmlns="http://www.xmlspy.com/schemas/orgchart" xmlns:ipo="http://www.altova.com/schemas/ipo" xsi:schemaLocation="http://www.xmlspy.com/schemas/orgchart OrgChart.xsd"
CompanyLogo	Element	
href	Attribute	nanonull.gif
Name	Element	
abc	Text	Organization Chart
Office	Element	
Name	Element	
Desc	Element	
Location	Element	
Address	Element	

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.

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.

Name	Type	Value
P personNeeded	xs:string	
	xs:string	Smith

Kontext Variablen XPath-Watch

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.

XPath-Watch

Das Fenster "XPath-Watch" 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.

Name	Typ	Wert
Sequence //*	Sequence	
OrgChart	Element	
ns xmlns	Namespace	http://www.xmlspy.com/schemas/orgchart
ns xmlns:ipo	Namespace	http://www.altova.com/IPO
ns xmlns:xsi	Namespace	http://www.w3.org/2001/XMLSchema-instance
xsi:schemaLocation	Attribute	http://www.xmlspy.com/schemas/orgchart OrgChart.xsd
CompanyLogo	Element	
href	Attribute	nanonull.gif

Kontext Variablen XPath-Watch

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-Watch". Verwenden Sie Ausdrücke, die gemäß der XPath-Version, die der XSLT-Version des XSLT Stylesheet entspricht (XPath 1.0 für XSLT 1.0

und XPath 2.0 für XSLT 2.0), korrekt sind.

Bitte beachten Sie: Wenn in der XML- bzw. XSLT-Datei Namespaces verwendet wurden, müssen Sie in Ihren XPath-Ausdrücken die richtigen Namespace-Präfixe verwenden.

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.

Call Stack		
Name	Pfad	
xsl:attribute	OrgChart\Variables.;	Temporary Result
img	OrgChart\Variables.;	Temporary Result
a	OrgChart\Variables.;	Temporary Result
xsl:for-each	OrgChart\Variables.;	Temporary Result
div	OrgChart\Variables.;	Temporary Result
Call Stack Info Meldungen Ablaufverfolgung		

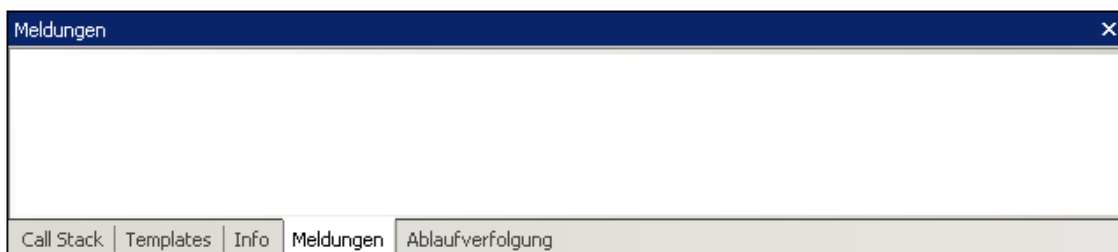
Wenn Sie in diesem Fenster auf einen Eintrag klicken, wird die jeweilige XSLT-Template/Anweisung im XSLT-Dokumentenfenster angezeigt. Wenn Sie auf eine Template/Anweisung klicken, die eine Built-in Template referenziert, wird die jeweilige built-in Template in einem separaten Fenster angezeigt, in dem alle Built-in Templates zu sehen sind.

Meldungen

Das Fenster "Meldungen" wird in XSLT- und XQuery-Debugger-Sitzungen angezeigt.

XSLT 1.0 und XSLT 2.0

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.

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.

Templates				
Match	Mode	Name	Priority	Pfad
n1:italic			0	OrgChart.xml
n1:bold			0	OrgChart.xml
/			-0.5	OrgChart.xml
processing-instruction()	#all		0	Built In Templates
processing-instruction()			0	Built In Templates

Call Stack | **Templates** | Info | Meldungen | Ablaufverfolgung

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 keinen 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.

Info

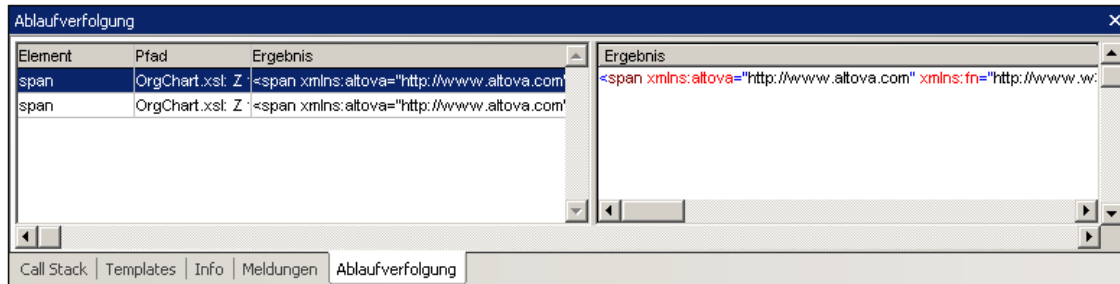
Das Fenster "Info" (siehe Abbildung unten) ist in XSLT- und XQuery Debugger-Sitzungen verfügbar und enthält Informationen über die aktuelle Debugger-Sitzung wie u.a. Informationen zur Version des verwendeten Debuggers, die Namen des Quell- und Zieldokuments und den Status des Debuggers.

Info	
Debugger-Typ	XSLT 2.0 Debugger
XML-Dokument	OrgChart.xml
XSL-Dokument	OrgChart.xsl
Ausgabedokument	OrgChart.html
Debugger-Status	Die Sitzung ist aktiv, der Debugger hat angehalten

Call Stack | Templates | **Info** | Meldungen | Ablaufverfolgung

Trace

Das Fenster "Ablaufverfolgung" wird in XSLT- und XQuery-Debugger-Sitzungen angezeigt.



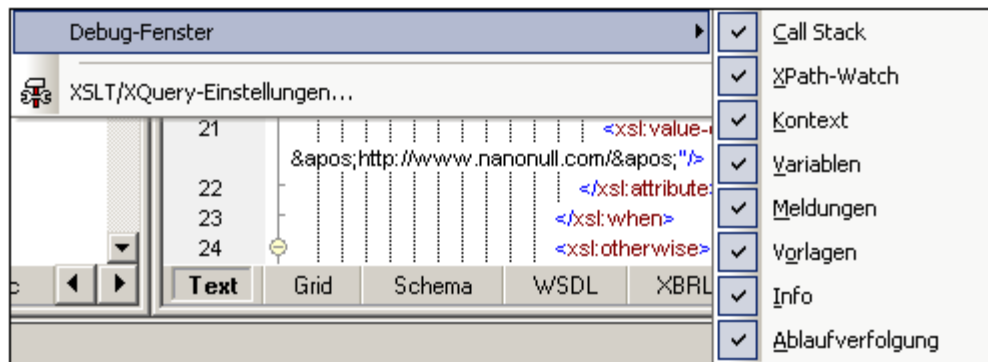
Das Fenster "Ablaufverfolgung" enthält das Element, für das der Tracepoint gesetzt wurde, seine Position im XSLT Stylesheet und das Ergebnis, das bei Ausführung des Elements generiert wird. Klicken Sie auf eine Zeile auf der linken Seite des Fensters um das vollständige Ergebnis auf der rechten Seite anzuzeigen.

Anordnen der Informationsfenster

Die Informationsfenster können an verschiedenen Stellen des XSLT und 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** andockt. Mit Hilfe der vier Pfeilsektoren wird das Fenster an der entsprechenden Seite des Zielfensters andockt. Mit Hilfe der Schaltfläche in der Mitte wird das Fenster als Register des Zielfensters andockt. Sie können ein Fenster auch als Register in einem anderen Fenster andocken, indem Sie es in die Registerleiste ziehen.



- 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 andockt.



Außerdem können Sie zwischen der an- und der abgedockten Position eines Fensters wechseln, indem Sie auf die Titelleiste des Fensters doppelklicken.

7.3.6 Breakpoints

Der XSLT und XQuery Debugger gestattet Ihnen, Breakpoints in XSLT-, XQuery- und XML-Dokumenten zu definieren. Breakpoints werden als strichlierte rote Linie angezeigt (*siehe Abb. unten*).

Bitte beachten Sie: 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 variable 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.

Bitte beachten Sie: 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:

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:

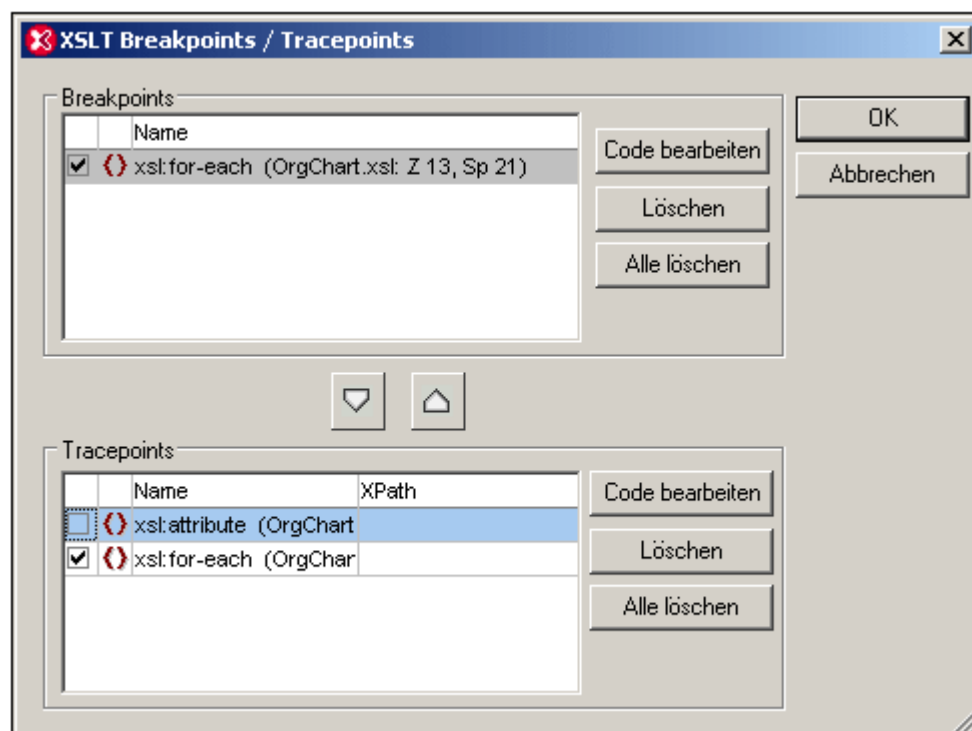
Sie können Breakpoints, die übersprungen werden sollen, auch deaktivieren ohne den Breakpoint zu löschen. Sie können diese Breakpoints später auch wieder aktivieren.

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**.

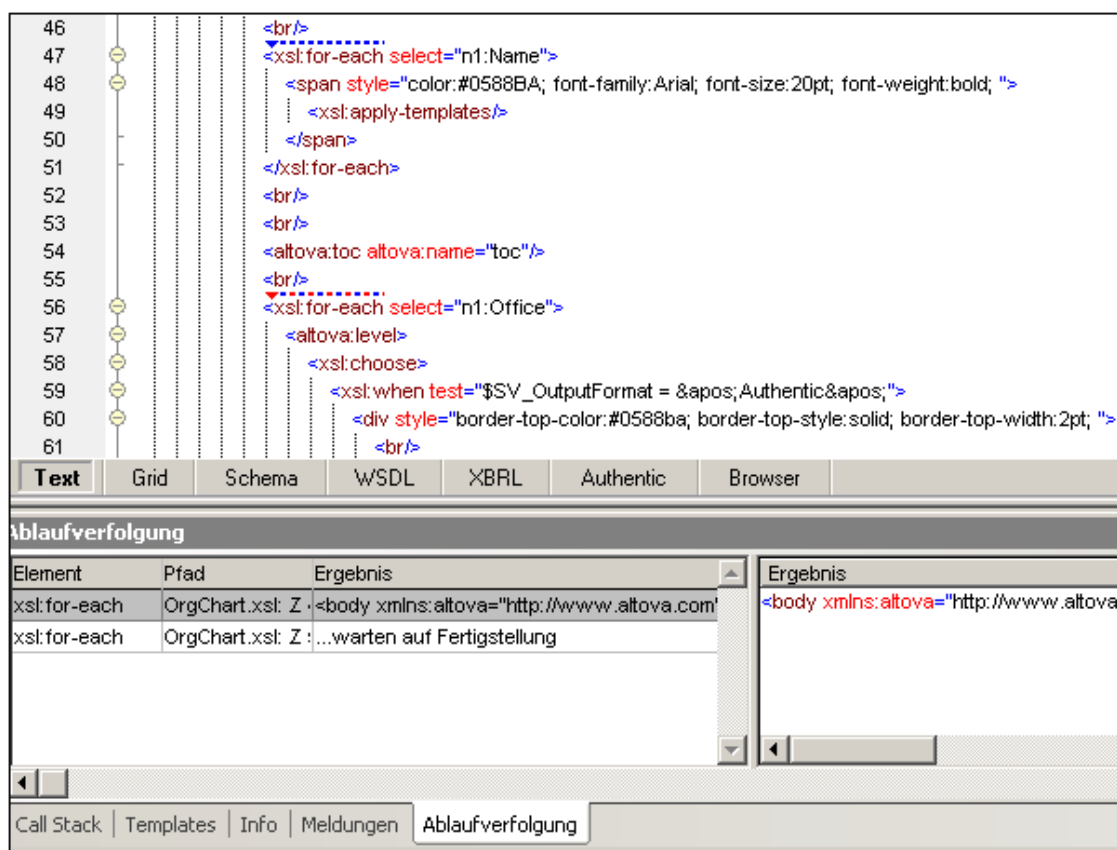
7.3.7 Tracepoints

Mit Hilfe des XSLT und 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 `xml: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*).

Bitte beachten Sie: Sie können innerhalb derselben Anweisung einen Tracepoint und einen Breakpoint setzen. Dies wird in Form einer blaurot gestrichelten Linie dargestellt (*siehe Screenshot*).



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.

Bitte beachten Sie: Die Tracepoints bleiben in einem Dokument erhalten, bis es geschlossen wird.

Tracepoints in XSLT-Dokumenten

Sie können in einem XSLT Stylesheet in XSLT-Anweisungen und literalen Ergebnissen Tracepoints setzen.

Tracepoints in XML- und XQuery-Dokumenten

Sie können in XML- und XQuery-Dokumenten Tracepoints setzen, jedoch haben diese hier keine Auswirkung.

Einfügen/Löschen von Tracepoints

So fügen Sie einen Tracepoint ein:

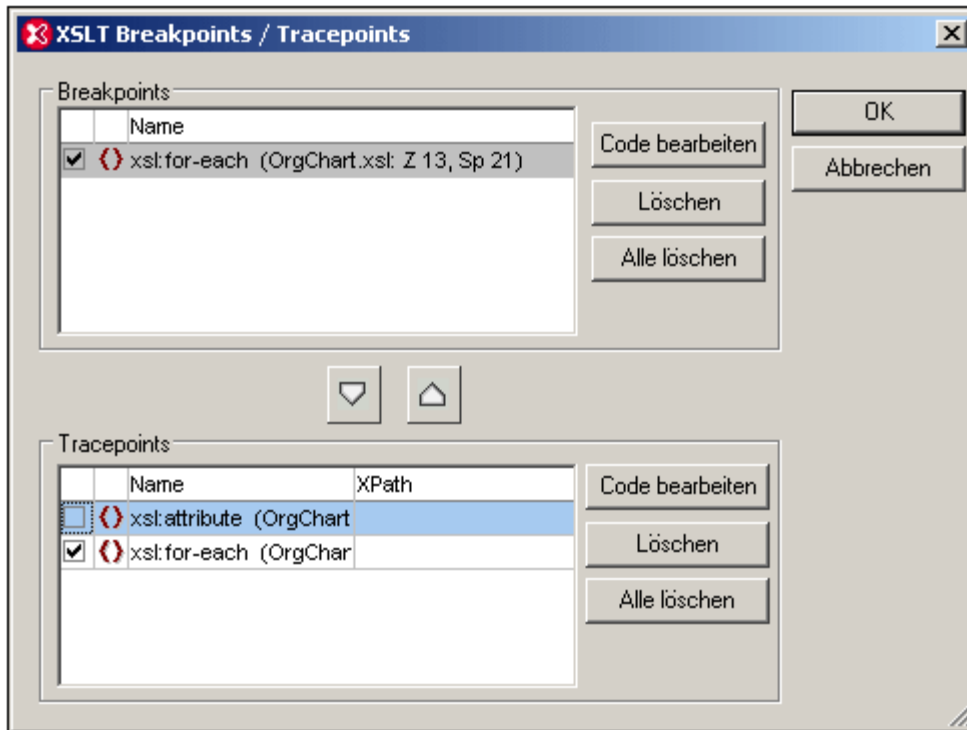
1. Platzieren Sie den Cursor an dem Punkt im XSLT-Dokument, an dem Sie den Tracepoint einfügen möchten. Sie können Tracepoints sowohl in der Text- als auch in der Grid-Ansicht setzen.
2. Wählen Sie eine der folgenden Methoden:
 - Wählen Sie **XSL/XQuery | Tracepoint einfügen/löschen**.
 - Drücken Sie **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.

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.

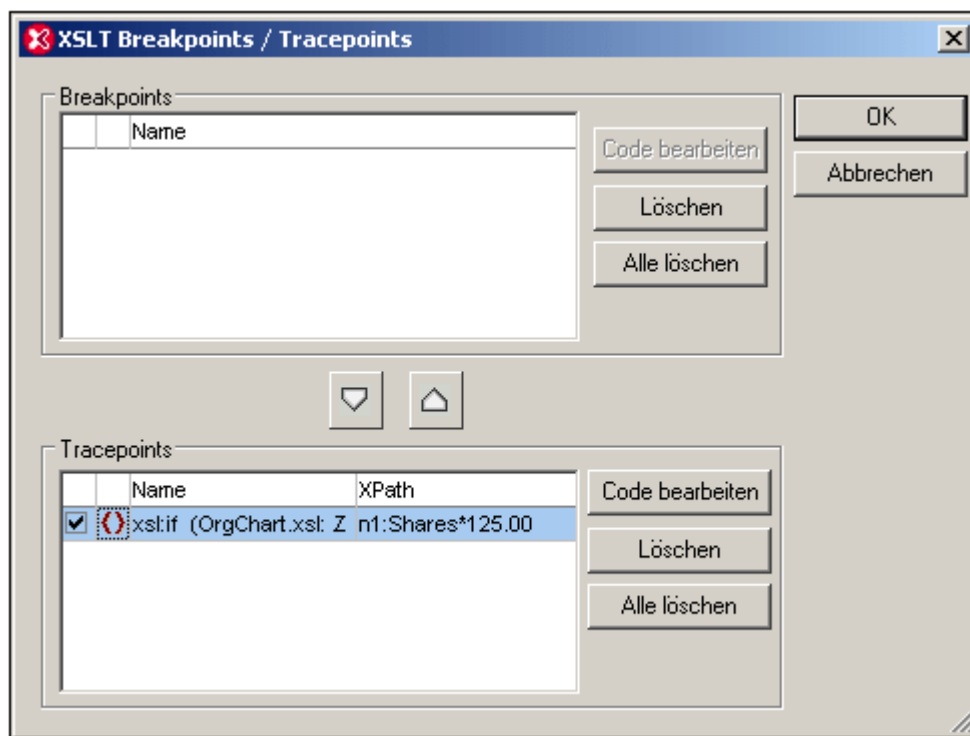
Beispiel:

Im folgenden Beispiel wird die Datei `OrgChart.xsl`, verwendet, die Sie im Ordner XMLSpy Examples finden.

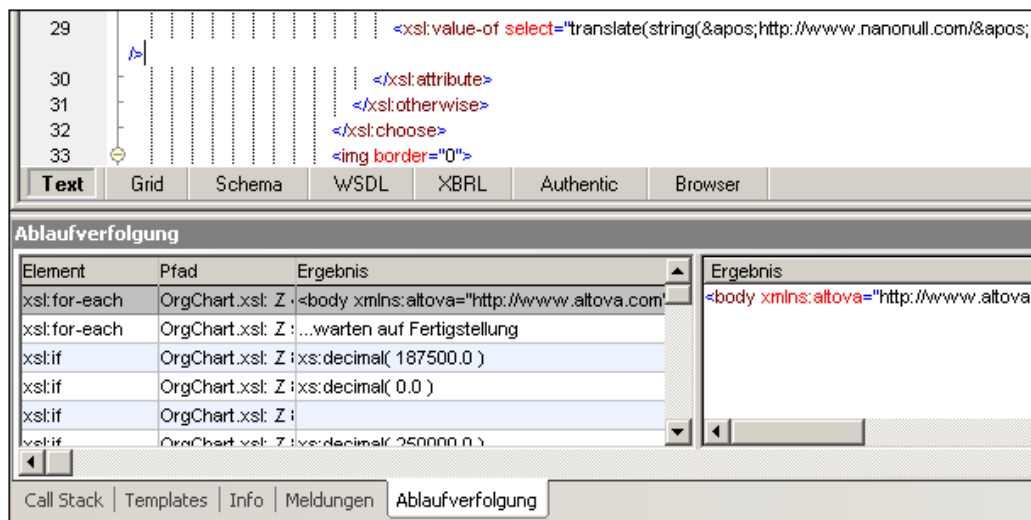
Der Tracepoint wird so gesetzt, dass der Kontext Node "Person" ist. Das Element "Person" enthält ein Element "Shares". Wir möchten die Anzahl der Aktien (Shares), die jede Person hat, multipliziert mit 125 (dem Wert einer einzelnen Aktie) anzeigen.

Gehen Sie folgendermaßen vor:

1. Öffnen Sie die Datei `OrgChart.xml`.
2. Setzen Sie in Zeile 848 einen Tracepoint.
3. Öffnen Sie das Dialogfeld "XSLT Breakpoints / Tracepoints" und geben Sie den XPath `n1:Shares*125.00` für diesen Tracepoint ein.



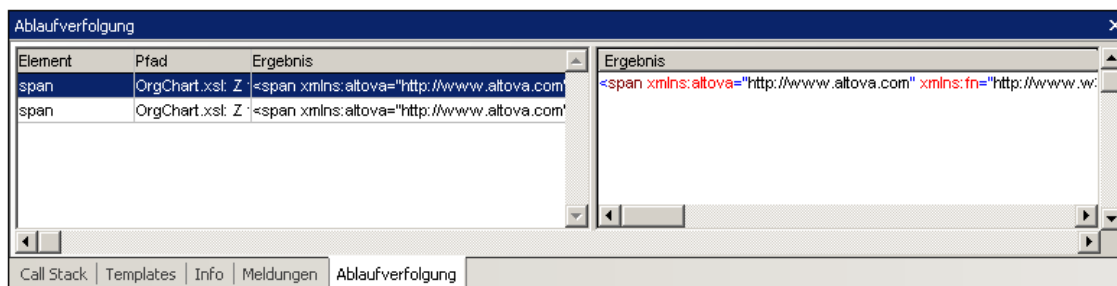
4. Starten Sie den Debugger. Das Ergebnis des XPath, den Sie als Tracepoint eingegeben haben, wird im Fenster "Ablaufverfolgung" angezeigt.



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](#) (siehe *Screenshot*) angezeigt. Klicken Sie auf eine Zeile im Fenster "Ablaufverfolgung", um das vollständige Ergebnis der Anweisung auf der rechten Seite des Dialogfelds (siehe *Screenshot*) anzuzeigen.

Bitte beachten Sie: Die Ergebnisse werden erst dann im Fenster "Ablaufverfolgung" angezeigt, wenn diese Ablaufverfolgung-Anweisung fertig ausgeführt wurde.



Aktivieren/Deaktivieren von Tracepoints:

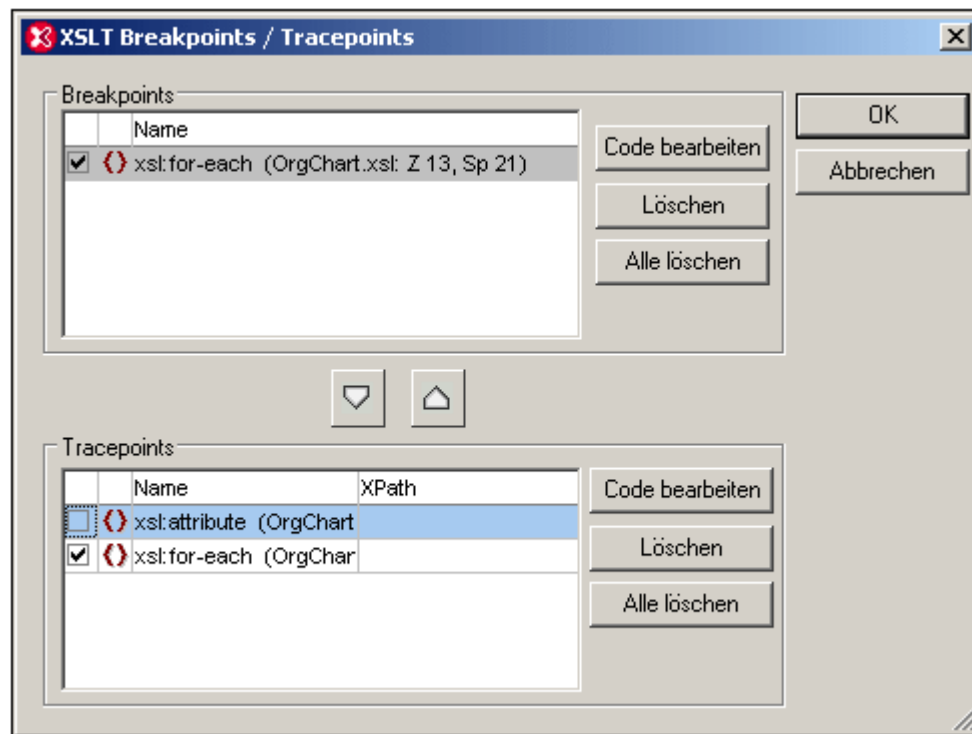
Sie können Tracepoints, nachdem Sie sie eingefügt haben, deaktivieren, wenn diese übersprungen aber nicht gelöscht werden sollen. Bei Bedarf können Sie diese jederzeit wieder aktivieren.

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.

8 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, Listefeldern und Optionsfeldern darstellen. Die Daten, die der Benutzer in der Authentic-Ansicht eingibt, werden in die XML-Datei eingegeben.

Nanonull, Inc.

Location:

Street: 119 Oakstreet, Suite 4876 City: Vereno State & Zip: DC 29213	Phone: +1 (321) 555 5155 0 Fax: +1 (321) 555 5155 4 E-mail: office@nanonull.com

Vereno Office Summary: 4 departments, 15 employees.

The company was established **in Vereno in 1995** as a privately held software company. Since 1996, Nanonull has been actively involved in developing nanoelectronic software technologies. It released the first version of its acclaimed *NanoSoft Development Suite* in February 1999. Also in 1999, Nanonull increased its capital base with invesment from a consortium of private investment firms. The company has been expanding rapidly ever since.

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 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](#), in dem das Bearbeitungsfenster der Authentic-Ansicht beschrieben wird, und (ii) im Abschnitt [Menü "Authentic"](#) der Benutzerreferenz.

8.1 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:

Nanonull, Inc.

Location:

Street: 119 Oakstreet, Suite 4876 City: Vereno State & Zip: DC 29213	Phone: +1 (321) 555 5155 0 Fax: +1 (321) 555 5155 4 E-mail: office@nanonull.com

Vereno Office Summary: 4 departments, 15 employees.

The company was established **in Vereno in 1995** as a privately held software company. Since 1996, Nanonull has been actively involved in developing nanoelectronic software technologies. It released the first version of its acclaimed *NanoSoft Development Suite* in February 1999. Also in 1999, Nanonull increased its capital base with invesment from a consortium of private investment firms. The company has been expanding rapidly ever since.

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

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](#).

Voraussetzungen für das Tutorial

Sie finden alle für das Tutorial benötigten **Dateien** im Ordner `C:\Documents and Settings\<username>\My Documents\Altova\<ProductName>\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)

Bitte beachten Sie: 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.

8.1.1 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.

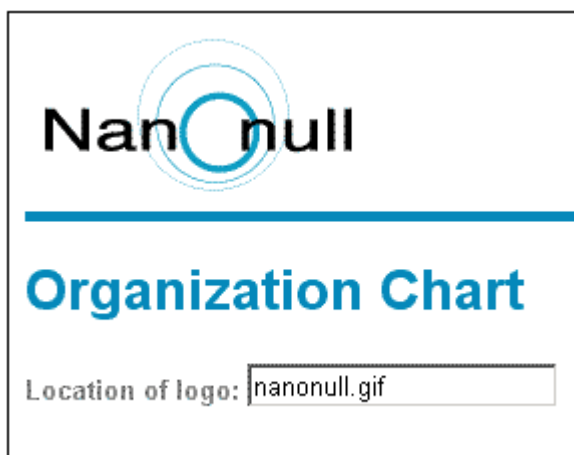
Öffnen eines vorhandenen XML-Dokuments

Die Datei, die Sie öffnen werden, heißt `NanonullOrg.xml`. Sie befindet sich im Ordner `Examples` der 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.

- Denn:
 1. der Datei wurde bereits ein StyleVision Power Stylesheet (SPS) zugewiesen und.
 2. 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. Wählen Sie **Datei | Neu** und wählen Sie im Dialogfeld "Neues Dokument anlegen" XML als den Dateityp der neu zu erstellenden Datei aus.
2. Klicken Sie auf **STYLEVISION Stylesheet auswählen...** und suchen Sie das gewünschte StyleVision Power Stylesheet.

Über Authentic | Neues Dokument

1. Wählen Sie **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.

8.1.2 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".

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>
```

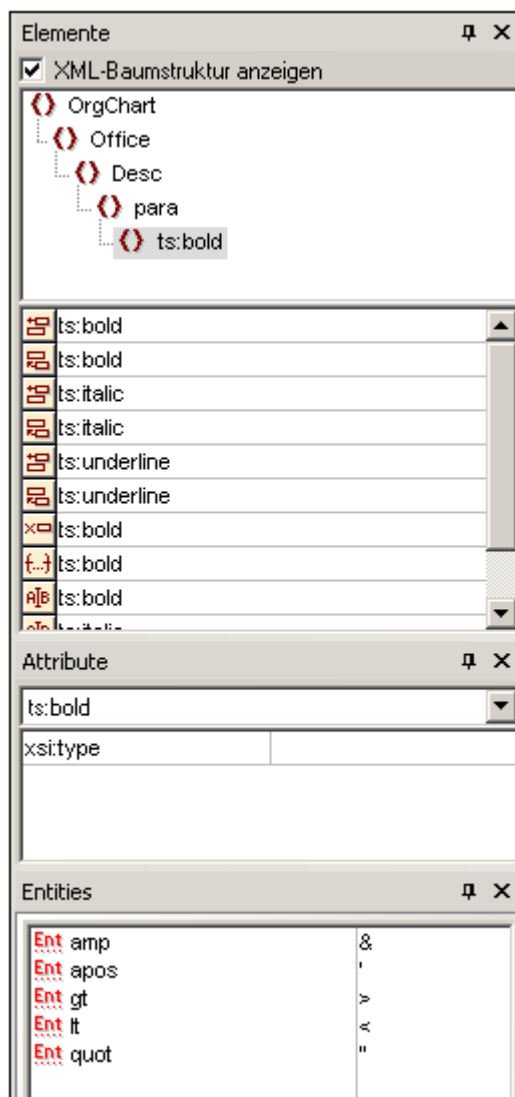
```
<Tel>1234567</Tel>
</Person>
</DocumentRoot>
```

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 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](#).

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 entfernbaren 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](#).

8.1.3 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](#). 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](#)

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](#) oder durch Klicken auf den Eintrag für die Node-Operation in der [Element-Eingabehilfe](#) 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](#) klicken. In speziellen Fällen, in denen Elemente als Absätze oder Listeneinträge definiert wurden, können Sie durch Drücken der [Eingabetaste](#) 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](#), [Node entfernen](#) und [Element löschen](#) 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.



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 Symbolleiste-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  (Einfügapunktauswahl) 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](#) näher beschrieben.

8.1.4 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](#). 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.



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](#) 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 Programmcode Listen), 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](#), die in der Entities-Eingabehilfe angezeigt wird. Die [Entity wird an der gewünschten Stelle eingefügt](#). 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 in der Optionen große Markup-Symbole \(oder kleine Markup-Symbole\) anzeigen](#).

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.

8.1.5 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:

- 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

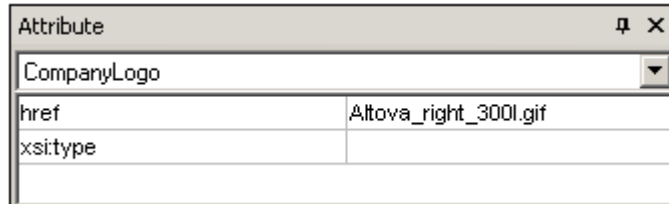
Attributwerte können als normaler Text oder als Text in einem Eingabefeld eingegeben werden oder als vom Benutzer aus einer Auswahl gewählter Eintrag, der auf einen XML-Wert gemappt wird. Attributwerte werden auf dieselbe Art eingegeben, wie Element-Inhalt: siehe [Eingabe von Daten in der Authentic-Ansicht](#). 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 Attribut-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_300l.gif` (ein Bild im Ordner `Examples`).

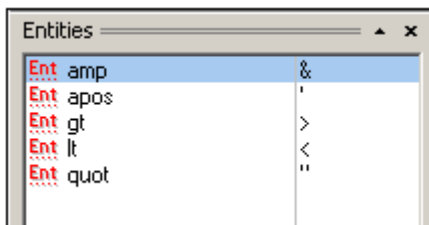


Daraufhin wird das Nanonull-Logo durch das Altova-Logo ersetzt.

Anmerkung: Entities können nicht in der Attribut-Eingabehilfe eingegeben werden.

8.1.6 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*)

Marketing (2)		
First	Last	Title
Joe	Martin	Marketing Manager Europe &

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](#).

8.1.7 Drucken des Dokuments

Bei Ausdruck eines XML-Dokuments von der Authentic-Ansicht aus bleibt die in der Authentic-Ansicht sichtbare Formatierung erhalten.

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 Teil einer um 50 % verkleinerten Seite in der Druckansicht.

Page 1 of 6

ALTOVA

www.altova.com

Organization Chart

Location of logo:

Nanonull, Inc.

Location:

Street: 119 Oakstreet, Suite 4876 City: Vereo State & Zip: DC 29213	Phone: +1 (321) 555 5155 0 Fax: +1 (321) 555 5155 4 E-mail: office@nanonull.com

Vereo Office Summary: 4 departments, 15 employees.

The company was established **in Vereo in 1995** as a privately held software company. Since 1996, Nanonull has been actively involved in developing nanoelectronic software technologies. It released the first version of its acclaimed *NanoSoft Development Suite* in February 1999. Also in 1999, Nanonull increased its capital base with investment from a consortium of private investment firms. The company has been expanding rapidly ever *my iText* since.

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".

8.2 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](#) 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](#).
- 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](#).
- In der Enterprise und der Professional Edition von Altova-Produkten kann der Benutzer der Authentic-Ansicht XML-Dokumente mit [digitalen XML-Signaturen](#) signieren und diese Signaturen überprüfen.
- Die [Bildformate](#), die in der Authentic-Ansicht angezeigt werden können.

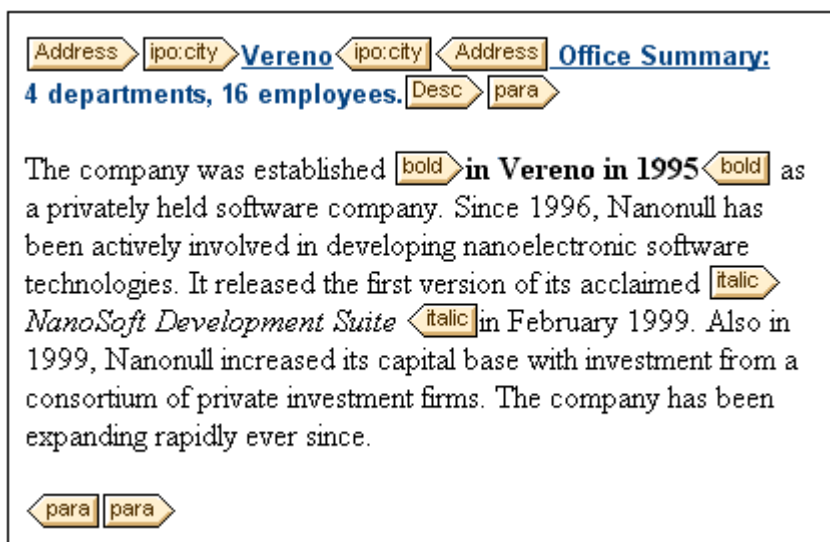
8.2.1 Grundlegendes zur Bearbeitung

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*).

Vereno Office Summary: 4 departments, 16 employees.

The company was established **in Vereno in 1995** as a privately held software company. Since 1996, Nanonull has been actively involved in developing nanoelectronic software technologies. It released the first version of its acclaimed *NanoSoft Development Suite* in February 1999. Also in 1999, Nanonull increased its capital base with investment from a consortium of private investment firms. The company has been expanding rapidly ever since.

Ein editierbares Dokument in der Authentic-Ansicht ohne XML-Markup-Tags

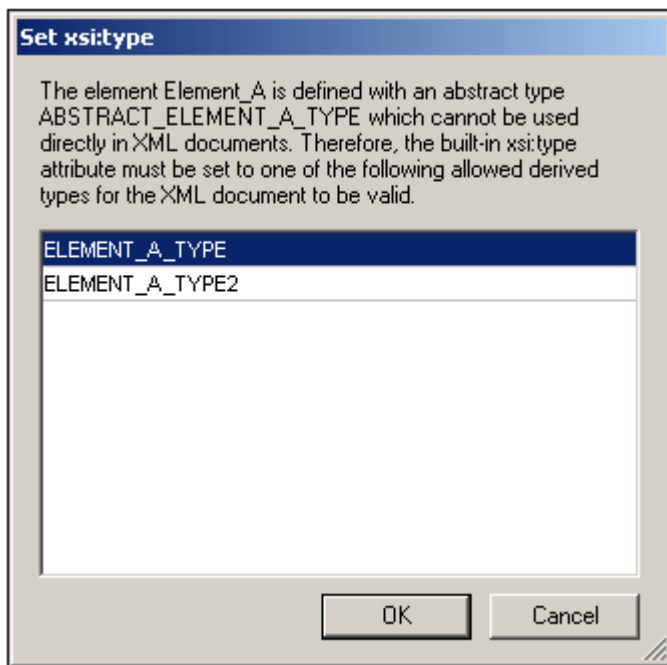


Ein editierbares Dokument in der Authentic-Ansicht mit XML-Markup-Tags

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.



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](#).

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](#) unter dem Befehl "Element einfügen".

Verwendung von RichEdit in der Authentic-Ansicht

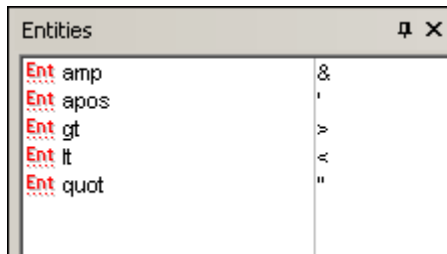
Wenn Sie den Cursor in der Authentic-Ansicht innerhalb von einem Element platzieren, das als RichEdit-Komponente erstellt wurde, werden die Schaltflächen und Steuerelemente in der RichEdit-Symbolleiste (*Abbildung unten*) aktiv. Anderfalls sind sie ausgegraut.



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"](#) (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](#). Nachdem Sie diese definiert haben, werden die Entities in der [Eingabehilfe "Entities"](#) 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](#). 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.)

8.2.2 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](#) 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](#) und die [Bearbeitungssymbole für CALS/HTML-Tabellen](#) werden im Folgenden beschrieben.

SPS-Tabellen

In der Authentic-Ansicht werden zwei Arten von SPS-Tabellen verwendet: statische Tabellen und dynamische 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.

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

Bitte beachten Sie: Die Symbole oder Befehle zum Bearbeiten dynamischer Tabellen **dürfen nicht** zum Bearbeiten statischer Tabellen verwendet werden.

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 einer Zeile die Tabulatortaste drücken, wird eine neue Zeile erstellt.

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](#)).

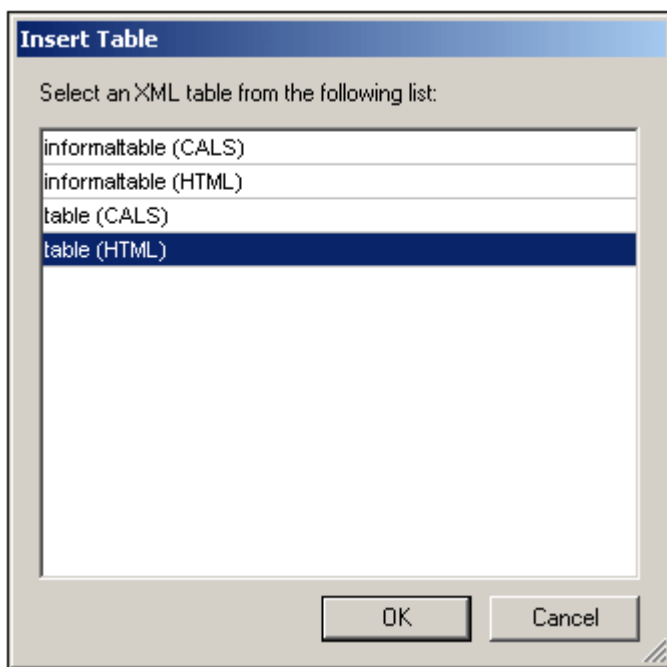
Einfügen von Tabellen

So fügen Sie eine CALS/HTML-Tabelle ein:

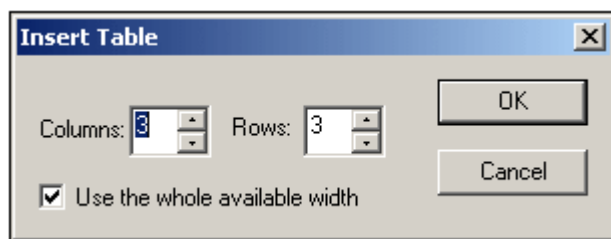
1. Setzen Sie Ihren Cursor an die Stelle, an der die Tabelle eingefügt werden soll und



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**.



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

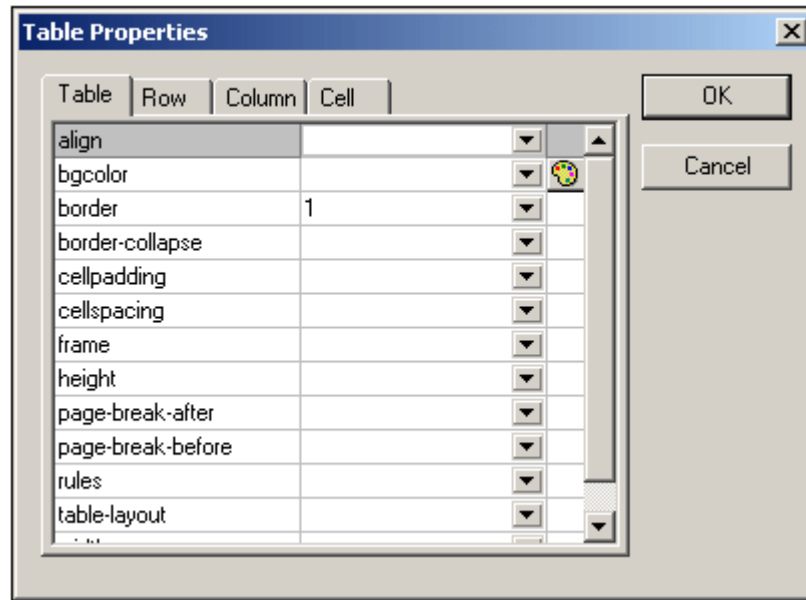
sind, stehen für die Formatierung zur Verfügung. Wenn im Design Werte für diese Attribute definiert wurden, können Sie diese in der Authentic-Ansicht außer Kraft setzen.

- 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:

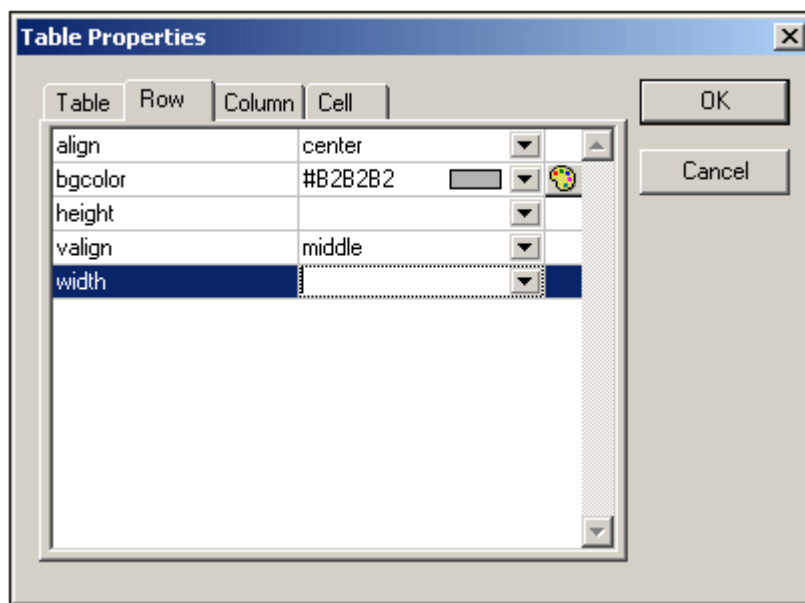
1. Platzieren Sie Ihren Cursor zum Formatieren der Tabelle an eine beliebige Stelle in der

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".



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](#)" 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.

Bitte beachten Sie: 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.

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](#).

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.

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

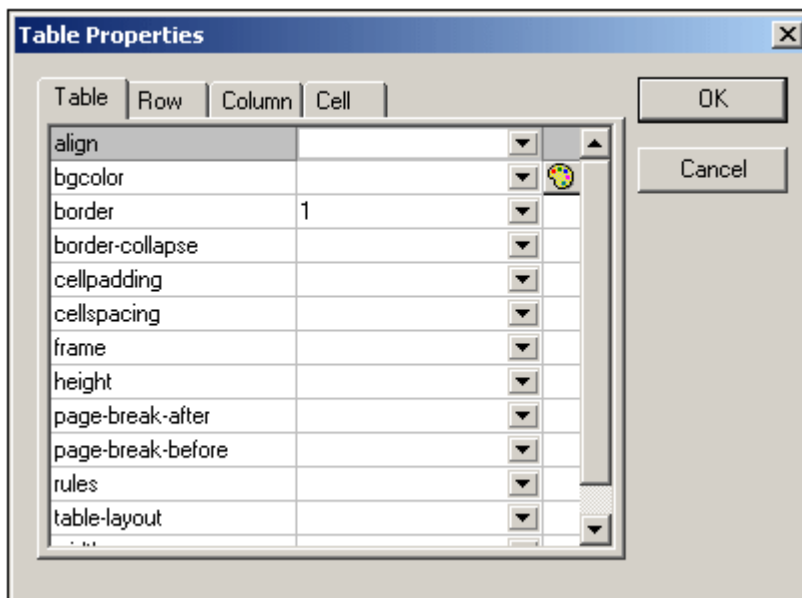


Dieser Befehl richtet den Inhalt der Zellen bündig am unteren Zellenrand aus.

Tabelleneigenschaften



Der Befehl "Tabelleneigenschaften" öffnet das Dialogfeld "Tabelleneigenschaften". Dieses Symbol wird nur bei HTML-Tabellen aktiv, bei CALS-Tabellen ist es deaktiviert.



8.2.3 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](#)) laden und anzeigen.
- Sie können eine [Datenbankabfrage](#) durchführen, um bestimmte Datensätze anzuzeigen.
- Sie können Datensätze hinzufügen, ändern und löschen und Ihre Änderungen in der DB speichern. Siehe [Ändern einer DB-Tabelle](#).

So öffnen Sie ein DB-basiertes StyleVision Power Stylesheet in der Authentic Ansicht:

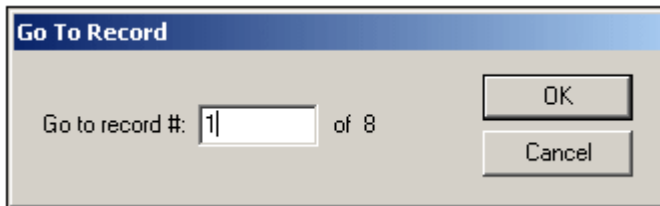
- Klicken Sie auf **Authentic | Datenbank bearbeiten**, und suchen Sie nach dem gewünschten StyleVision Power Stylesheet.

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.



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](#).

DB-Abfragen

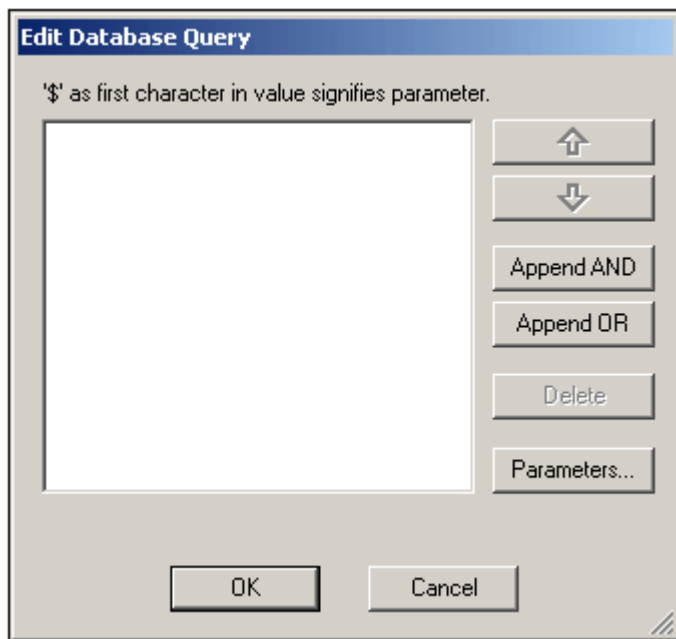
Mit Hilfe einer DB-Abfrage können Sie die Datensätze einer in der Authentic-Ansicht angezeigten Tabelle abfragen. Eine Abfrage wird für eine einzelne Tabelle erstellt. Es kann für jede Tabelle nur eine Abfrage erstellt werden. Sie können während der Bearbeitung eines Dokuments jederzeit eine Abfrage 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.

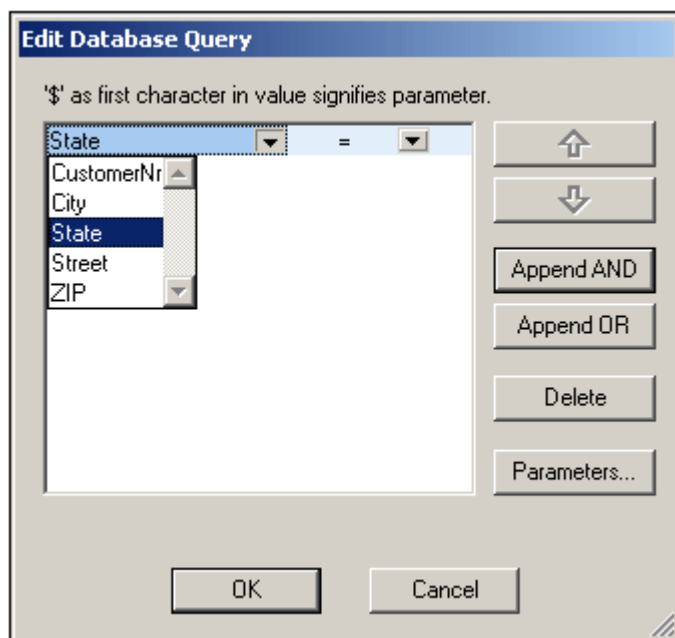
Bitte beachten Sie: Falls eine Meldung angezeigt wird, dass zu viele Tabellen geöffnet sind, können Sie die Anzahl der offenen Tabellen mit Hilfe einer Abfrage durch Filtern reduzieren.

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.



4. 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](#) weiter unten.
5. 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](#).

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**:

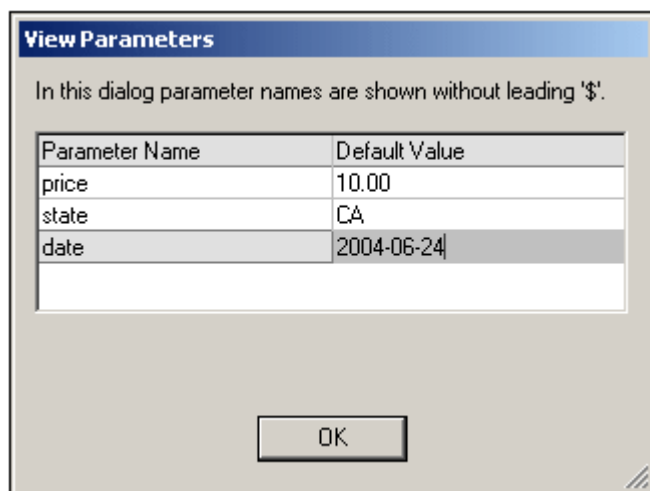
=	Ist gleich
<>	Ist nicht gleich
<	Kleiner als
<=	Kleiner oder gleich
>	Größer als
>=	Größer oder gleich
LIKE	Phonetisch ähnlich
NOT LIKE	Phonetisch nicht ähnlich
IS NULL	Ist leer
NOT NULL	Ist nicht leer

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).

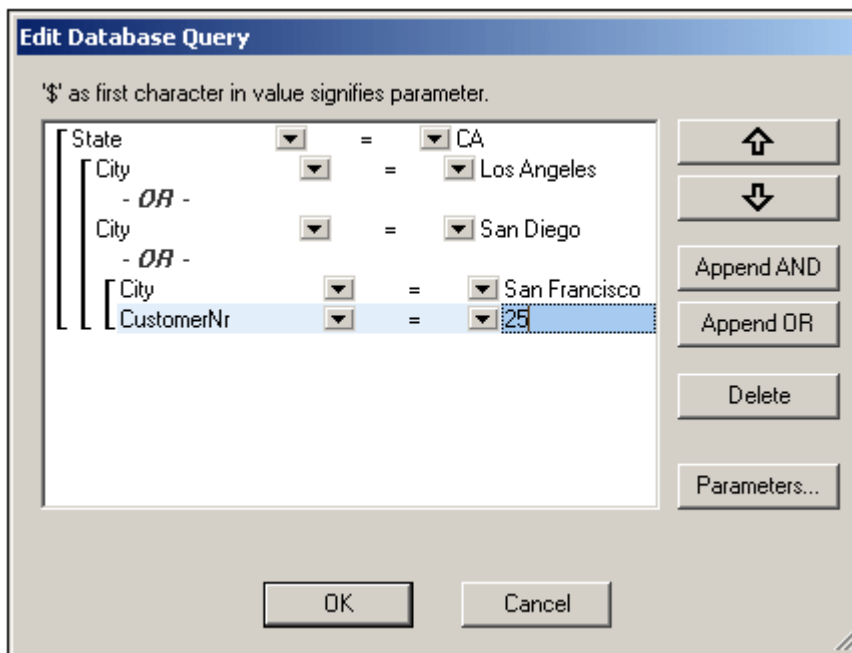
Um die Parameter, die für das SPS definiert wurden, anzuzeigen, klicken Sie im Dialogfeld **Datenbankabfrage bearbeiten** auf die Schaltfläche **Parameter**, um das unten gezeigte Dialogfeld **Parameter anzeigen** aufzurufen.



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) )
```

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.

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 Texteingenschaften 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](#)) 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:

- 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....**

Bitte beachten Sie: 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.

8.2.4 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](#).
- Das Datum wird durch [Eingabe des Werts](#) 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.

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).

Organization Chart

Location of logo:

Last Updated: 2003-09-01 

Um die Datumswahl (siehe unten) anzuzeigen, klicken Sie auf das Datumswahl-Symbol.

Location of logo:

Last Updated: 2003-09-01

Nanonull, Inc.

Location:

September 2003

M	T	W	T	F	S	S
25	26	27	28	29	30	31
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	1	2	3	4	5

Today No Timezone

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.

Texteintrag

In Datumsfeldern, die keine Datumswahl haben (*siehe Screenshot*) können Sie das Datum direkt durch Eingabe des neuen Werts bearbeiten.

Bitte beachten Sie: Beim Bearbeiten des Datums darf das Format nicht geändert werden.

Invoice Number: 001
2006-03-10
Customer: The ABC Company
Invoice Amount: 40.00

Wenn Sie ein Datum ändern und es sich daraufhin außerhalb des gültigen Bereichs für Datumsangaben befindet, wird das Datum rot angezeigt, um den Fehler anzuzeigen. Wenn Sie den Mauszeiger über das ungültige Datum platzieren, erscheint eine Fehlermeldung (*siehe Screenshot*).

Invoice Number: 001
2006-03-32
Customer: ERROR: Invalid value for datatype date in element
Invoice Amount: 40.00

Wenn Sie versuchen, das Datumsformat zu ändern, wird das Datum rot angezeigt (siehe *Screenshot*).

Invoice Number: 001
2006/03/10
Customer: The ABC Company
Invoice Amount: 40.00

8.2.5 Definieren von 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.

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. Klicken Sie auf **Authentic | XML-Entities definieren....** Daraufhin wird das Dialogfeld "Entities definieren" geöffnet. (*Abbildung unten*)

Name	Type	PUBLIC	Value/Path	NDATA
nano_dc	Internal		Nanonull, Inc	
nano_eu	Internal		Nanonull Europe, AG	
nano_ma	Internal		Nanonull Partners, Inc	
website	Internal		http://www.nanonull.com/	
branches	SYSTEM		branches.xml	
logo	SYSTEM		nanonull.gif	GIF

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 gepackte 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 gepackt, sondern an den entsprechenden Prozessor gesendet werden soll. Das Feld "NDATA" muss daher einen Wert enthalten, der angibt, dass es sich um eine ungepackte Entity handelt.

Dialogfeldfunktionen

Im Dialogfeld "Entities definieren" können Sie Folgendes tun:

- Entities anhängen
- Entities einfügen
- Entities löschen
- 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.

8.2.6 XML-Signaturen

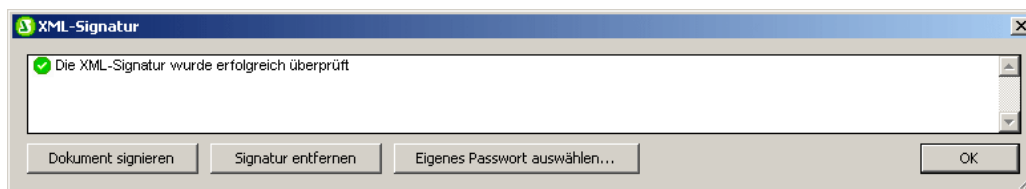
Bei der Erstellung eines SPS kann eine XML-Signatur für die Authentic-Ansicht konfiguriert werden. Wenn XML-Signaturen im SPS aktiviert sind, kann der Benutzer der Authentic-Ansicht die Authentic XML-Datei mit der aktivierten Signatur digital signieren. Wenn nach der Signierung des Dokuments Änderungen 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 Symbolleiste-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. Zertifikat-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 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.

8.2.7 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:

- 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.

8.2.8 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).

Bitte beachten Sie: 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.

8.3 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

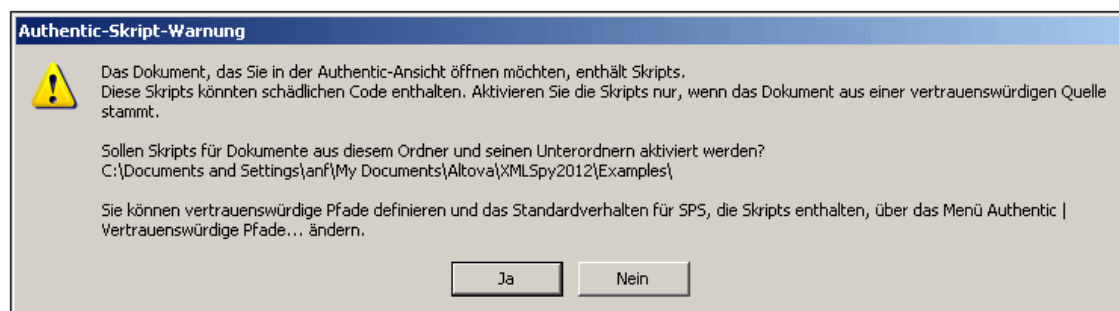
** Sowohl Trusted als auch Untrusted Version

*** 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.



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](#) 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](#) und fügen Sie den Ordner hinzu oder ändern Sie die Pfade nach Wunsch.

Eine Beschreibung des Dialogfelds "Vertrauenswürdige Pfade" finden Sie in der Benutzerreferenz unter der Beschreibung des Menübefehls [Authentic | Vertrauenswürdige Pfade](#).

Anmerkung: Bei Aufruf von XMLSpy über die COM-Schnittstelle (Beschreibung dazu siehe [Referenz für Programmierer](#)) **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"](#).

9 HTML, CSS, JSON

XMLSpy bietet intelligente Bearbeitungsfunktionen für [HTML](#)-, [CSS](#)- und [JSON](#)-Dokumente. Alle drei Dokumenttypen werden in der [Textansicht](#) bearbeitet. Außerdem können JSON-Dokumente in der Grid-Ansicht bearbeitet werden 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](#), [CSS](#) und [JSON](#) beschrieben.

9.1 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](#) genauer beschrieben, doch die wichtigsten Funktionen sowie die HTML-spezifischen Optionen werden unten aufgelistet.

Grad der Unterstützung

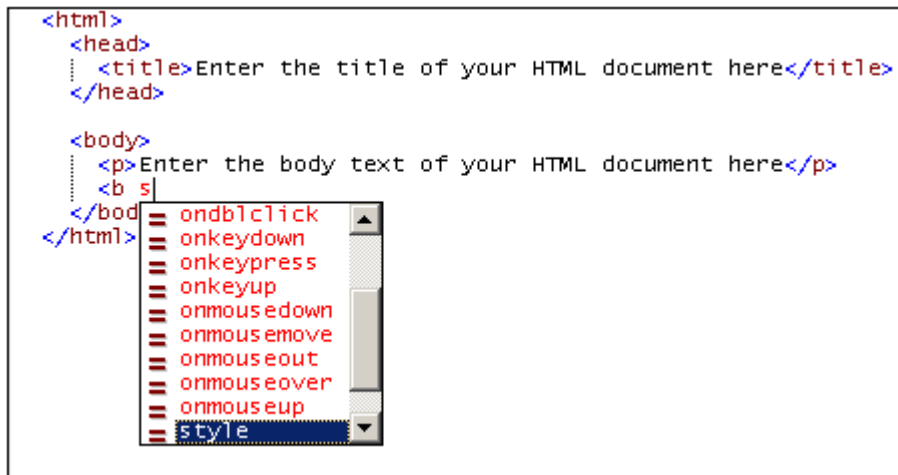
XMLSpy unterstützt HTML 4.0 und HTML 5.0. Es stehen Eingabehilfen und intelligente Bearbeitungsfunktionen für die entsprechende HTML-Version zur Verfügung. Eine Beschreibung zu diesen Funktionen finden Sie weiter unten.

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](#) 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.



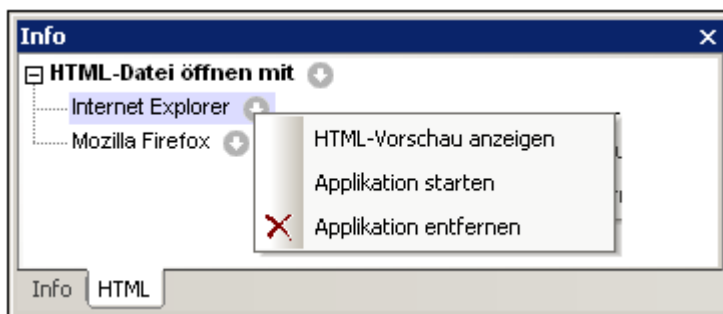
Die Autokomplettierung für Elemente wird angezeigt, wenn die linke Klammer des Node-Tag eingegeben wird. Bei Eingabe des Start-Tag eines Element-Node wird automatisch auch der End-Tag eingefügt. Dadurch wird die Wohlgeformtheit des Dokuments gewährleistet.

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.

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](#).

Befehle der Browser-Ansicht

Die Befehle der Browser-Ansicht finden Sie im Menü **Browser**.

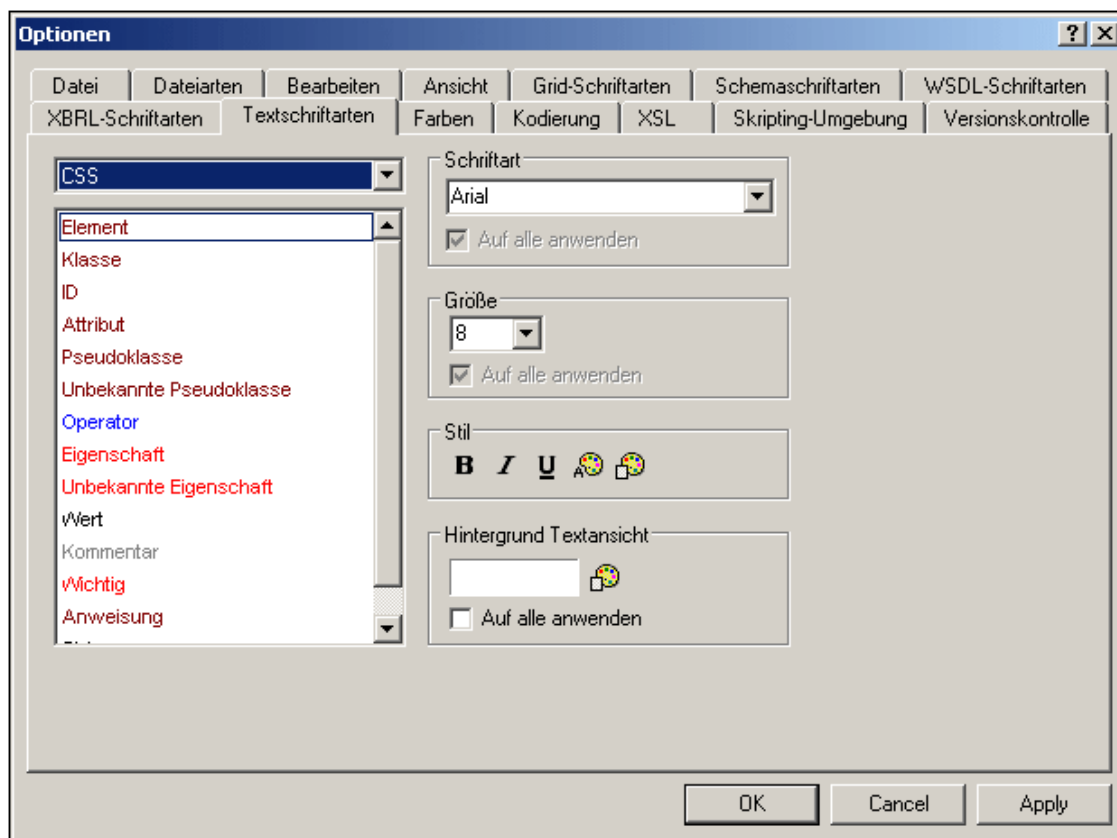
9.2 CSS

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;
}
```

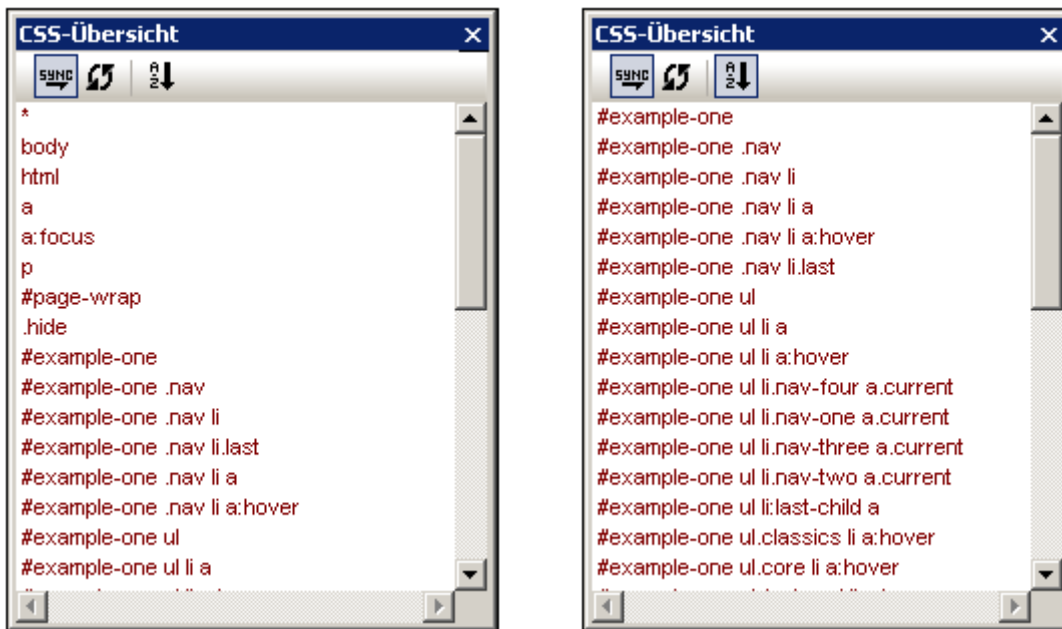
Sie können die Farben der verschiedenen CSS-Komponenten im Dialogfeld "Optionen" (*Abbildung unten*) auf dem Register "Textschriftarten" definieren. Wählen Sie in der Auswahlliste links oben die Option "CSS" aus und wählen Sie anschließend im Bereich "Stile" für die einzelnen CSS-Elemente die gewünschte Farbe aus.



Klappleisten: Jede Regel kann erweitert und reduziert angezeigt werden, indem Sie links von der Regel (siehe *Abbildung oben*) auf das Plus- bzw. Minus-Symbol klicken. Beachten Sie außerdem, dass die geschwungenen Klammern, innerhalb derer eine Regel angezeigt wird (*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.

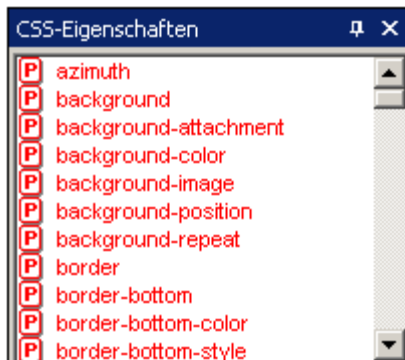


Synchronisiert die Eingabehilfe mit dem aktuellen Zustand des Dokuments.

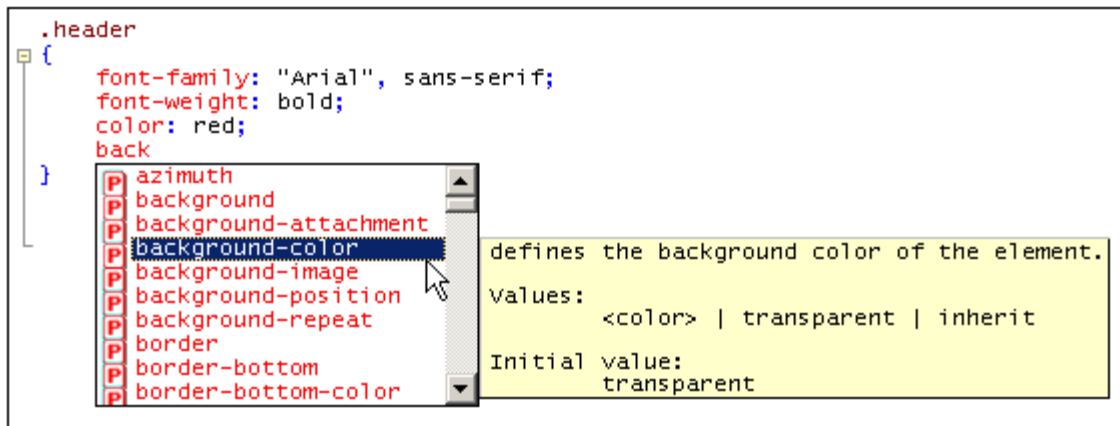


Schaltet die alphabetische Sortierung ein bzw. aus. Wenn diese Sortierung deaktiviert ist, werden die Selektoren in der Reihenfolge aufgelistet, in der sie im Dokument aufscheinen. Bei alphabetischer Sortierung werden ID-Selektoren als erste angezeigt, da ihnen ein Rautensymbol vorangestellt ist (z. 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.



Autokomplettierung von Eigenschaften und Tooltips 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.

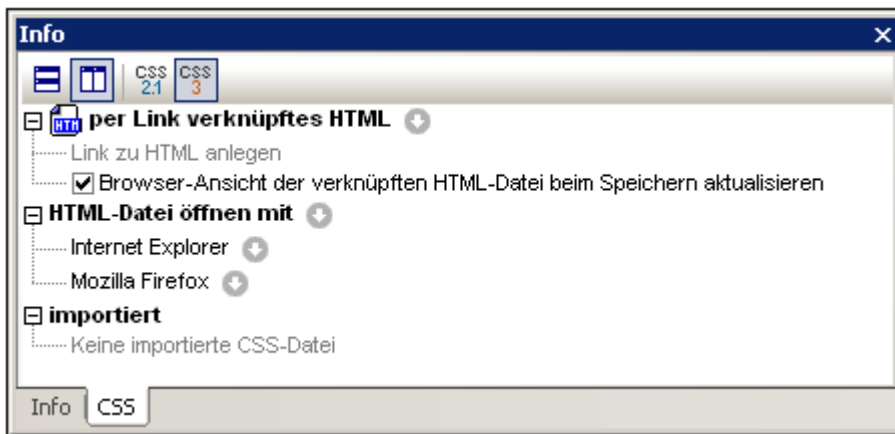


Sie können einen Tooltip mit der Definition der Eigenschaft und ihren möglichen Werten anzeigen, indem Sie die Liste hinunterscrollen oder mit der Nach oben- und Nach unten-Taste Ihrer Tastatur zu der Eigenschaft navigieren. Der Tooltip der markierten Eigenschaft wird angezeigt. Um eine Eigenschaft einzufügen, 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 unten*). 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 HTML-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.
- 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.

9.3 JSON

Altova Website:  [JSON-Editor](#)

JSON-Dokumente können über die Grid-Ansicht und mit Hilfe der intelligenten Bearbeitungsfunktionen der Textansicht bearbeitet werden.

- Die [Grid-Ansicht](#) bietet eine Reihe von Bearbeitungsfunktionen, die sich beim Bearbeiten großer Dateien als sehr nützlich erweisen; diese Funktionen sind im Abschnitt [Grid-Ansicht](#) beschrieben.
- Die Funktionen der [Textansicht](#), die für die Bearbeitung von JSON-Dokumenten verwendet werden können, sind unten aufgelistet.

Anmerkung: XMLSpy unterstützt auch die Konvertierung zwischen JSON- und XML-Dokumenten in beiden Richtungen. Wenn Sie in XMLSpy auf den Befehl [Von/in JSON konvertieren](#) klicken, wird das aktive Dokument in das andere Format konvertiert.

Klappleisten: Jedes JSON-Element kann durch Klicken auf das Minus- bzw. Plussymbol links vom Lineal (*siehe Abbildung unten*) reduziert bzw. erweitert werden. Beachten Sie, dass auch die Zeilennummerierung aktiviert ist.

```

10  "OrgChart": {
11      "xmlns": "http://www.xmlspy.com/schemas/orgchart",
12      "xmlns:ipo": "http://www.altova.com/IPD",
13      "xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",
14      "xmlns:ts": "http://www.xmlspy.com/schemas/textstate",
15      "xsi:schemaLocation": "http://www.xmlspy.com/schemas/orgchart OrgChart.xsd",
16      "CompanyLogo": {
17          "href": "nanonull.gif"
18      },

```

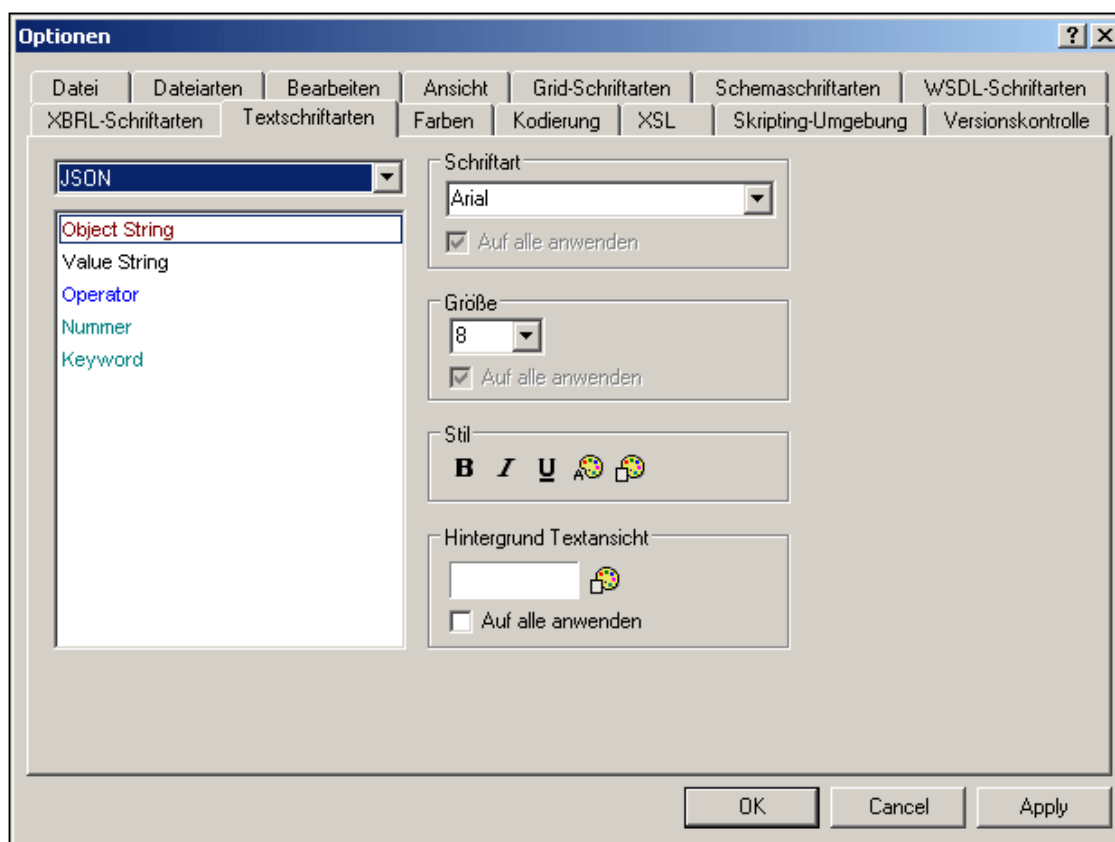
Strukturmarkierungen: Die zwei geschwungenen Klammern bzw. die eckigen Klammern, in die ein JSON-Element eingeschlossen ist (*Abbildung unten*) werden fett angezeigt, wenn der Cursor entweder vor oder hinter eine dieser Klammern platziert wird. Dies zeigt an, wo die Definition eines bestimmten Elements beginnt und endet.

```

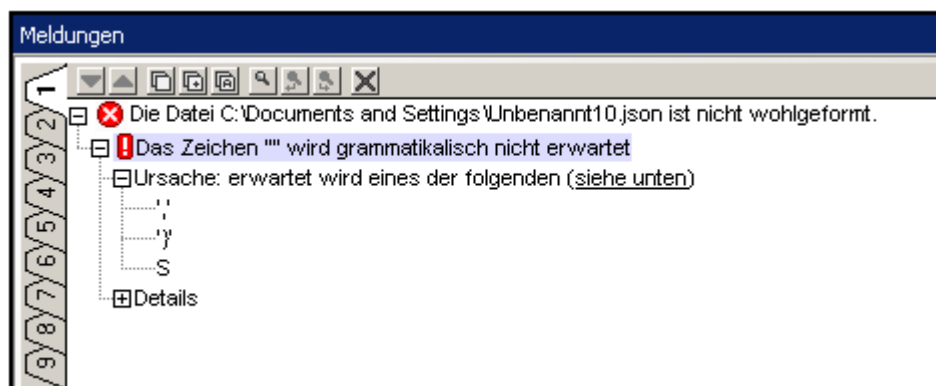
"Department": [ { "Name": "Administration",
"Person": [ { "First": "Vernon",
"Last": "Callaby",
"Title": "Office Manager",
"PhoneExt": 582,
"EMail": "v.callaby@nanonull.com",
"Shares": 1500,
"LeaveTotal": 25,
"LeaveUsed": 4,
"LeaveLeft": 21}],

```

Syntaxfärbung: Ein JSON-Dokument besteht aus Objektstrings, Wertestings, Operatoren, Zahlen und Schlüsselwörtern. In der Textansicht kann jede Elementkategorie gemäß den im [Dialogfeld "Optionen"](#) vorgenommenen Einstellungen in einer anderen Farbe angezeigt werden (*Abbildung oben*). Sie können die Farben der verschiedenen JSON-Komponenten im Dialogfeld "Optionen" auf dem Register "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.



Syntaxprüfung: Die Syntax eines JSON-Dokuments 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: Zwei doppelte Anführungszeichen (ein schließendes Anführungszeichen gefolgt von einem öffnenden) kommen nacheinander vor; von der Syntax her muss zwischen zwei Anführungszeichen ein Komma oder ein geschwungene schließende Klammer stehen.

Konvertierung zwischen XML und JSON

Der Befehl **Von/in JSON konvertieren** im Menü **Konvertieren** konvertiert ein XML-Dokument, wenn es das aktive Dokument ist, in ein JSON-Dokument. Außerdem kann ein aktives JSON-

Dokument in ein XML-Dokument konvertiert werden. Nähere Informationen finden Sie in der [Benutzerreferenz](#).

10 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 Archiv-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 Archiv-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 Dateiarart. Es gibt jedoch eine gemeinsam Struktur: eine XML-Datei namens

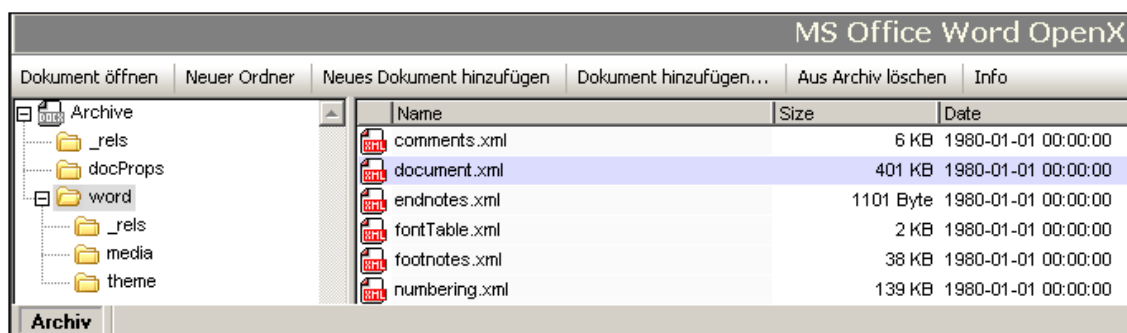
[Content_Types].xml an der Root der Verzeichnisstruktur und drei Verzeichnisse: _rels, docProps und ein Verzeichnis für die Dokumentart (im Fall von .docx-Dokumenten hieße dieser Ordner z.B. word; in .xlsx-Dokumenten xl und in .pptx-Dokumenten ppt).

```
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 Archiv-Ansicht von XMLSpy können ZIP-Dateien erstellt, die innere Struktur geändert und Dateien im Archiv bearbeitet werden. Diese Vorgänge werden im Unterabschnitt ZIP-Dateien 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 Archiv-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](#).

10.1 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 Archiv-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/XMLSpy2012/Examples` finden Sie EPUB-Beispieldateien.

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 Archiv-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.

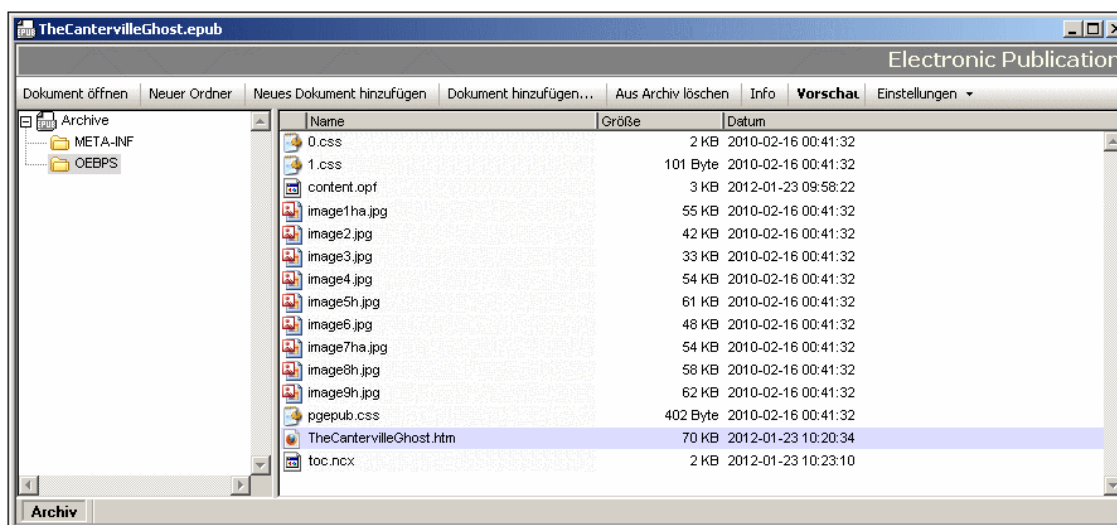
Dieser Abschnitt

Die Beschreibung der EPUB-Funktionen in XMLSpy ist in die folgenden Unterabschnitte gegliedert:

- [Öffnen von EPUB-Dateien in der Archiv-Ansicht](#)
- [Erstellen einer neuen EPUB-Datei](#)
- [Anzeigen einer Vorschau eines EPUB-Buchs](#)
- [Ändern des Inhalts und der Struktur eines EPUB-Archivs](#)
- [Bearbeiten von Archivdateien direkt in XMLSpy](#)
- [Eingabehilfen für Archivdateien](#)
- [Validieren der EPUB-Datei](#)

Öffnen von EPUB-Dateien in der Archiv-Ansicht

Wählen Sie den Menübefehl **Datei | Öffnen**, navigieren Sie zur EPUB-Datei und klicken Sie auf **Öffnen**. Daraufhin wird die EPUB-Datei in der Archiv-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](#), 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 Archiv-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 Archiv-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 Archiv-Ansicht zur aktiven und klicken Sie dann in der Symbolleiste der Archiv-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 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 Archiv-Ansicht) können Sie festlegen, dass alle geänderten Dateien vor der Vorschau automatisch gespeichert werden.

Beachten Sie Folgendes:

- Wenn Sie auf die Schaltfläche **Vorschau** klicken, während ein Vorschaufenster des EPUB-Buchs noch geöffnet ist, so wird das offene Vorschaufenster aktualisiert und neu geladen.
- Wenn Sie das Vorschaufenster mit "Aktualisieren" aktualisieren, wird die Vorschau nicht neu geladen. 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 Archiv-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.

Bearbeiten von Archivdateien direkt in XMLSpy

Um eine Archivdatei direkt in XMLSpy zu bearbeiten, doppelklicken Sie in der Archiv-Ansicht auf die Datei. Alternativ dazu können Sie die Datei in der Archiv-Ansicht auswählen und in der Symbolleiste der Archiv-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

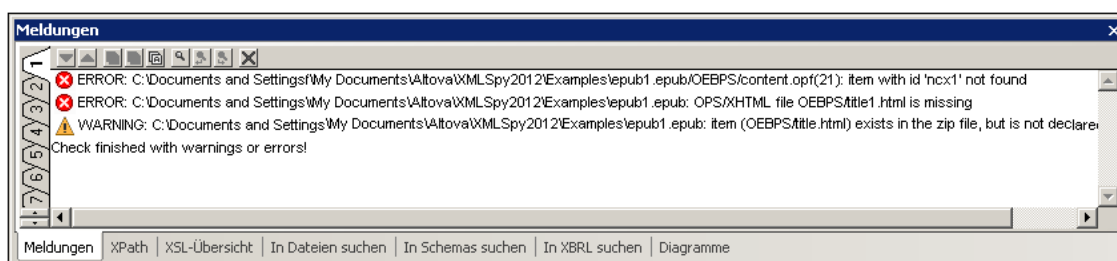
Die Eingabehilfen für standardbasierte Archivdateien stehen zur Verfügung, wenn diese Archivdateien in XMLSpy geöffnet werden. Diese Archivdateien sind:

- 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 Archiv-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.

11 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 einer 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 [Benutzerreferenz](#) 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](#): Dies ist eine Voraussetzung, um einen der XMLSpy Befehle für die Arbeit mit Datenbanken ausführen zu können.
- [Liste aller Datenbanken](#), 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](#) und [Konvertieren](#) ausgeführt.

- [Datenbank abfragen](#): 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](#): 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.
- [Oracle XML DB](#): 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](#): Im Menü **Konvertieren**. Importiert Datenbankdaten in eine XML-Datei.
- [XML-Schema von DB-Struktur erzeugen](#): Im Menü **Konvertieren**. Generiert anhand der Struktur einer Datenbank ein XML-Schema.
- [DB-Import auf Basis von XML-Schema](#): 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](#): Im Menü **Konvertieren**. Datenbanktabellen ohne Daten werden auf Basis der Struktur des bestehenden XML-Schemas erzeugt.
- [Export in Datenbank](#): 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 [Datatypes in DB auf XML-Schema](#) und [Datentypen in XML-Schema auf DB](#)

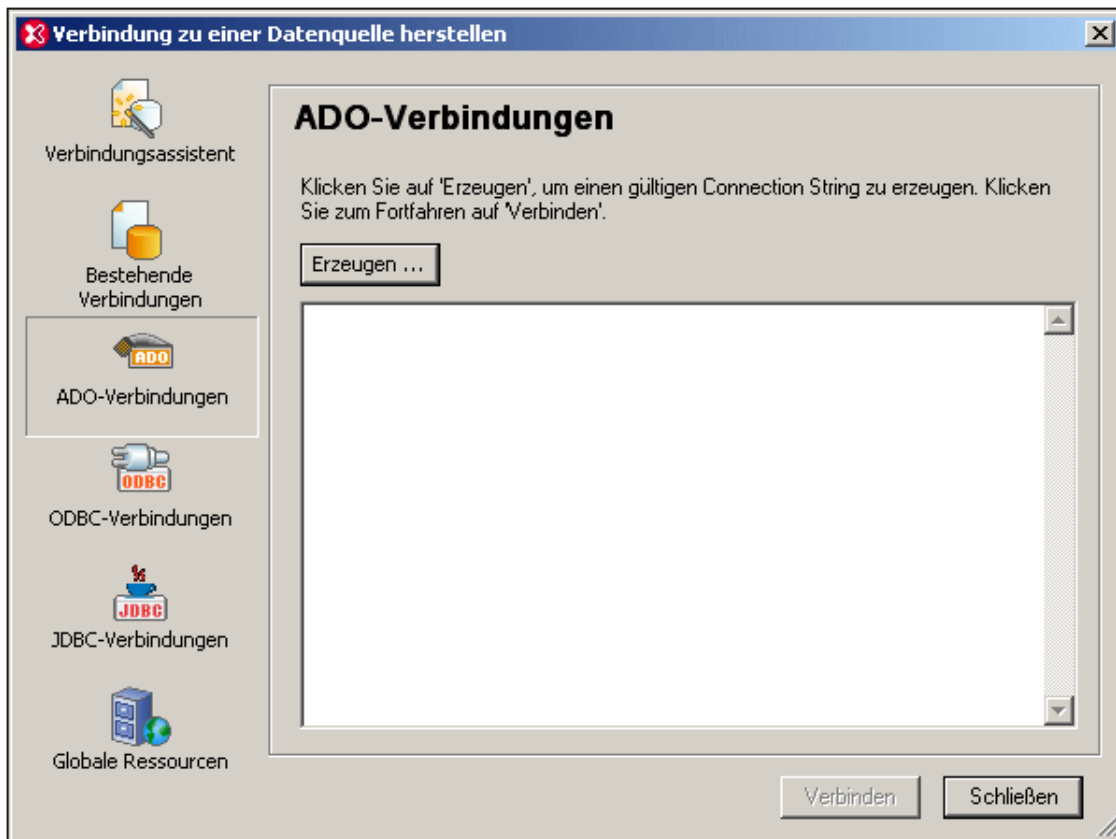
wird beschrieben, wie Datentypen in XMLSpy konvertiert werden.

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](#).

11.1 Herstellen einer Verbindung zu einer Datenquelle

Für eine Reihe von Befehlen in den Menüs **DB** und **Konvertieren** muss eine Verbindung zu einer Datenbank hergestellt werden. XMLSpy unterstützt die Verbindung mit verschiedenen Datenbanken (*siehe Liste unten*). Die Verbindung wird über das Dialogfeld "Verbindung zur Datenquelle herstellen" oder das Dialogfeld "Verbindungsassistent" (*Abbildung unten*) konfiguriert und hergestellt. Beide Dialogfelder sind im Wesentlichen gleich. Sie führen Sie durch die einzelnen Schritte. In diesem Abschnitt finden Sie Informationen über die Funktionsweise dieser Dialogfelder. In der Beschreibung der **Datenbank-** und **Konvertierungsbefehle**, bei denen eine Datenbankverbindung erforderlich ist, finden Sie in diesem Abschnitt nähere Informationen über das Herstellen der Verbindung.



Die Optionen zum Herstellen einer Verbindung zur Datenbank sind:

- [Über einen Verbindungsassistenten](#), der Sie durch den Vorgang führt.
- Über eine [bestehende Verbindung](#)
- [Über eine ADO-Verbindung](#)
- [Über eine ODBC-Verbindung](#)
- Über eine [JDBC-Verbindung](#)
- Über eine [globale Ressource](#).

Die einzelnen Optionen werden in den Unterabschnitten dieses Abschnitts näher beschrieben. In diesem Abschnitt wird nur die Herstellung einer Verbindung beschrieben. Die Dialogfelder, die anschließend angezeigt werden, sind abhängig vom jeweils ausgeführten Befehl und werden daher in den Abschnitten zu den einzelnen Befehlen beschrieben. Wenn Sie z.B. eine Beschreibung zum Befehl [Konvertieren | Datenbankdaten importieren](#) suchen, finden Sie die

Anleitung zum Herstellen der Verbindung in diesem Abschnitt und die Informationen zur Auswahl der DB-Tabellen im Abschnitt [Konvertieren | Datenbankdaten importieren](#).

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.

11.1.1 Verbindungsassistent

Bei Auswahl der Schaltfläche "Verbindungsassistent" im Dialogfeld "Verbindung zu einer Datenbank herstellen" wird der Verbindungsassistent (*siehe Abbildung unten*) aufgerufen. Der Verbindungsassistent hilft Ihnen, eine Verbindung zu den gängigsten Datenbanktypen herzustellen. In diesem Abschnitt wird die Herstellung der Verbindung zu einer Microsoft Access Datenbank und einer IBM DB 2-Datenbank beschrieben.

MS Access

Um eine Verbindung zu einer MS Access-Datenbank herzustellen, führen Sie die folgenden Schritte durch. Wählen Sie Microsoft Access (ADO) (*Abbildung unten*) und klicken Sie auf **Weiter**.



Suchen Sie im Dialogfeld "Verbindung zu MS Access herstellen" die MS Access-Datenbank und klicken Sie auf **Weiter**. Die Verbindung zur Datenquelle wird hergestellt. Informationen über Dialogfelder, die in der Folge angezeigt werden, finden Sie unter der Beschreibung des ausgeführten Befehls.

IBM DB2

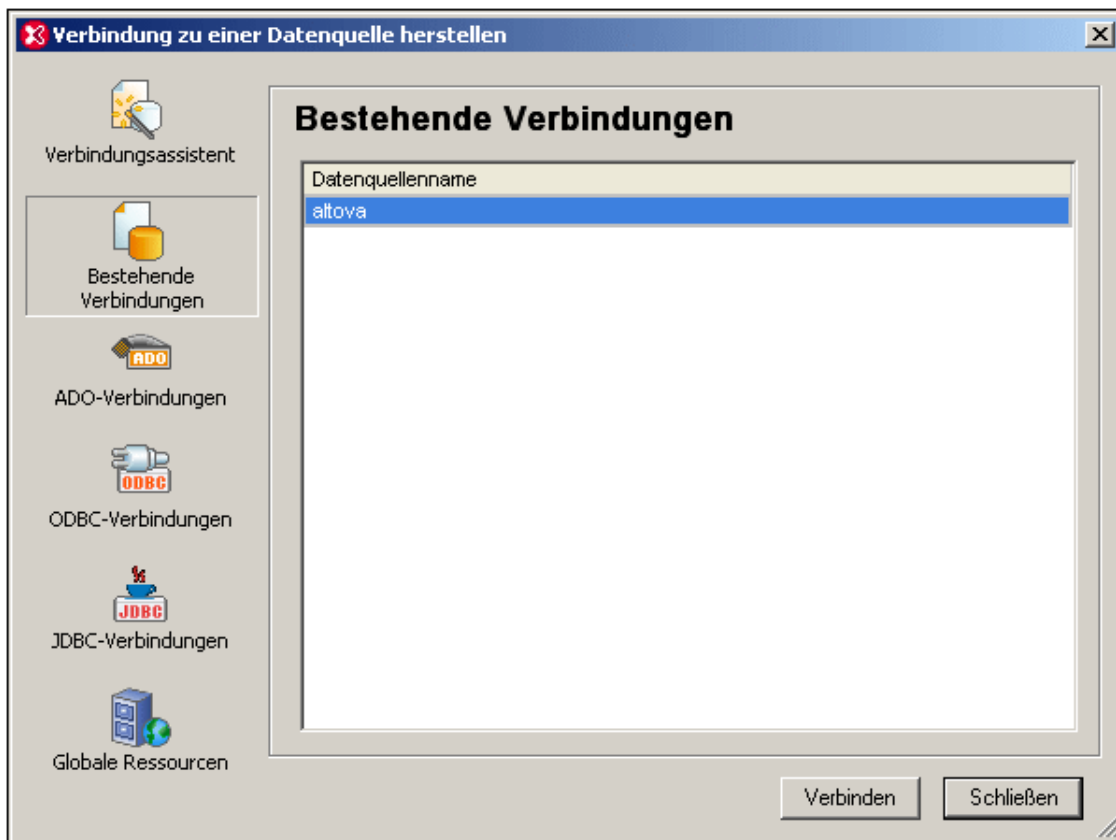
Wählen Sie im ersten Bildschirm des Verbindungsassistenten (*siehe erste Abbildung in diesem Abschnitt*) den Eintrag IBM DB2 und klicken Sie auf **Weiter**. Daraufhin wird das Konfigurationsdialogfeld (*Abbildung unten*) angezeigt. Der Assistent führt Sie anschließend durch den Konfigurationsvorgang für die IBM DB2-Datenbank und stellt die Verbindung her. Dabei wird zuerst der entsprechende Treiber für die Verbindung mit der DB ausgewählt, anschließend die DB und schließlich werden die für die Verbindung erforderlichen Benutzerinformationen eingegeben.



Wählen Sie im Konfigurationsdialogfeld oben eine bestehende Datenquelle aus oder erstellen Sie eine neue Datenquelle mit einem entsprechenden Treiber (Zusätzliche Treiber können zur Liste der verfügbaren Treiber durch Klicken auf die Schaltfläche **Treiber bearbeiten** und Auswahl des gewünschten Treibers hinzugefügt werden). Im folgenden Bildschirm werden Sie aufgefordert, die Benutzerdaten (ID und Passwort) einzugeben. Wenn Sie auf **OK** klicken, wird die Verbindung zur Datenbank hergestellt. Nähere Informationen finden Sie auch unter [ODBC-Verbindungen](#). Informationen über Dialogfelder, die in der Folge angezeigt werden, finden Sie unter der Beschreibung des ausgeführten Befehls.

11.1.2 Bestehende Verbindungen+

Bei Auswahl der Schaltfläche "Bestehende Verbindungen" im Dialogfeld "Verbindung zu einer Datenbank herstellen" wird das Fenster "Bestehende Verbindungen" (*Abbildung unten*) geöffnet, in dem alle derzeit bestehenden Verbindungen angezeigt werden.

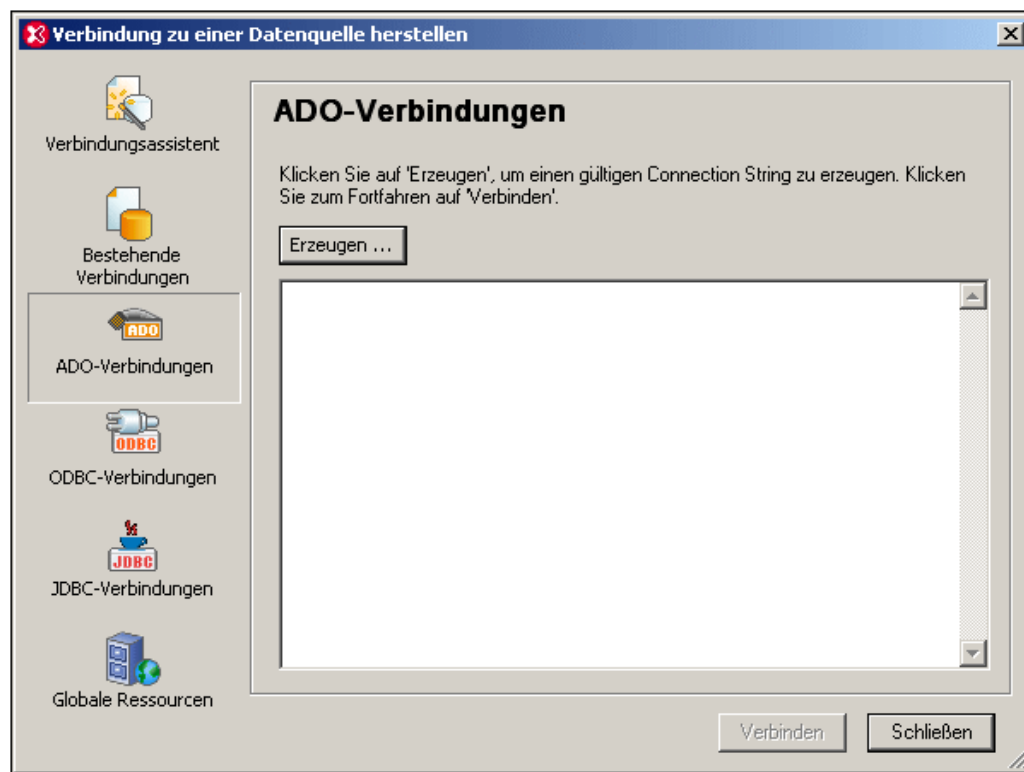


Wählen Sie die gewünschte Datenbank aus und klicken Sie auf **Verbinden**. Die Verbindung zur Datenbank wird hergestellt. Informationen über Dialogfelder, die in der Folge angezeigt werden, finden Sie unter der Beschreibung des ausgeführten Befehls.

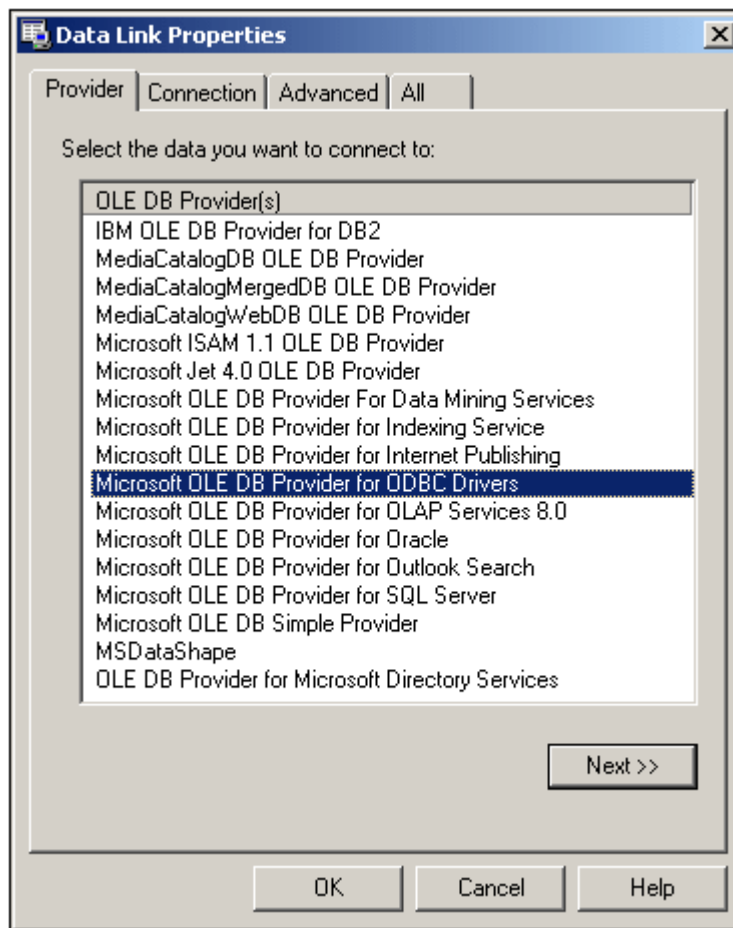
11.1.3 ADO-Verbindungen

Über die Option "ADO-Verbindungen" können Sie über ADO eine Verbindung zu einer Datenbank herstellen. In diesem Abschnitt wird die Herstellung einer Verbindung über ADO anhand einer IBM DB2-Datenbank als Ziel erklärt. Dort wo, sich die Optionen bei Verwendung einer anderen Datenbankart als Zieldatenbank unterscheiden, wird dies erwähnt.

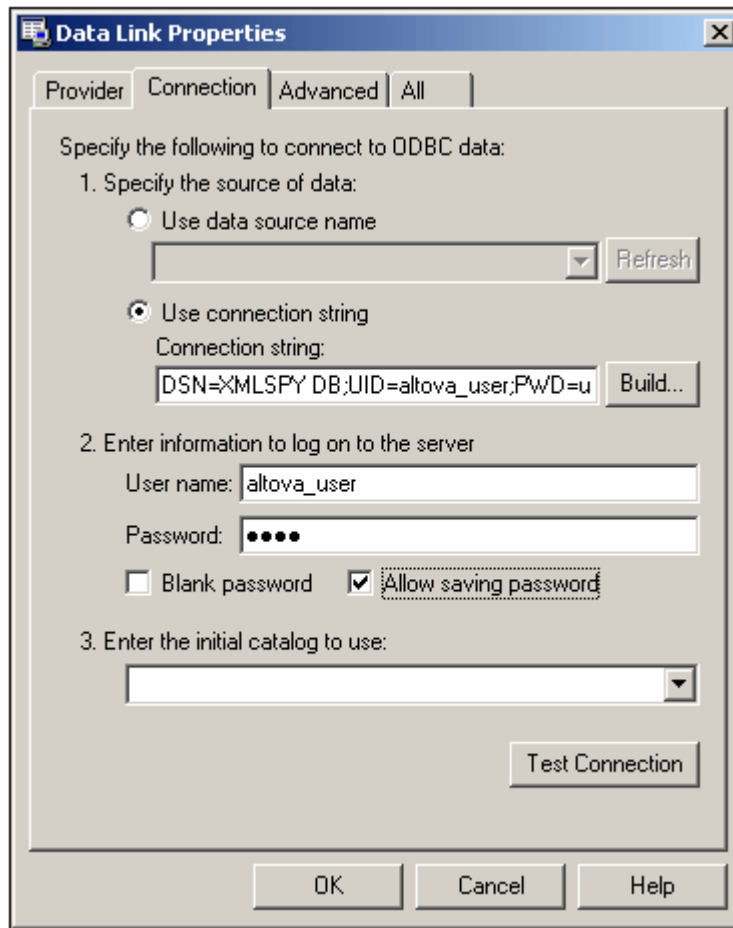
1. Wählen Sie im Dialogfeld "Verbindung zur Datenquelle herstellen" die Option "ADO-Verbindungen" aus. Daraufhin wird das Dialogfeld "ADO-Verbindungen" angezeigt (*Abbildung unten*).



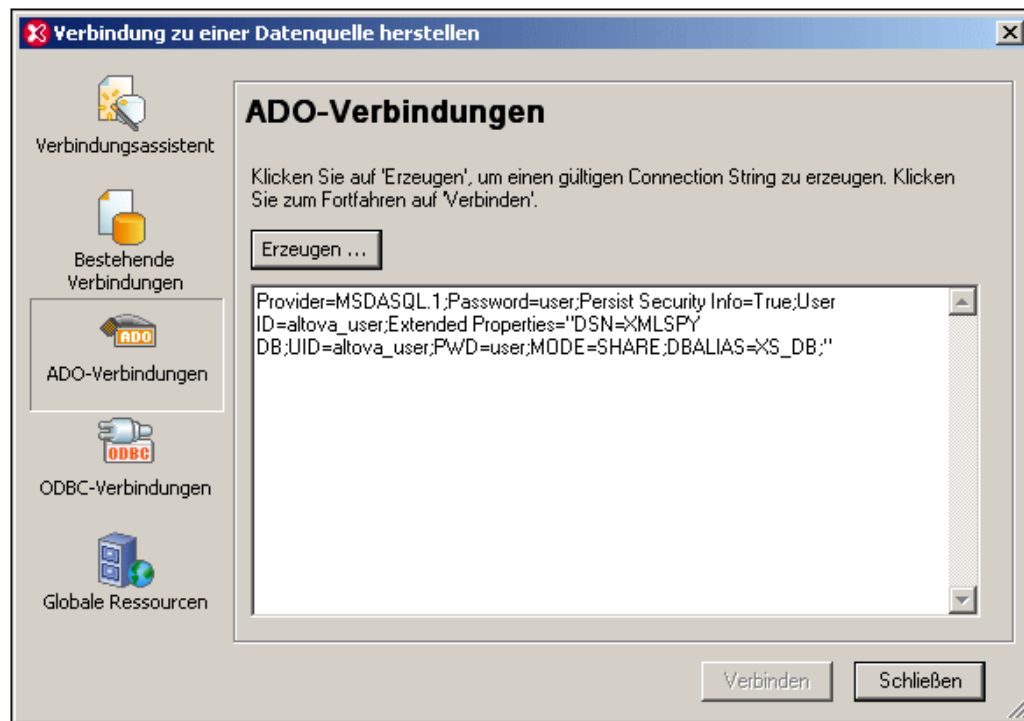
2. Klicken Sie auf die Schaltfläche **Erzeugen**. Das Dialogfeld "Datenverknüpfungseigenschaften" (*Abbildung unten*) wird geöffnet.



3. Wählen Sie "Microsoft OLE DB Provider for ODBC Drivers" in der Liste aus und klicken Sie auf **Next**. Das Register "Verbindung" ist aktiv. (Bei Microsoft SQL-Datenbank empfehlen wir die Auswahl von *Microsoft OLE DB Provider for SQL Server*; bei Oracle, MySQL, Sybase und IBM DB2-Datenbanken empfehlen wir *Microsoft OLE DB Provider for ODBC Drivers*.)



4. Geben Sie über die Schaltfläche **Erzeugen** einen Connection String ein. (In dem Dialogfeld, das erscheint, werden Sie aufgefordert die Datenquelle und Benutzerdaten einzugeben). Bei einigen Datenbanken wird kein Connection String erstellt, sondern der Server und die Datenbank werden aus Auswahllisten ausgewählt.
5. Falls Login-Informationen benötigt werden, geben Sie den Benutzernamen und das Passwort ein und aktivieren Sie das Kontrollkästchen "Speichern des Kennworts zulassen".
6. Klicken Sie auf **OK**. Das Dialogfeld "Datenverknüpfungseigenschaften" wird geschlossen. Der Connection String wird im Feld "ADO-Verbindungen" angezeigt (Abbildung unten).



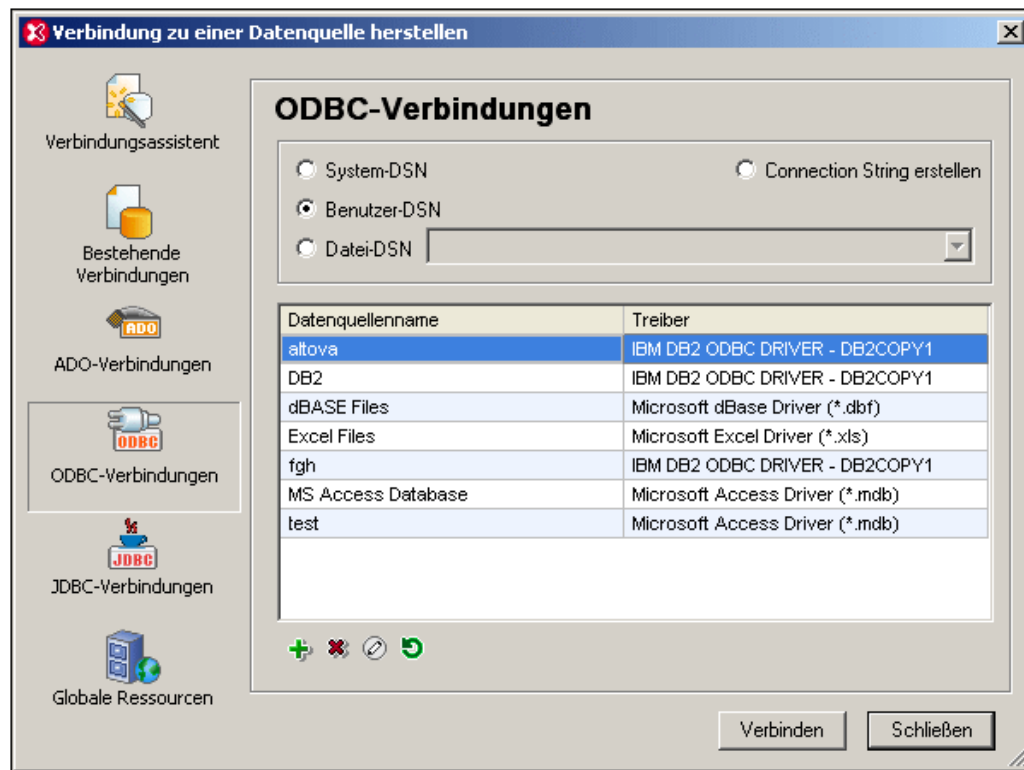
7. Klicken Sie auf die Schaltfläche **Verbindung herstellen**. Daraufhin wird eine Verbindung zur Datenquelle hergestellt. Informationen über Dialogfelder, die in der Folge angezeigt werden, finden Sie unter der Beschreibung des ausgeführten Befehls.

Anmerkung: Für eine ADO-Verbindung zu MS SQL Server über den Treiber SQL Server Native Client 10.0 müssen die folgenden Eigenschaftswerte auf dem Register *All* des Dialogfelds "Data Link Properties" definiert sein: (i) Setzen Sie den Eigenschaftswert von Integrated Security auf ein Leerzeichen, (ii) Setzen Sie den Eigenschaftswert von Persist Security Info auf `true`.

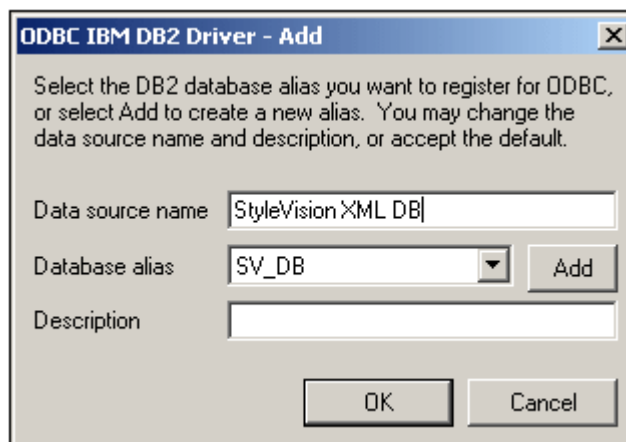
11.1.4 ODBC-Verbindungen

In diesem Abschnitt wird beschrieben, wie Sie über ODBC eine Datenbankverbindung herstellen. In der Anleitung unten wird eine Verbindung zu einer IBM DB2-Datenbank hergestellt. Dieselben Schritte werden aber auch bei anderen Datenbankarten, zu denen über ODBC eine Verbindung hergestellt werden kann, ausgeführt. Um eine Verbindung zu einer IBM DB2-Datenbank über ODBC herzustellen, gehen Sie folgendermaßen vor:

1. Wählen Sie im Dialogfeld "Verbindung zu einer Datenquelle herstellen" die Option **ODBC-Verbindungen** aus.



- Wählen Sie eine der Optionen zur Definition eines Datenquellennamens (DSN). Wenn Sie "System-DSN" oder "Benutzer-DSN" wählen, werden die verfügbaren DSNs im Fenster "Datenquelle" angezeigt. Bei Auswahl von "Datei-DSN" werden Sie aufgefordert, die DSN-Datei auszuwählen. Alternativ dazu können Sie einen Connection String zur DB erzeugen, indem Sie die Option "Connections String erstellen" auswählen.
- Wenn Sie einen DSN zu den DSNs in der Liste der Datenquellen hinzufügen möchten, klicken Sie auf die Schaltfläche "Neuen DSN erstellen" .
- Wählen Sie im Dialogfeld "ODBC DSN erstellen" den gewünschten Treiber aus und klicken Sie anschließend auf die Schaltfläche **Benutzer DSN** oder **System-DSN**.
- Wählen Sie im Dialogfeld, das daraufhin angezeigt wird (*Abbildung unten*), den DB-Alias aus und geben Sie ihm einen DSN.



- Klicken Sie zum Fertigstellen auf **OK**. Die DB wird als Datenquelle zur Liste im Bereich

- "Datenquelle" hinzugefügt.
7. Nachdem Sie eine Datenquelle (über die Option "System-DSN" oder "Benutzer-DSN") oder eine DSN-Datei (über die Option "Datei-DSN") ausgewählt, oder einen Connection String (über die Option "Connection String") erstellt haben, klicken Sie auf "Verbindung herstellen".
 8. Wenn Sie nach Ihrer Benutzer-ID und dem Passwort gefragt werden, geben Sie diese ein und klicken Sie auf **OK**. Daraufhin wird die Verbindung hergestellt. Informationen über daraufhin angezeigte Dialogfelder finden Sie unter der Beschreibung des ausgeführten Befehls.

Anmerkung: Die Liste im Fenster "Datenquelle" kann mittels der Schaltflächen *Den ausgewählten DSN bearbeiten*, *Den ausgewählten DSN entfernen* und *Aufgelistete DSNs aktualisieren* am unteren Rand des Fensters "ODBC-Verbindungen" bearbeitet werden (wenn die Option "System-DSN" oder "Benutzer-DSN" ausgewählt ist).



Ausgewählten DSN bearbeiten.



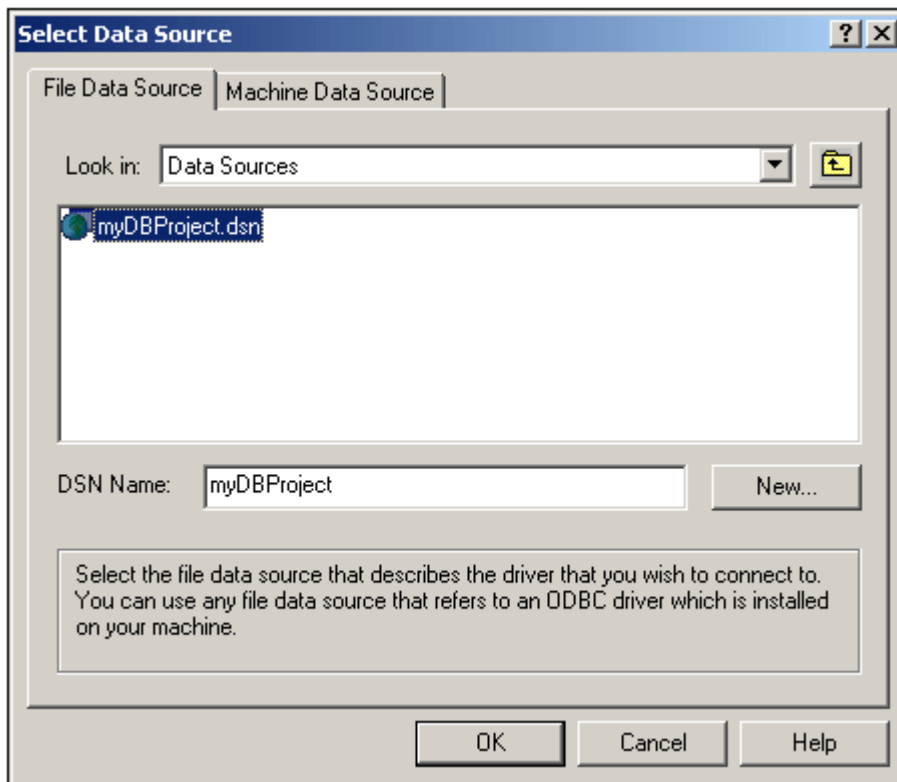
Ausgewählten DSN entfernen.



Aufgelistete DSNs aktualisieren.

Erstellung eines Connection String

Wenn Sie im Bereich "ODBC-Verbindungen" auf das Optionsfeld "Connection String erstellen" klicken und auf die daraufhin angezeigte Schaltfläche "Erzeugen", erscheint das Dialogfeld "Select Data Source" (*Abbildung unten*). Sie können entweder einen Datei-DSN (auf dem Register "File Data Source") oder eine Datenquelle auswählen, die auf Ihrem Rechner zur Verfügung steht (wird auf dem Register "Machine Data Source" aufgelistet).



Wenn Sie auf **OK** klicken, erscheint ein Dialogfeld, in dem Sie nach Benutzerdaten wie u.a.

Benutzer-ID und Passwort gefragt werden. Wenn Sie in diesem Dialogfeld auf **OK** klicken, wird der Connection String in den Bereich "ODBC-Verbindungen" eingegeben.

11.1.5 JDBC-Verbindungen

In diesem Abschnitt wird beschrieben, wie Sie über JDBC eine Verbindung zu einer Datenbank herstellen. In der Anleitung unten, wird beschrieben, wie Sie eine Verbindung zu einer IBM DB2-Datenbank herstellen. Die Anleitung gilt aber auch für andere Arten von Datenbanken, die über JDBC verbunden werden können.

Schritte vor dem Herstellen der Verbindung

Bevor Sie über JDBC eine Verbindung zu einer Datenbank herstellen, gehen Sie folgendermaßen vor:

1. Vergewissern Sie sich, dass das Java Runtime Environment (JRE) installiert ist. Ist dies nicht der Fall, dann installieren Sie es. Verwenden Sie für einen 32-Bit-Rechner ein 32-Bit JRE und für einen 64-Bit-Rechner ein 64-Bit JRE.
2. Installieren Sie einen JDBC-Treiber. Dazu ist keine spezielle Installation vonnöten. Kopieren Sie einfach den Treiber in ein lokales Verzeichnis, z.B. `c:\jdbc`. Beachten Sie, dass JDBC-Treiber (d.h. Jar-Dateien) plattformunabhängig sind.
3. Richten Sie den `CLASSPATH` ein, sodass er den Pfad enthält, unter dem der JDBC-Treiber gespeichert ist. (Die Applikation liest die `CLASSPATH`-Umgebungsvariable um den JDBC-Treiber zu finden.) Um den `CLASSPATH`, aufzurufen und zu bearbeiten, gehen Sie folgendermaßen vor: Klicken Sie auf **Start | Systemsteuerung | System | Erweitert | Umgebungsvariablen**. Wählen Sie im Dialogfeld "Umgebungsvariablen", das daraufhin angezeigt wird, entweder die Benutzerumgebungsvariable `CLASSPATH` oder die Systemumgebungsvariable `CLASSPATH` aus und klicken Sie auf die Schaltfläche **Bearbeiten**. Fügen Sie den Pfad zum JDBC-Treiber zum `CLASSPATH` hinzu. Beispiel: `CLASSPATH=C:\jdbc\sqljdbc.jar; C:\jdbc\db2jcc.jar; .`
4. Melden Sie sich beim System ab und dann wieder an, damit die `CLASSPATH` Änderungen aktiv werden.

Anmerkung: Wenn Sie den kompletten IBM DB2- oder Oracle Client installieren, wird der `CLASSPATH` automatisch mit den JDBC-Treibern der entsprechenden Pakete befüllt. Nähere Informationen finden Sie in den Beschreibungen weiter unten.

Herstellen einer Verbindung über JDBC

Um über JDBC eine Verbindung herzustellen, gehen Sie folgendermaßen vor:

1. Wählen Sie im Dialogfeld "Verbindung zu einer Datenquelle herstellen" die Option **JDBC-Verbindungen** aus. Daraufhin wird das Fenster "JDBC-Verbindungen" angezeigt (*Abbildung unten*).



2. Wählen Sie aus der Dropdown-Liste *Treiber* einen JDBC-Treiber aus (alle gefundenen Treiber, d.h. alle, die sich in der `CLASSPATH` Umgebungsvariablen befinden, werden aufgelistet). Geben Sie in das Textfeld *Datenbank-URL* einen Connection String ein und geben Sie gegebenenfalls einen Benutzernamen und ein Passwort ein. Im Folgenden finden Sie die Connection String-Syntax für gebräuchliche Datenbanken. Zu jeder Datenbank ist ein Beispiel-Connection String aufgelistet.

Oracle `jdbc:oracle:thin:[ user/password]@[ host][: port]/SID`
 `jdbc:oracle:thin:@//abcd234/ORA11`

IBM DB2 `jdbc:db2://host_name:port/dbname`
 `jdbc:db2://MyDB2:50000/boz`

MySQL `jdbc:mysql://host_name:port/dbname`
 `jdbc:mysql://MyDB2:3306/moz`

MSSQL `jdbc:sqlserver://host:port;databasename=name; user=name; password=Pwd`
 `jdbc:sqlserver://abcd38:1433;databasename=coz`
 `jdbc:sqlserver://Q5;DatabaseName=coz;SelectMethod=Cursor`

PostgreSQL jdbc:postgresql://host:port/database
L

jdbc:postgresql://abc993:5432/qanoz

Sybase jdbc:sybase:Tds:host:port/dbname

jdbc:sybase:Tds:abc12:2048/QUE

3. Klicken Sie auf die Schaltfläche **Verbinden**. Daraufhin wird die Verbindung zur Datenquelle hergestellt. Informationen über die nachfolgenden Dialogfelder finden Sie in der Beschreibung zum ausgeführten Befehl.

Schritt für Schritt: MSSQL, MySQL, PostGre und andere Nicht-XML-Datenbanken

Unten finden Sie eine schrittweise Anleitung für die Herstellung einer Verbindung zu einer Nicht-XML-Datenbank über JDBC.

1. JDBC JAR-Dateien (Treiberdateien): Kopieren Sie die Dateien in einen lokalen Ordner auf Ihrem Rechner. Fügen Sie anschließend zur Windows `CLASSPATH` Variablen den vollständigen Pfad und Dateinamen hinzu, z.B.: `C:\jdbc\sqljdbc.jar`;
`C:\jdbc\db2jcc.jar`;
2. Melden Sie sich ab und wieder an, um die Änderungen am `CLASSPATH` aktiv zu machen.
3. Starten Sie XMLSpy und rufen Sie das Dialogfeld "Verbindung zu einer Datenbank herstellen" auf.
4. Im Fenster "JDBC-Verbindungen" werden die gefundenen JDBC-Treiber in der Dropdown-Liste *Treiber* aufgelistet. Wenn die Dropdown-Liste leer ist, vergewissern Sie sich, dass die Datei `altovadb.jar` im Ordner `C:\Program Files\Altova\Common 2012\jar` vorhanden ist.
5. Stellen Sie wie im Abschnitt oben unter *Herstellen einer Verbindung über JDBC* beschrieben, eine Verbindung zu einer Datenbank her.

Schritt für Schritt: Oracle

Unten finden Sie eine schrittweise Anleitung für die Herstellung einer Verbindung zu einer Oracle-Datenbank über JDBC. Wenn Sie die XML- und Datenbankfunktionen der Oracle-Datenbank nicht benötigen, befolgen Sie die Anleitung für Nicht-XML-Datenbanken. Der Installationsordner des Oracle Client wird in der Anleitung unten durch den Platzhalter `%ORACLE_HOME%` angegeben.

1. Installieren Sie Oracle Client Software, wobei die OCI- und ODBC-Funktionen aktiviert sind. Wenn bereits ein Oracle Client installiert ist, überprüfen Sie, ob die zwei unten aufgelisteten jar-Dateien vorhanden sind:

`%ORACLE_HOME%\LIB\xmlparserv2.jar`

`%ORACLE_HOME%\RDBMS\jlib\xdb.jar`

2. Fügen Sie die folgenden Dateien zur Windows `CLASSPATH` Umgebungsvariablen hinzu:

`%ORACLE_HOME%\jdbc\lib\ojdbc6.jar`

`%ORACLE_HOME%\LIB\xmlparserv2.jar`

`%ORACLE_HOME%\RDBMS\jlib\xdb.jar`

3. Melden Sie sich ab und wieder an, um die `CLASSPATH` Änderungen aktiv zu machen.
4. Starten Sie XMLSpy und rufen Sie das Dialogfeld "Verbindung zu einer Datenbank

- herstellen" auf.
5. Im Fenster "JDBC-Verbindungen" werden die gefundenen JDBC-Treiber in der Dropdown-Liste *Treiber* aufgelistet. Wenn die Dropdown-Liste leer ist, vergewissern Sie sich, dass die Datei `altovadb.jar` im Ordner `C:\Program Files\Altova\Common 2012\jar.` vorhanden ist.
 6. Stellen Sie wie im Abschnitt oben unter *Herstellen einer Verbindung über JDBC* beschrieben, eine Verbindung zu einer Datenbank her.

Schritt für Schritt: IBM DB2

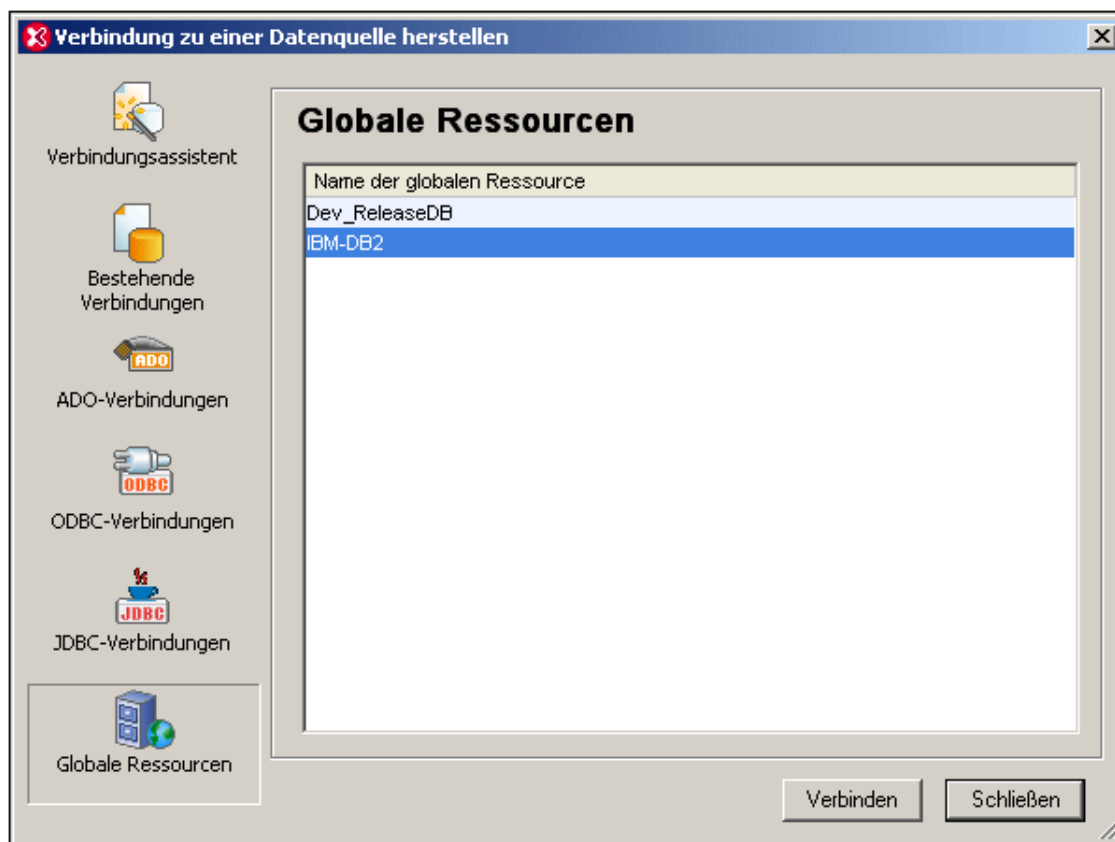
Unten finden Sie eine schrittweise Anleitung für die Herstellung einer Verbindung zu einer IBM DB2-Datenbank über JDBC.

1. Wenn bereits ein IBM DB2 Client installiert wurde, müssen Sie nichts unternehmen, da der `CLASSPATH` bereits während der Installation eingerichtet wurde.
2. Wenn kein IBM DB2 Client installiert wurde, fügen Sie die IBM DB2 JDBC jar-Treiberdateien `db2jcc.jar` und `db2jcc_license_cu.jar` zum Windows `CLASSPATH` hinzu. Melden Sie sich ab und wieder an, um die `CLASSPATH` Änderungen aktiv zu machen.
3. Starten Sie XMLSpy und rufen Sie das Dialogfeld "Verbindung zu einer Datenbank herstellen" auf.
4. Im Fenster "JDBC-Verbindungen" werden die gefundenen JDBC-Treiber in der Dropdown-Liste *Treiber* aufgelistet. Wenn die Dropdown-Liste leer ist, vergewissern Sie sich, dass die Datei `altovadb.jar` im Ordner `C:\Program Files\Altova\Common 2012\jar.` vorhanden ist.
5. Stellen Sie wie im Abschnitt oben unter *Herstellen einer Verbindung über JDBC* beschrieben, eine Verbindung zu einer Datenbank her.

Anmerkung: Wenn eine Datenbankverbindung über JDBC erfolgt, ist aufgrund mangelnder vom Treiber zurückgegebenen Informationen (i) die Bearbeitung von Daten in Tabellen ohne Primärschlüssel nicht möglich; (ii) und die Ausführung einer Datenbearbeitung im SQL-Editor funktioniert nicht.

11.1.6 Globale Ressourcen

Wenn Sie im Dialogfeld "Verbindung zu Datenquelle herstellen" auf die Schaltfläche **Globale Ressourcen** klicken, wird der Bereich "Globale Ressourcen" (*Abbildung unten*) geöffnet. Sie sehen darin alle globalen Ressourcen vom Typ Datenbank, die in der derzeit aktiven [XML-Datei für globale Ressourcen](#) definiert sind.



Wählen Sie die gewünschte globale Ressource aus und klicken Sie auf **Verbinden**. Die Verbindung zur Datenbank wird hergestellt. Informationen zu Dialogfeldern, die in der Folge angezeigt werden, finden Sie in der Beschreibung zum gerade ausgeführten Befehl.

11.2 Unterstützte Datenbanken

Altova XMLSpy bietet vollständige Unterstützung für die unten aufgelisteten Datenbanken. Altova ist bestrebt, auch Unterstützung für andere ODBC/ADO-Datenbanken zu gewährleisten, doch wurde die erfolgreiche Herstellung der Verbindung und Verarbeitung der Daten nur bei den aufgelisteten Datenbanken getestet.

Auch das jeweils verfügbare Root-Objekt für die einzelnen unterstützten Datenbanken wird aufgelistet:

Datenbank (nativ unterstützt)	Root-Objekt
MS SQL Server 2000, 2005 und 2008	database
MS SQL Server 2005 und 2008	schema
Oracle 9i, 10g und 11g	schema
MS Access 2003 und 2007	database
MySQL 4.x und 5.x	database
IBM DB2 8.x und 9	schema
Sybase 12	database
IBM DB2 für i 5.4 und i 6.1	schema
PostgreSQL 8.0, 8.1, 8.2, 8.3	database

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.

12 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.

Bei Verwendung einer globalen Ressource als Input können Sie mittels der Schaltflächen 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 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](#): 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](#): In einer Altova-Applikation 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, und DatabaseSpy.

12.1 Definieren globaler Ressourcen

Globale Altova-Ressourcen werden im Dialogfeld **Globale Ressourcen verwalten** definiert, welches auf zwei Arten aufgerufen werden kann:

- Klicken Sie in der Menüleiste auf "Extras", um das Menü **Extras** anzuzeigen (*Abbildung unten*) und wählen Sie den Befehl **Globale Ressourcen**. Daraufhin wird das Dialogfeld "Globale Ressourcen verwalten" angezeigt.
- Wählen Sie den Menübefehl **Extras | Anpassen**, um das Dialogfeld "Anpassen" aufzurufen. Klicken Sie anschließend auf das Register **Symbolleiste** und aktivieren Sie die Option "Globale Ressourcen". Daraufhin wird die Symbolleiste "Globale Ressourcen" eingeblendet (*Abbildung unten*).



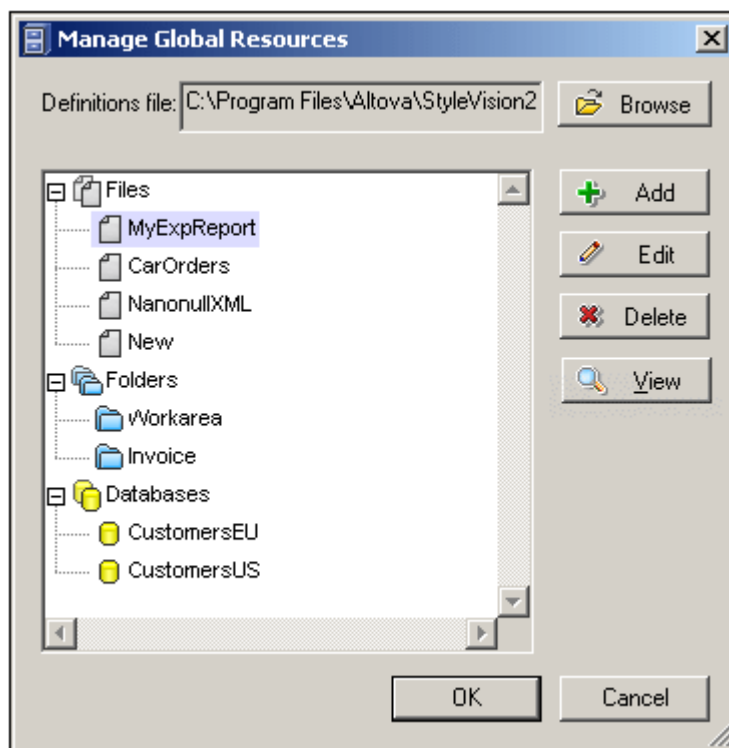
Sobald die Symbolleiste angezeigt wird, klicken Sie auf die Schaltfläche "Globale Ressourcen verwalten". Daraufhin wird das Dialogfeld "Globale Ressourcen verwalten" aufgerufen.

Die XML-Datei für globale Ressourcen

Die Informationen über globale Ressourcen, die definiert wurden, werden in einer XML-Datei gespeichert. Standardmäßig hat diese XML-Datei den Namen `GlobalResources.xml` und ist im Unterverzeichnis `\Altova\` des Ordners (Eigene) Dateien gespeichert. Diese Datei ist bei allen Altova-Applikationen als Standard-XML-Datei für globale Ressourcen definiert. Infolgedessen steht eine globale Ressource, die in einer beliebigen Altova-Applikation definiert wurde, allen Altova-Applikationen zur Verfügung - vorausgesetzt, dass diese Datei in allen Applikationen verwendet wird.

Sie können die Datei umbenennen und in einem beliebigen Ordner speichern. Sie können daher mehrere XML-Dateien für globale Ressourcen haben; es kann jedoch immer nur eine dieser Dateien aktiv sein und nur die in dieser Datei enthaltenen Definitionen stehen der Applikation zur Verfügung.

Um eine XML-Datei für globale Ressourcen als aktive Datei auszuwählen, suchen Sie im Dialogfeld "Globale Ressourcen verwalten" (*Abbildung unten*) im Feld "Definitionsdatei" danach und wählen Sie sie aus.



Verwalten globaler Ressourcen: Hinzufügen, Bearbeiten, Löschen

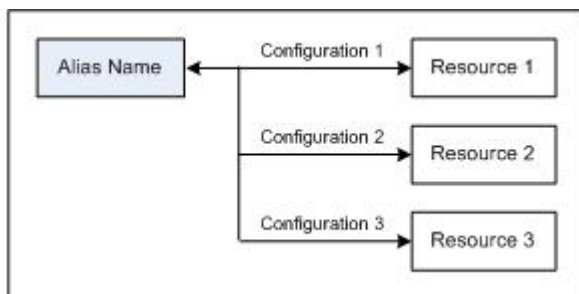
Im Dialogfeld "Globale Ressourcen verwalten" (*Abbildung oben*) können Sie eine globale Ressource zur ausgewählten XML-Datei für globale Ressourcen hinzufügen oder eine ausgewählte globale Ressource löschen oder bearbeiten. In der XML-Datei für globale Ressourcen werden die von Ihnen hinzugefügten Aliasse in Abschnitten 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 Dialogfeld **Globale Ressource**, das angezeigt wird (*Beschreibung siehe oben*). Nachdem Sie eine globale Ressource definiert und gespeichert haben, wird die globale Ressource (Alias) zur Bibliothek der globalen Definitionen in der ausgewählten XML-Datei für globale Ressourcen hinzugefügt. Um eine globale Ressource zu bearbeiten, wählen Sie sie aus und klicken Sie auf **Bearbeiten**. Daraufhin wird das Dialogfeld **Globale Ressource** angezeigt, in dem Sie die nötigen Änderungen vornehmen können (*siehe dazu die Beschreibungen in den Unterabschnitten [Dateien](#), [Ordner](#) und [Datenbanken](#) 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 an der XML-Datei für globale Ressourcen zu speichern.

Hinzufügen einer globalen Ressource

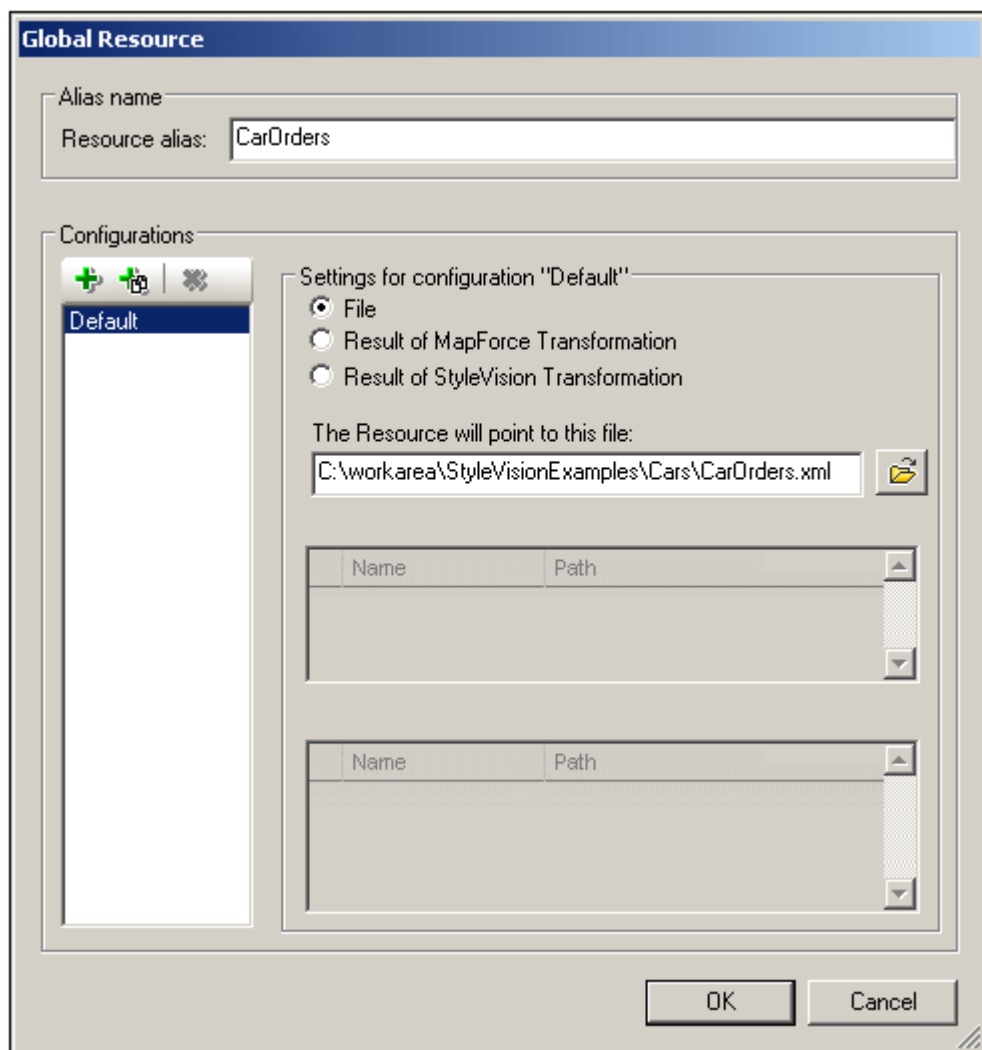
Beim Erstellen einer globalen Ressource wird ein Aliasname auf eine oder mehrere Ressourcen (Datei, Ordner oder Datenbank) gemappt. Jedes Mapping wird als Konfiguration bezeichnet. Ein einziger Aliasname kann daher mittels unterschiedlicher Konfigurationen mit mehreren Ressourcen verknüpft werden (*Abbildung unten*).



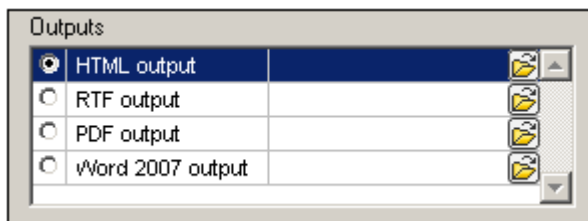
Wenn Sie im Dialogfeld **Globale Ressourcen verwalten** (Abbildung oben) auf die Schaltfläche **Hinzufügen** klicken, können Sie auswählen, ob Sie eine Ressource vom Typ Datei, Ordner oder Datenbank hinzufügen möchten. Wie man die einzelnen Ressourcenarten hinzufügt, ist in den Unterabschnitten dieses Abschnitts beschrieben.

12.1.1 Dateien

Im Dialogfeld **Globale Ressource** (Abbildung unten) können Sie auf folgende Art eine Dateiressource hinzufügen.



1. Geben Sie einen Aliasnamen ein.
2. 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**  und geben Sie im angezeigten Dialogfeld "Konfiguration hinzufügen" den Konfigurationsnamen ein. Klicken Sie auf **OK**. Die neue Konfiguration wird im Fenster "Konfigurationen" aufgelistet. Wiederholen Sie diesen Schritt, um so viele Konfigurationen für diesen Alias (globale Ressource) hinzuzufügen, wie erforderlich. Sie können eine Konfiguration auch (mit Hilfe der Schaltfläche "Konfiguration als Kopie hinzufügen") kopieren und anschließend ändern.
3. Wählen Sie eine Konfiguration im Bereich "Konfigurationen" aus und definieren Sie anschließend die Ressource, auf die die Konfiguration gemappt werden soll. Im Fenster für die Einstellungen der Konfiguration X können Sie auswählen, ob es sich bei der Ressource um eine Datei oder das Ergebnis einer Altova MapForce- oder Altova StyleVision-Transformation handelt. Nachdem Sie den Ressourcentyp durch Klicken auf das entsprechende Optionsfeld ausgewählt haben, suchen Sie die Datei, die MapForce-Datei oder die StyleVision-Datei. Wo mehrere Inputs oder Outputs für die Transformationen möglich sind, stehen Optionen zur Auswahl. Wenn z.B. das Ergebnis einer StyleVision-Transformation als Ressourcentyp ausgewählt wurde, werden, je nachdem welche Edition installiert ist (*in der Abbildung unten sehen Sie die Ausgabeoptionen für die Enterprise Edition*), die verschiedenen Ausgabeoptionen angezeigt.



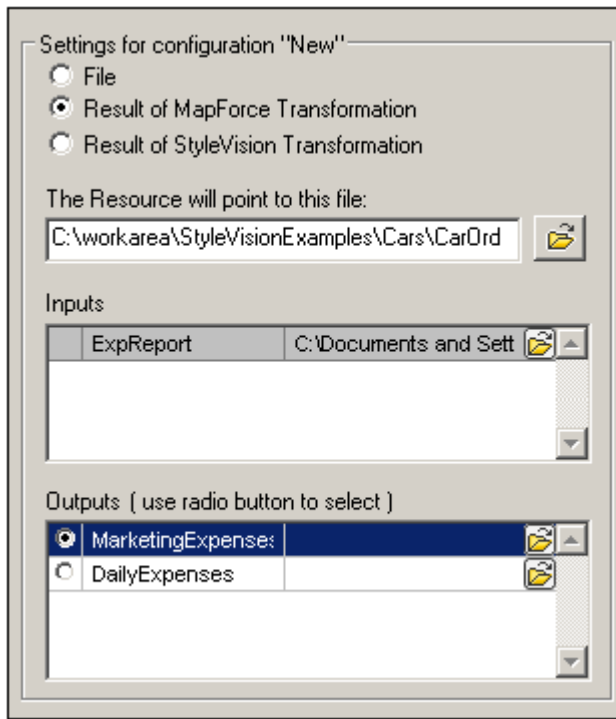
- Aktivieren Sie das gewünschte Optionsfeld (in der Abbildung oben ist "HTML-Ausgabe" ausgewählt). Das Ergebnis einer Transformation kann selbst als globale Ressource oder als Dateipfad (über die Schaltfläche  und anschließendes Auswählen von "Globale Ressource" oder "Durchsuchen") gespeichert werden. Wenn keine dieser beiden Speicheroptionen ausgewählt wird, wird das Transformationsergebnis bei Aufruf der globalen Ressource als temporäre Datei geladen.
4. Definieren Sie eine Dateiressource für jede Konfiguration (d.h. wiederholen Sie Schritt 3 für die verschiedenen von Ihnen erstellten Konfigurationen).
 5. 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 **Globale Ressourcen verwalten** unter "Dateien" aufgelistet.

Auswählen des Ergebnisses einer MapForce-Transformation als globale Ressource

Altova MapForce mappt ein oder mehrere (bestehende) Schemas auf ein oder mehrere (neue) vom MapForce-Benutzer erstellte Schemas. XML-Dateien, die den Input-Schemas entsprechen, werden als Datenquellen verwendet und MapForce kann eine XML-Ausgabedatei generieren, die auf dem vom Benutzer erstellten Schema basiert. Diese generierte Ausgabedatei (Ergebnis der MapForce-Transformation) ist die Datei, die als globale Ressource verwendet wird.

In einer MapForce-Transformation, die mehrere Ausgabeschemas hat, können Sie durch Klicken auf das entsprechende Optionsfeld auswählen, welches der Ausgabeschemas für die globale Ressource verwendet werden soll (*Abbildung unten*). Die XML-Datei, die für dieses

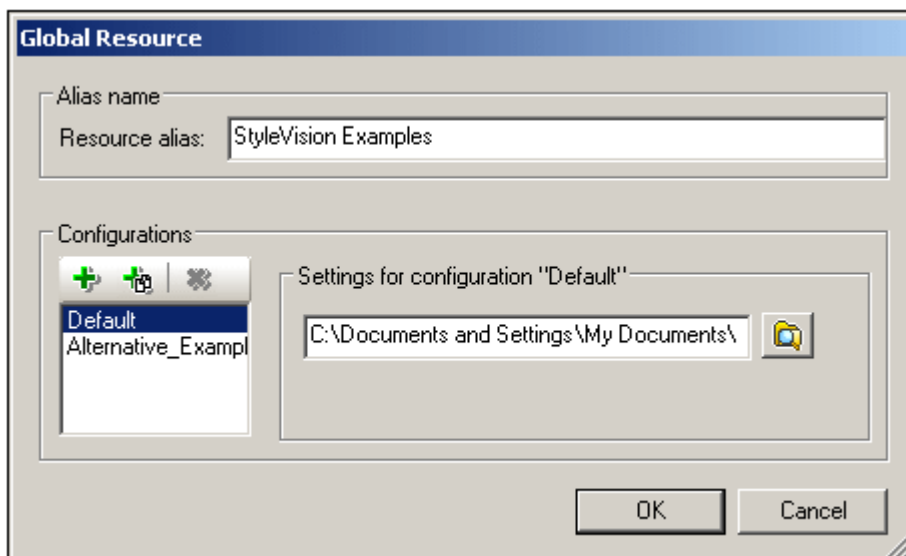
Schema generiert wird, kann als globale Ressource oder (über die Schaltfläche ) als Datei gespeichert werden. Wenn keine dieser Optionen ausgewählt ist, wird bei Verwendung der globalen Ressource eine temporäre XML-Datei erstellt.



Beachten Sie, dass auch jeder Input als globale Ressource oder als Datei (klicken Sie auf die Schaltfläche ) und wählen Sie "Globale Ressource" bzw. "Durchsuchen") gespeichert werden kann.

12.1.2 Ordner

Im Dialogfeld **Globale Ressource** für Ordner (Abbildung unten) können Sie eine Ordnerressource auf folgende Art hinzufügen:

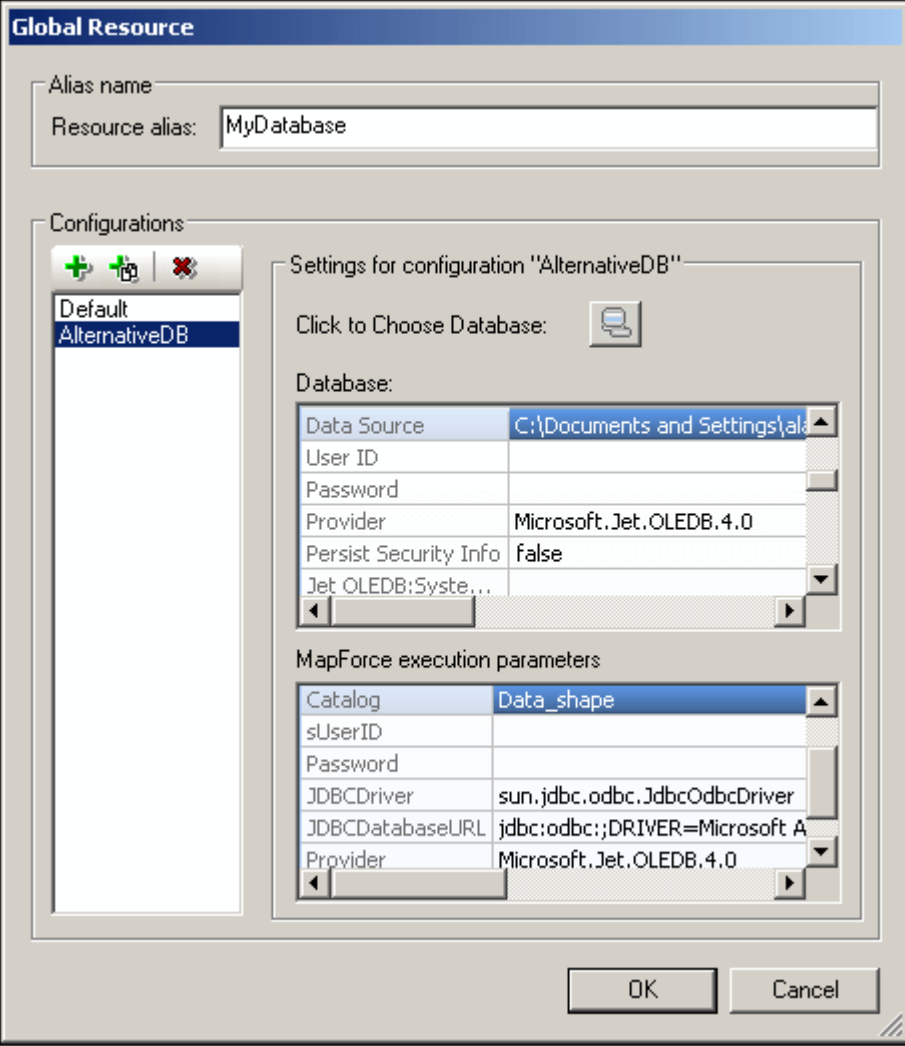


Geben Sie einen Aliasnamen ein.

1. 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**  und geben Sie im angezeigten Dialogfeld **Konfiguration hinzufügen** den Konfigurationsnamen ein. Klicken Sie auf **OK**. Die neue Konfiguration wird im Fenster "Konfigurationen" aufgelistet. Wiederholen Sie diesen Schritt um so viele Konfigurationen für diesen Alias (globale Ressource) hinzuzufügen, wie erforderlich.
2. Wählen Sie im Fenster "Konfigurationen" eine der Konfigurationen aus und suchen Sie den Ordner, den Sie als globale Ressource erstellen möchten.
3. Definieren Sie eine Ordnerressource für jede Konfiguration (d.h. wiederholen Sie Schritt 3 für die verschiedenen von Ihnen erstellten Konfigurationen).
4. 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 **Globale Ressourcen verwalten** unter "Ordner" aufgelistet.

12.1.3 Datenbanken

Im Dialogfeld Globale Ressource für Datenbanken (*Abbildung unten*) können Sie auf folgende Weise eine Datenbankressource hinzufügen:



Global Resource

Alias name
Resource alias: MyDatabase

Configurations

- +   
- Default
- AlternativeDB

Settings for configuration "AlternativeDB"

Click to Choose Database: 

Database:

Data Source	C:\Documents and Settings\al...
User ID	
Password	
Provider	Microsoft.Jet.OLEDB.4.0
Persist Security Info	false
Jet OLEDB:Sys...	

MapForce execution parameters

Catalog	Data_shape
sUserID	
Password	
JDBCDriver	sun.jdbc.odbc.JdbcOdbcDriver
JDBCDatabaseURL	jdbc:odbc:;DRIVER=Microsoft A
Provider	Microsoft.Jet.OLEDB.4.0

OK Cancel

So definieren Sie eine globale Ressource vom Typ Datenbank:

1. Geben Sie einen Aliasnamen ein.
2. 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**  und geben Sie im angezeigten Dialogfeld **Konfiguration hinzufügen** den Konfigurationsnamen ein. Klicken Sie auf **OK**. Die neue Konfiguration wird im Fenster "Konfigurationen" aufgelistet. Wiederholen Sie diesen Schritt um so viele Konfigurationen für diesen Alias (globale Ressource) hinzuzufügen, wie erforderlich.
3. Wählen Sie im Fenster "Konfigurationen" eine der Konfigurationen aus und klicken Sie auf die Schaltfläche **Datenbank wählen**. Daraufhin wird das Dialogfeld Verbindung zu Globaler Ressource erstellen angezeigt (*Abbildung unten*).
- 4.



Wählen Sie aus, ob Sie die Verbindung zur Datenbank mit dem Verbindungsassistenten herstellen, eine bestehende Verbindung verwenden möchten oder eine ADO-Verbindung oder eine ODBC-Verbindung herstellen möchten. Stellen Sie die Definition der Verbindungsmethode fertig, wie im Abschnitt [Herstellen einer Verbindung zu einer Datenbank](#) beschrieben. Sie können den [Verbindungsassistenten](#), [ADO-Verbindungen](#) oder [ODBC-Verbindungen](#) 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.

4. Bei Herstellung einer Verbindung zu einem Datenbankserver werden Sie aufgefordert, ein Datenbank-Root-Objekt auf dem Server auszuwählen. Das ausgewählte Root-Objekt wird dann bei Verwendung dieser Konfiguration als Root-Objekt geladen. Wenn Sie kein Root-Objekt auswählen, können Sie dies nachholen, wenn die globale Ressource geladen wird.

5. Geben Sie für jede Konfiguration eine Datenbankressource an (d.h. wiederholen Sie Schritt 3 und 4 für die verschiedenen von Ihnen erstellten Konfigurationen).
6. 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.

12.1.4 Kopieren von Konfigurationen

Im Dialogfeld **Globale Ressourcen verwalten** können Sie bestehende Konfigurationen für alle Konfigurationstypen duplizieren. Dazu wählen Sie eine Konfiguration und klicken auf das

Symbol **Konfiguration kopieren** . Dann wählen Sie einen Konfigurationsnamen aus oder geben einen neuen ein und klicken auf **OK**. Dadurch wird eine Kopie der ausgewählten Konfiguration erstellt, die Sie nun entsprechend bearbeiten können.

12.2 Verwendung globaler Ressourcen

Es gibt mehrere Arten von globalen Ressourcen (Datei, Ordner und Datenbank). In bestimmten Szenarios können Sie in XMLSpy bestimmte Arten von globalen Ressourcen verwenden. So können Sie z.B. für eine XML-Arbeitsdatei oder eine CSS-Datei globale Ressourcen vom Typ Datei oder Ordner verwenden. Oder Sie können mit Hilfe einer Ressource vom Typ Datenbank ein neues datenbankbasiertes SPS erstellen. Die verschiedenen Szenarios, in denen Sie globale Ressourcen in XMLSpy verwenden können, sind im Unterabschnitt dieses Abschnitts aufgelistet: [Dateien und Ordner](#) und [Datenbanken](#).

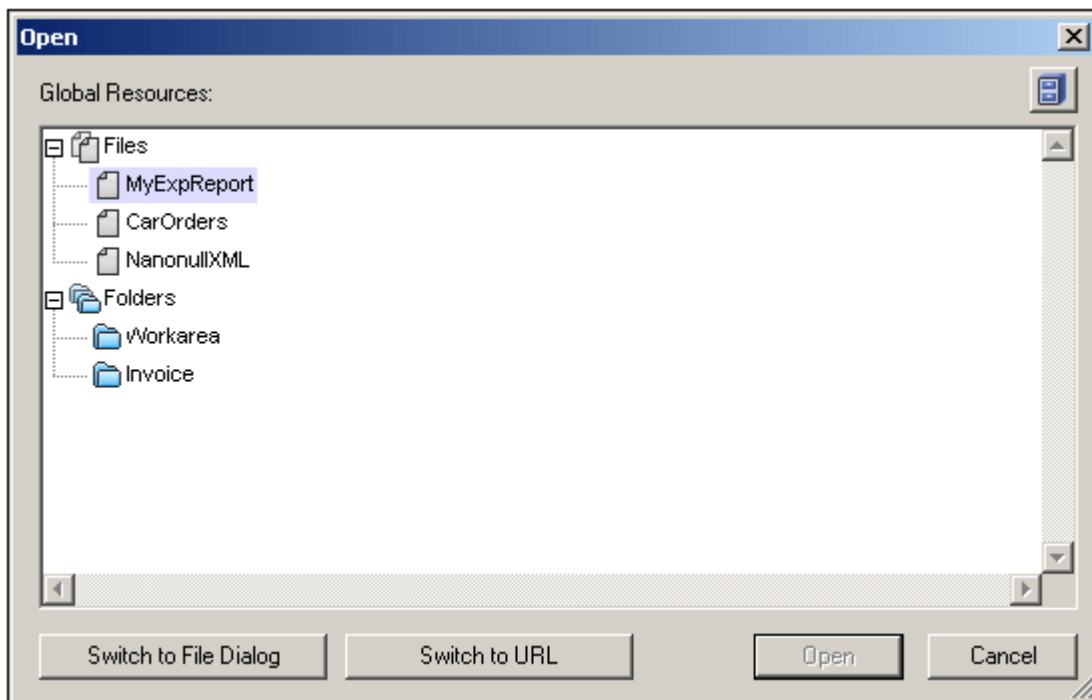
Auswahloptionen, die festlegen, welche Ressource verwendet wird

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" ausgewählt. Die globalen Ressource-Definitionen, die in der aktiven XML-Datei für globale Ressourcen enthalten sind, stehen 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** 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.

12.2.1 Zuweisen von Dateien und Ordnern

In diesem Abschnitt wird beschrieben, wie globale Ressourcen vom Typ Datei und Ordner zugewiesen werden. Die Zuweisung für diese beiden Arten erfolgt auf unterschiedliche Weise. In jedem der unten beschriebenen Szenarios erscheint bei Auswahl des Befehls **Zu globalen Ressourcen wechseln** das Dialogfeld "Globale Ressource öffnen" (*Abbildung unten*).



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.

Im Dialogfeld "Globale Ressource öffnen" können Sie durch Klicken auf die entsprechende Schaltfläche am unteren Rand des Dialogfelds zur Datei- bzw. URL-Dialogfeld wechseln. Mit der Schaltfläche **Globale Ressourcen verwalten**  im rechten oberen Eck erscheint das Dialogfeld [Globale Ressourcen verwalten](#) dialog.

Verwendungsszenarios

Globale Ressourcen vom Typ Datei und Ordner können in den folgenden Szenarios verwendet werden:

- [Öffnen globaler Ressourcen](#)
- [Speichern als globale Ressource](#)
- [Zuweisen von Dateien zu XSLT-Transformationen](#)
- [XSLT-Transformationen und XQuery-Ausführungen](#)
- [Zuweisen eines SPS](#)

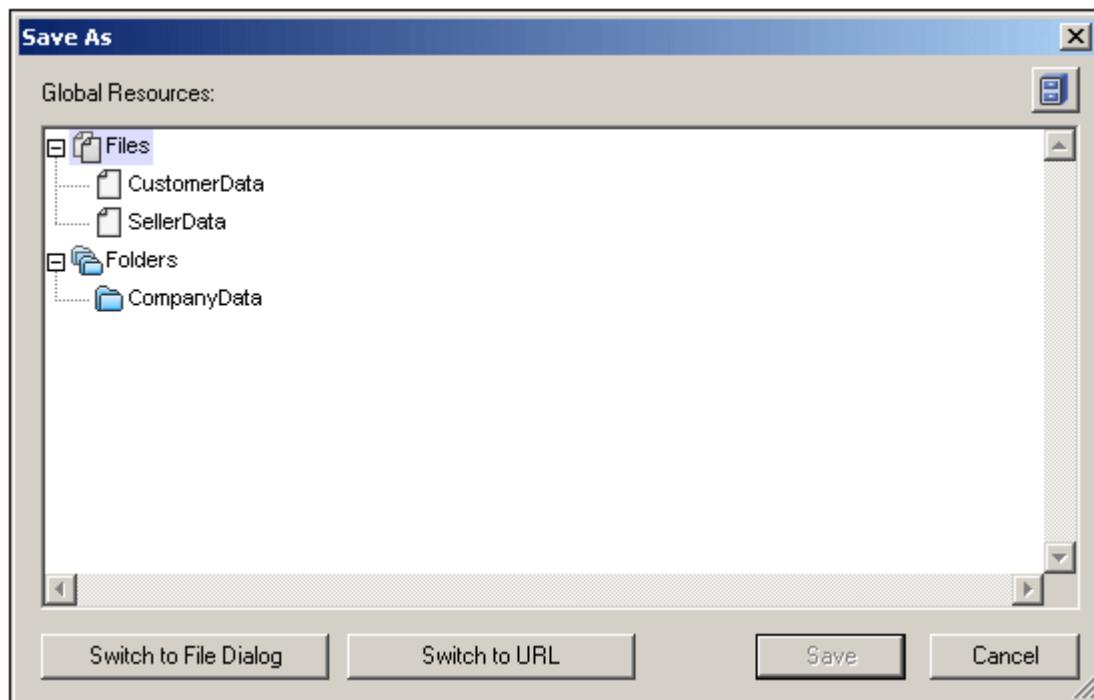
Öffnen von globalen Ressourcen

Eine globale Ressource kann in XMLSpy mit dem Befehl [Datei | Öffnen \(Zu globaler Ressource wechseln\)](#) 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 **Zu globaler Ressource wechseln**, 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](#) handelt, wird die Datei direkt gespeichert. Wenn es sich um einen [Ordneralias](#) 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](#) 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 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 für 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](#) und [XSL/XQuery | XSL:FO zuweisen](#)) und XML-Dateien ([XSL/XQuery | XML-Beispieldatei zuweisen](#)) 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

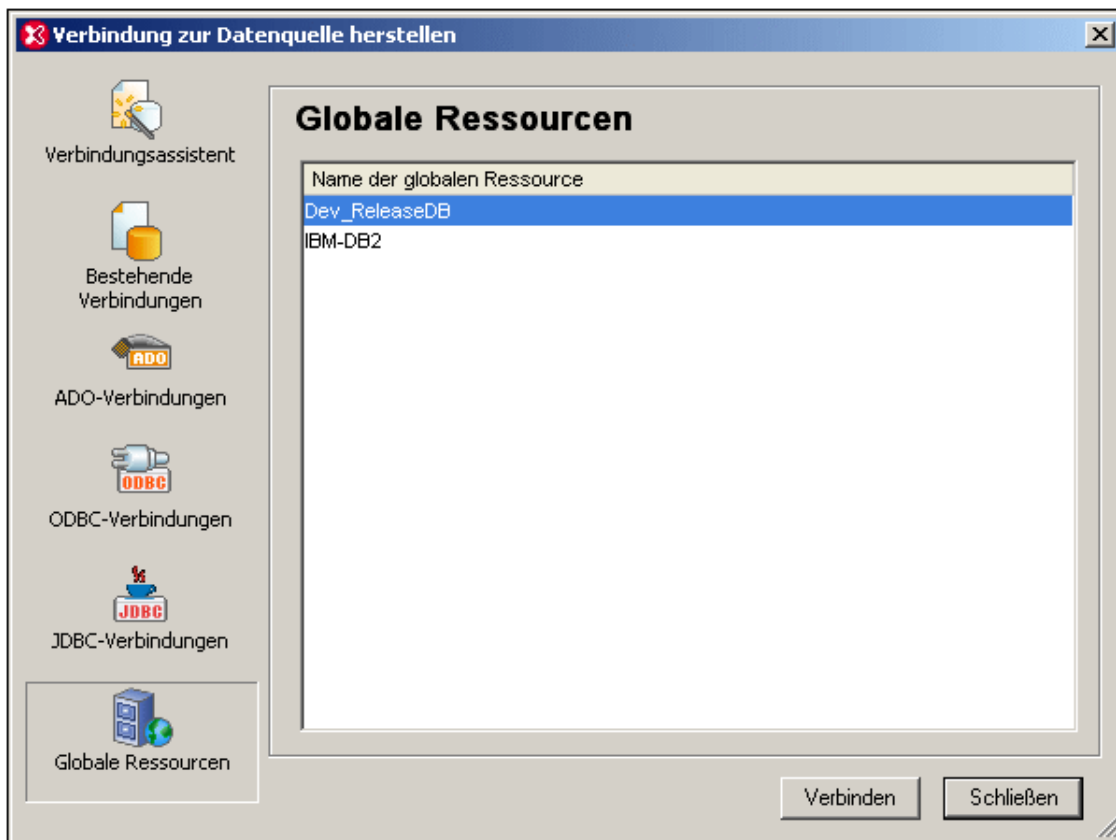
Bei Auswahl des Befehls [XSL/XQuery | XSL-Transformation](#), [XSL/XQuery | XSL:FO-Transformation](#) oder [XSL/XQuery | XQuery-Ausführung](#) wird ein Dialogfeld angezeigt, in dem Sie nach der gewünschten XSLT-, XQuery- oder XML-Datei suchen können. Klicken Sie auf die Schaltfläche **Durchsuchen** und anschließend auf **Zu globaler Ressource wechseln** um das Dialogfeld "Globale Ressource öffnen" aufzurufen ([Abbildung ganz oben im Abschnitt](#)). Für die Transformation wird die Datei verwendet, die mit der aktuellen 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 **Zu globaler Ressource wechseln**, um das Dialogfeld "Globale Ressource öffnen" aufzurufen ([Abbildung ganz oben im Abschnitt](#)). 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.

12.2.2 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.



Wenn Sie im Dialogfeld "Datenbank öffnen" auf die Schaltfläche "Globale Ressourcen" klicken, werden alle globalen Ressourcen vom Typ Datenbank, die in der derzeit aktiven [XML-Datei für globale Ressourcen](#) definiert wurden, angezeigt. Wählen Sie die gewünschte globale Ressource aus und klicken Sie auf **Verbindung herstellen**. 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](#) und [Importieren von Datenbankdaten](#)).

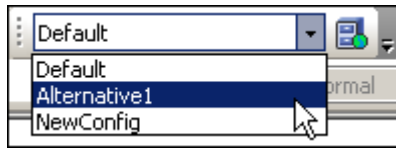
12.2.3 Wechseln von Konfigurationen

Es kann immer nur eine globale Ressourcenkonfiguration aktiv sein und diese ist applikationsweit aktiv. D.h. die aktive Konfiguration ist für alle Aliasse in den derzeit offenen Dateien 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 einem XML-Dokument mehrere XSLT-Transformationsdateien über eine globale Ressource mit mehreren Konfigurationen zugewiesen wurde. gewechselt werden, indem Sie einfach die Konfiguration der globalen Ressource ändern. Dies kann auf zwei Arten erfolgen:

- Wenn Sie die Maus über den Menübefehl **Extras | Aktive Konfiguration** platzieren, wird ein Untermenü mit einer Liste aller Konfigurationen in der XML-Datei für globale Ressourcen angezeigt. Wählen Sie die gewünschte Konfiguration aus.
- Wählen Sie in der Auswahlliste der Symbolleiste "Globale Ressourcen" (*Abbildung*

unten) die gewünschte Konfiguration aus. (Die Symbolleiste "globale Ressourcen" kann über den Menübefehl **Extras | Anpassen | Symbolleisten | Globale Ressourcen** ein- und ausgeblendet werden.)

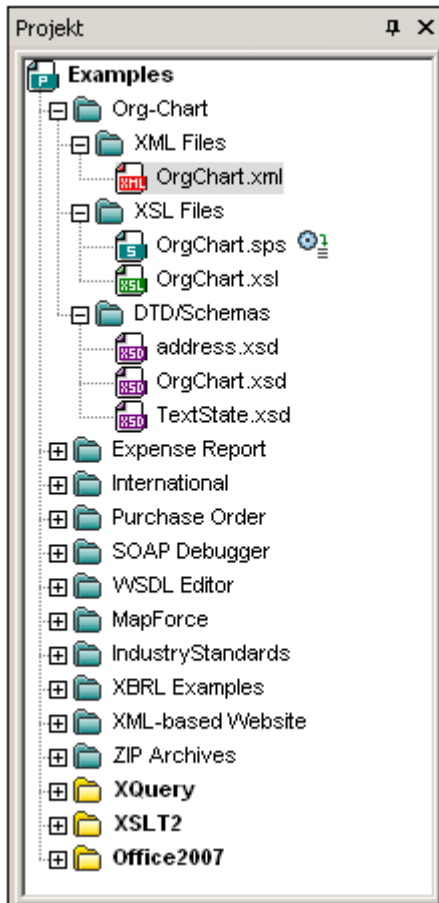


Die XSLT-Datei wird sofort gewechselt.

Auf diese Art können Sie durch Ändern der aktiven Konfiguration Quelldateien auswechseln, die über eine globale Ressource zugewiesen sind.

13 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.

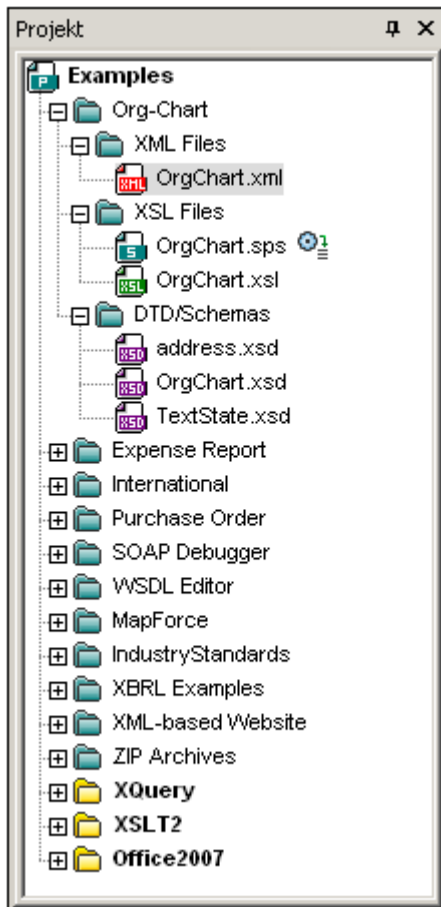


Sie können also in Projekten Dateien ablegen, die gemeinsam verwendet werden, um diese schneller aufrufen zu können. Außerdem können Sie Schemas und XSLT-Dateien für einzelne Ordner definieren und so die Batch-Verarbeitung von Dateien in einem Ordner ermöglichen.

In diesem Abschnitt wird beschrieben, [wie man Projekte erstellt und bearbeitet](#) und [wie man Projekte verwendet](#).

13.1 Erstellen und Bearbeiten von Projekten

Projekte werden über das [Projektfenster](#) (Abbildung unten) und das [Projektmenü](#) verwaltet. Es kann immer nur ein Projekt gleichzeitig in der Applikation geöffnet sein. Das offene Projekt wird im [Projektfenster](#) 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 Ordernamen; *siehe Abbildung unten*), werden Sie gefragt, ob Sie diese Änderungen speichern möchten.

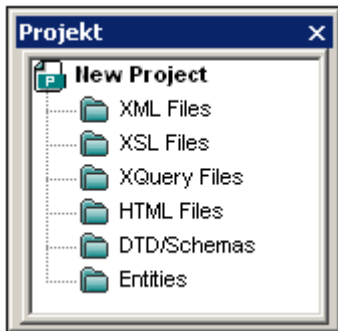
Benennen und Speichern von Projekten

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:

- *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](#).
- *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](#).

Projekteigenschaften

Die Eigenschaften eines Ordners werden im Dialogfeld "Eigenschaften" dieses Ordners gespeichert. Um dieses Dialogfeld aufzurufen, wählen Sie zuerst den Ordner aus und anschließend im Menü **Projekt** oder im Kontextmenü (das durch Rechtsklick auf den Ordner aufgerufen wird) den Befehl **Eigenschaften**. Beachten Sie, dass Eigenschaften nicht nur für den Projektordner der obersten Ebene definiert werden können, sondern auch für Ordner auf verschiedenen untergeordneten Ebenen der Projekthierarchie. Im Dialogfeld "Eigenschaften" können die folgenden Ordneigenschaften 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.
- *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.

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 [Benutzerreferenz](#) dieses Handbuchs.

Speichern von Projekten

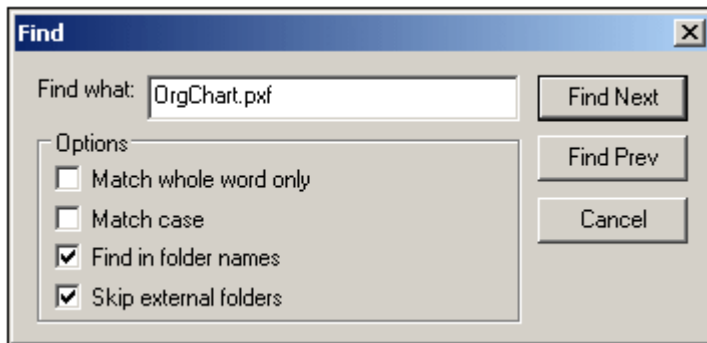
Alle an einem Projekt vorgenommenen Änderungen wie z.B. das Hinzufügen oder Löschen einer Datei oder die Änderung einer Projekteigenschaft müssen mit dem Befehl **Projekt speichern** gespeichert werden.

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*)



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. Ordernamen 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 Ordernamen übereinstimmen muss.
- Ordernamen können in die Suche einbezogen werden. Andernfalls werden nur Dateinamen durchsucht.
- [Externe Ordner](#) 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 innerhab 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.

13.2 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.

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.
- *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.

14 Datei-/Verzeichnisvergleiche

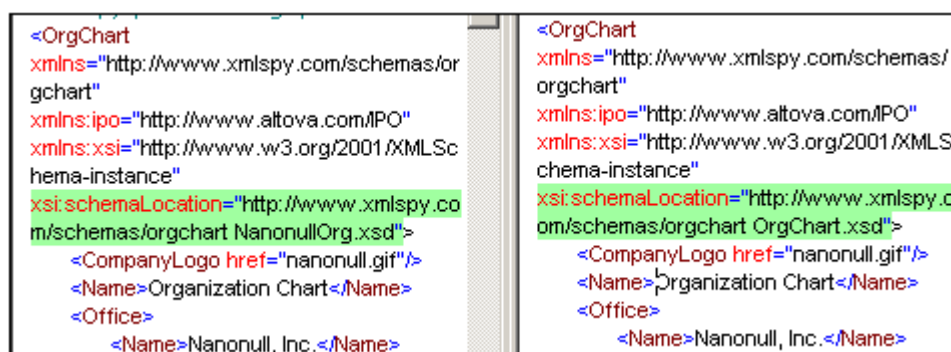
XMLSpy bietet eine Dateivergleichsfunktion sowie eine Verzeichnisvergleichsfunktion, die miteinander verknüpft sind. Datei- und Verzeichnisvergleiche werden mit dem Befehl **Geöffnete Datei vergleichen mit** bzw. **Verzeichnisse vergleichen** aus dem Menü **Extras** gestartet. Die Vergleichsoptionen 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 [Benutzerreferenz](#) näher beschrieben. In den Unterabschnitten dieses Abschnitts finden Sie eine Übersicht über die Funktionsweise von [Dateivergleichen](#) und [Verzeichnisvergleichen](#).

14.1 Dateivergleiche

Mit Hilfe der [Dateivergleichsfunktion](#) können Sie die aktive Datei mit einer anderen Datei, welche über ein "Datei öffnen" Dialogfeld oder eine [globale Ressource](#) ausgewählt wird, verglichen. Im Folgenden finden Sie eine Anleitung zur Vorgehensweise. Nähere Informationen dazu finden Sie im Abschnitt [Benutzerreferenz](#).

- Die aktuellen Einstellungen im Dialogfeld [Vergleichsoptionen](#) 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"](#) 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*).

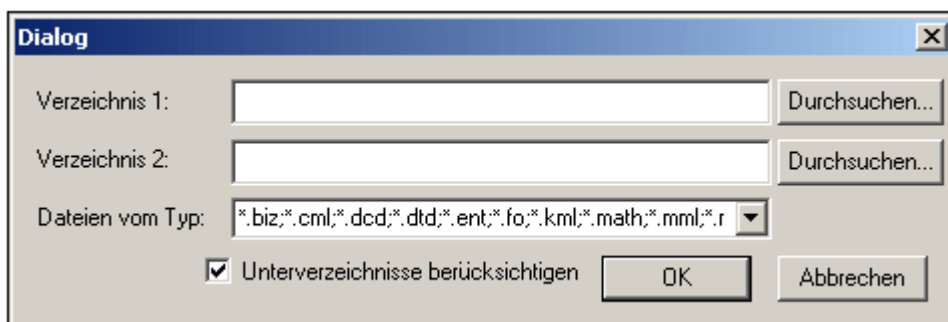


Außerdem wird ein Fenster mit Steuerelementen angezeigt, über das Sie durch die Unterschiede navigieren und diese zusammenführen können.

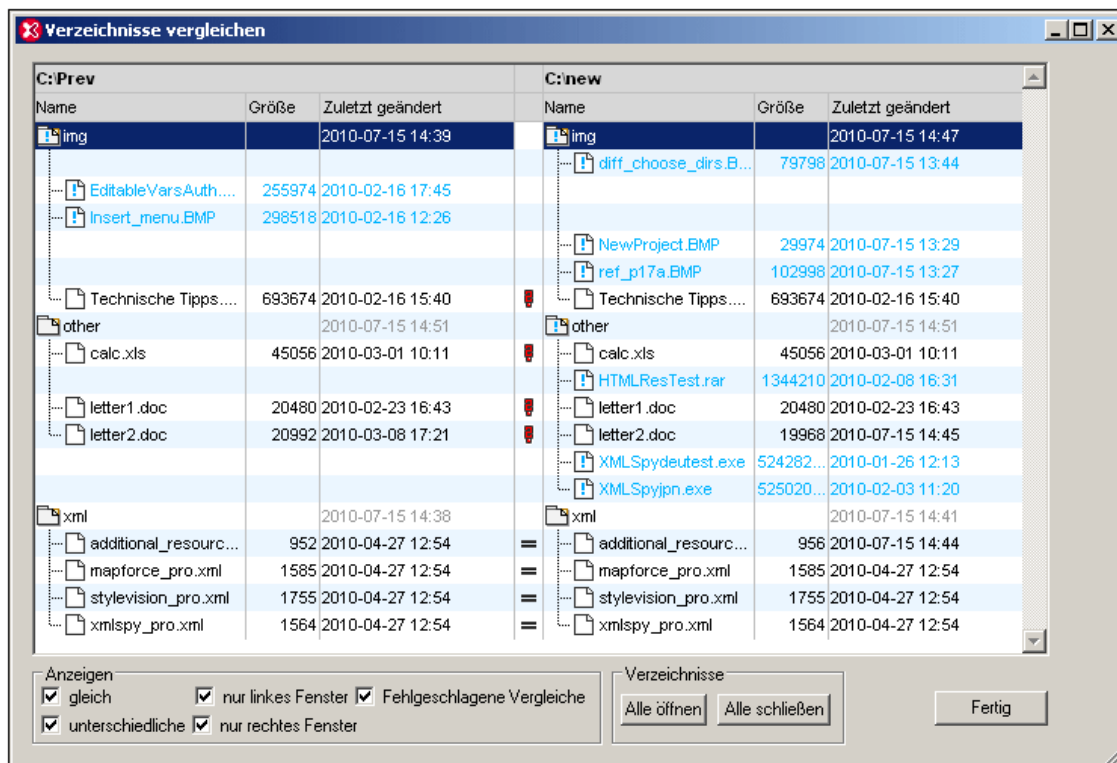
Das [Dialogfeld "Einstellungen"](#) enthält eine Reihe von Optionen, um zu definieren, welche Aspekte der XML-Dokumente beim Vergleich berücksichtigt und welche ignoriert werden sollen. Nähere Informationen dazu finden Sie in der [Benutzerreferenz](#) im Abschnitt [Vergleichsoptionen](#).

14.2 Verzeichnisvergleiche

Mit Hilfe der [Verzeichnisvergleichsfunktion](#) 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 Dateiarnten beim Verzeichnisvergleich berücksichtigt werden sollen.



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"](#). 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 [Benutzerreferenz](#) unter der Beschreibung des Befehls **Verzeichnisse vergleichen**. Sie können anschließend auf eine Dateizeile doppelklicken, um einen Dateivergleich sofort zu starten.

15 Versionskontrolle

Projektdateien können unter Versionskontrolle gestellt werden. Es wird eine ganze Reihe von Versionskontrollsystemen unterstützt und Altova hat die Funktionen mit den verschiedensten Treibern und Versionskontrollsystemen getestet; eine Liste davon finden Sie im Abschnitt [Unterstützte Versionskontrollsysteme](#).

Da XMLSpy die Microsoft Source Code Control Interface (MSSCCI) v1.1 – v1.3 implementiert hat, werden mehrere Versionskontrollsysteme wie u.a. Microsoft SourceSafe und andere kompatible Speichersysteme unterstützt.

Anmerkung: Die **64-Bit** Version von XMLSpy unterstützt automatisch alle der in dieser Dokumentation angeführten unterstützten 32-Bit-Versionskontrollprogramme. Bei Verwendung einer 64-Bit-Version von XMLSpy mit einem 32-Bit-Versionskontrollprogramm ist die Option "Statusaktualisierungen alle ...ms durchführen" ([Extras | Optionen](#)) automatisch **ausgegraut** und kann nicht ausgewählt werden.

Übersicht über die Funktion "Versionskontrolle"

Um Dateien in einem XMLSpy-Projekt 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](#) 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 Untermenü](#) 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](#)) oder (ii) im Dialogfeld "Versionskontrolle wechseln" ([Projekt | Versionskontrolle | Versionskontrolle wechseln](#)).

Anmerkung: Beachten Sie, dass ein Versionskontrollprojekt nicht dasselbe wie ein XMLSpy-Projekt ist. Versionskontrollprojekte sind verzeichnisabhängig, während XMLSpy-Projekte logische Konstruktionen ohne direkte Verzeichnisabhängigkeit sind.

Registry-Eintrag und Plug-ins

Microsoft hat einen Registry-Eintrag definiert, bei dem alle SCC-kompatiblen Programme sich selbst registrieren. Dies ist der Eintrag für XMLSpy.

```
HKEY_LOCAL_MACHINE\SOFTWARE\SourceCodeControlProvider\InstalledSCCProviders
```

Bitte beachten Sie außerdem, dass **Plugins** für Versionskontrollsysteme nicht von allen SSC-Produkten automatisch installiert werden. Nähere Informationen finden Sie in der

Dokumentation zu Ihrer Versionskontrollsoftware.

In diesem Abschnitt

Dieser Abschnitt ist folgendermaßen gegliedert:

- [Unterstützte Versionskontrollsysteme](#) enthält eine Liste der Versionskontrollsysteme (SCS), die von XMLSpy unterstützt werden. Die Systeme werden in alphabetischer Reihenfolge nach Server aufgelistet. Für jeden Server werden auch die unterstützten Clients aufgelistet.
- [Installieren von Versionskontrollsystemen](#) enthält Anmerkungen, wie die verschiedenen Versionskontrollsysteme installiert werden.
- [Versionskontrollsysteme SCSs und Vergleich mit Altova DiffDog](#) enthält eine Beschreibung, wie die verschiedenen Versionskontrollsysteme für das Vergleichen mit Altova DiffDog konfiguriert werden .

Menübefehle

Die Menübefehle für die Verwendung der Versionskontrollsysteme befinden sich im Untermenü [Projekt | Versionskontrolle](#) und werden im Abschnitt [Benutzerreferenz](#) beschrieben.

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.

15.1 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. In den Anmerkungen im Anschluss an die Liste finden Sie Informationen über den Grad der Unterstützung.

AccuRev

Version: AccuRev 4.7.0 Windows

Clients: • AccuBridge for Microsoft SCC 2008.2

Bazaar

Version: Bazaar 1.9 Windows

Clients: • Aigenta Unified SCC 1.0.6

Borland StarTeam 2008

Version: StarTeam 2008 Release 2

Clients: • Borland StarTeam Cross-Platform Client 2008 R2

Codice Software Plastic SCM

Version: Codice Software Plastic SCM Professional 2.7.127.10 (Server)

Clients: • Codice Software Plastic SCM Professional 2.7.127.10 (SCC Plugin)

Collabnet Subversion 1.5

Version: 1.5.4

Clients: • Aigenta Unified SCC 1.0.6
• PushOK SVN SCC 1.5.1.1
• <%AC-PUSHOKSVNX64%>
• TamTam SVN SCC 1.2.21

ComponentSoftware CS-RCS (PRO)

Version: ComponentSoftware CS-RCS (PRO) 5.1

Clients: • ComponentSoftware CS-RCS (PRO) 5.1

Dynamsoft SourceAnywhere for VSS

Version: Dynamsoft SourceAnywhere for VSS 5.3.2 Standard/Professional Server

Clients: • Dynamsoft SourceAnywhere for VSS 5.3.2 Client

Dynamsoft SourceAnywhere Hosted

Version: Server hosted in a Bell Data Center

Clients: • Dynamsoft SourceAnywhere Hosted Client (22252)

Dynamsoft SourceAnywhere Standalone

Version: SourceAnywhere Standalone 2.2 Server

Clients: • Dynamsoft SourceAnywhere Standalone 2.2 Client

IBM Rational ClearCase 7

Version: 7.0.1 (LT)

Clients: • IBM Rational ClearCase 7.0.1 (LT)

March-Hare CVSNT 2.5

Version: 2.5.03.2382

Clients: • Aigenta Unified SCC 1.0.6

March-Hare CVS Suite 2008

Version: Server 2008 [3321]

Clients:

- Jalindi Igloo 1.0.3
- March-Hare CVS Suite Client 2008 (3321)
- PushOK CVS SCC NT 2.1.2.5
- <%AC-PUSHOKCVSX64%>
- TamTam CVS SCC 1.2.36

Mercurial

Version: Mercurial 1.0.2 for Windows

Clients: • Sergey Antonovs HgSCC 1.0.1

Microsoft SourceSafe 2005

Version: 2005 with CTP

Clients:

- Microsoft SourceSafe 2005 with CTP

Microsoft Visual Studio Team System 2008 Team Foundation Server

Version: 2008

Clients:

- Microsoft Team Foundation Server 2008 MSSCCI Provider

Perforce 2008

Version: P4S 2008.1

Clients:

- Perforce P4V 2008.1

PureCM

Version: PureCM Server 2008/3a

Clients:

- PureCM Client 2008/3a

QSC Team Coherence Version Manager

Version: QSC Team Coherence Server 7.2.0.30

Clients:

- QSC Team Coherence Client 7.1.4.30

Qumasoft QVCS Enterprise

Version: QVCS Enterprise 2.1.18

Clients:

- Qumasoft QVCS Enterprise 2.1.18

Qumasoft QVCS Pro

Version: 3.10.18

Clients:

- Qumasoft QVCS Pro 3.10.18

Reliable Software Code Co-Op

Version: Code Co-Op 5.1a

Clients:

- Reliable Software Code Co-Op 5.1a

Seapine Surround SCM

Version: Surround SCM Client/Server for Windows 2009.0.0

Clients: • Seapine Surround SCM Client 2009.0.0

Serena Dimensions

Version: Dimensions Express/CM 10.1.3 for Win32 Server

Clients: • Serena Dimensions 10.1.3 for Win32 Client

Softimage Alienbrain

Version: Alienbrain Server 8.1.0.7300

Clients: • Softimage Alienbrain Essentials/Advanced Client 8.1.0.7300

SourceGear Fortress

Version: 1.1.4 Server

Clients: • SourceGear Fortress 1.1.4 Client

SourceGear SourceOffsite

Version: SourceOffsite Server 4.2.0

Clients: • SourceGear SourceOffsite Client 4.2.0 (Windows)

SourceGear Vault

Version: 4.1.4 Server

Clients: • SourceGear Vault 4.1.4 Client

VisualSVN Server 1.6

Version: 1.6.2

Clients: • Aigenta Unified SCC 1.0.6
• PushOK SVN SCC 1.5.1.1
• <%AC-PUSHOKSVNX64%>
• TamTam SVN SCC 1.2.21

Bitte beachten Sie:

- Altova hat Microsoft Source Code Control Interface (MSSCCI) v1.1 – v1.3 in XMLSpy implementiert und die Unterstützung der oben aufgelisteten Treiber und Versionskontrollsysteme getestet. Es wird erwartet, dass XMLSpy auch etwaige zukünftige Versionen dieser Produkte unterstützen wird.
- Versionskontrollsysteme, die in der obigen Tabelle nicht aufgelistet sind, die aber der MSSCCI 1.1-1.3 Spezifikation entsprechen sollten ebenfalls mit XMLSpy funktionieren.

15.2 Installation von Versionskontrollsystemen

Dieser Abschnitt enthält nähere Informationen zum Einrichten und Installieren der verschiedenen Versionskontrollsysteme.

AccuBridge for Microsoft SCC 2008.2

<http://www.accurev.com/>

1. Installieren Sie die AccuRev Client Software, führen Sie den Installer aus, definieren Sie den Server, zu dem eine Verbindung hergestellt werden soll, (Hostname und Port) und erstellen Sie anschließend einen Arbeitsbereich.
2. Installieren Sie den AccuBridge SCC Provider. Extrahieren Sie das ZIP-Archiv in das Verzeichnis <AccuRev installation dir>\bin.

Registrieren Sie AccuRev.dll und SccAcc.dll wie folgt:
3. Öffnen Sie ein Befehlseingabefenster (wenn Sie mit Vista arbeiten, starten Sie den Windows Explorer, gehen Sie zu C:\Windows\System32, klicken Sie auf die rechte Maustaste und führen Sie die cmd.exe "As administrator" aus).
4. Gehen Sie zum Verzeichnis <installation AccuRev dir>\bin.
5. Geben Sie an der Eingabeaufforderung den folgenden Befehl ein:
 - Regsvr32 AccuRev.dll
 - Regsvr32 SccAcc.dll
6. Führen Sie das Programm SwitchScc.exe aus und definieren Sie als den Provider AccuRev.
7. Führen Sie in Windows ein Logoff und anschließend ein Login durch.

Aigenta Unified SCC 1.0.6

<http://aigenta.com/products/UnifiedScc.aspx>

Voraussetzungen: Source Control Client. Aigenta Unified SCC funktioniert mit:

- Subversion Befehlszeilen-Client 1.5.4 unter <http://subversion.tigris.org>
- CVSNT 2.5 (Client) unter <http://www.cvsnt.org>
- Bazaar 1.9 Windows (und damit in Zusammenhang stehenden Komponenten) unter <http://bazaar-vcs.org/> (<http://bazaar-vcs.org/WindowsInstall>)

Eine Standard-Installation funktioniert mit allen Altova-Produkten ordnungsgemäß.

Borland StarTeam Cross-Platform Client 2008 R2

<http://www.borland.com/us/products/starteam>

Um die Borland StarTeam Microsoft SCC Integration zu installieren, führen Sie das Setup-Programm aus und wählen Sie bei der Installation die SCC API Integration.

Altova-Produkte können nun bei Angabe der Server-Adresse, des Endpunkts, des einzuloggenden Benutzernamens und Passworts und Ihres Projekt- und Arbeitspfads eine Verbindung zur Speicherschnittstelle herstellen.

Codice Software Plastic SCM Professional 2.7.127.10 (SCC Plugin)

<http://www.codicesoftware.com/xpproducts.aspx>

Eine Standard-Installation funktioniert mit allen Altova-Produkten ordnungsgemäß. Es müssen nur die Komponenten "client" und das "Visual Studio SCC plug-in" installiert werden.

ComponentSoftware CS-RCS (PRO) 5.1

<http://www.componentsoftware.com/Products/RCS>

1. Um ComponentSoftware CS-RCS (PRO) zu installieren, starten Sie das Setup-Programm und aktivieren Sie die Option "Workstation Setup".
2. Definieren Sie die Root für die Struktur der Speicherschnittstelle und starten Sie Ihren Rechner nach Abschluss der Installation neu.
3. Über "ComponentSoftware RCS Properties" können Sie ein Projekt auswählen oder erstellen und einen Arbeitsordner definieren.

Dynamsoft SourceAnywhere for VSS 5.3.2 Client

http://www.dynamsoft.com/Products/SAW_Overview.aspx

Eine Standard-Installation funktioniert mit allen Altova-Produkten ordnungsgemäß.

Für die Integration mit Altova Produkten muss das Plug-in für Adobe DreamWeaver CS3 nicht installiert werden. Stellen Sie nach der Installation eine Server-Verbindung her und definieren Sie einen Arbeitsordner.

Dynamsoft SourceAnywhere Hosted Client (22252)

<http://www.dynamsoft.com/Products/SourceAnywhere-Hosting-Version-Control-Source-Control.aspx>

Eine Standard-Installation funktioniert mit allen Altova-Produkten ordnungsgemäß.

Für die Integration mit Altova Produkten muss das Plug-in für Adobe DreamWeaver CS3 nicht installiert werden.

Dynamsoft SourceAnywhere Standalone 2.2 Client

<http://www.dynamsoft.com/Products/SourceAnywhere-SourceSafe-VSS.aspx>

Eine Standard-Installation funktioniert mit allen Altova-Produkten ordnungsgemäß.

Für die Integration mit Altova Produkten muss das Plug-in für Adobe DreamWeaver CS3 nicht installiert werden. Stellen Sie nach der Installation eine Server-Verbindung her und definieren Sie einen Arbeitsordner.

IBM Rational ClearCase 7.0.1 (LT)

<http://www-01.ibm.com/software/awdtools/clearcase/>

Um IBM Rational ClearCase LT zu installieren, führen Sie das Setup-Programm aus.

- Sie werden aufgefordert, die Version des InstallShield Scripting-Prozessors zu aktualisieren, wenn diese älter als Version 10.5 ist. Wählen Sie "Update it if necessary".
Die Aktualisierung wird vor Beginn der Installation durchgeführt.
- Wählen Sie die Standardoption "Enterprise deployment, create a network release area and customize it using Siteprep".

Um die Versionskontrolle mit Altova-Produkten zu integrieren, muss nur der Client installiert werden. Aktivieren Sie nur das Kontrollkästchen "Client".

- Geben Sie einen Server-Namen sowie das/die Lizenzserver Element(e) an. Gehen Sie vor wie in den Beispielen, die vom Installer geliefert werden ([port@server_name](#)).
- Geben Sie einen Namen für die Konfiguration an, indem Sie den Namen entsprechend bearbeiten und den Pfad zu einem Release-Bereich (Release Area) eingeben. Dieser Pfad muss einen freigegebenen Ordner definieren.
- Sie können einen neuen Ordner auf Ihrem Rechner anlegen, ihn freigeben und als

Release-Bereich (Release Area) verwenden.

In Vista müssen Sie "Network discovery" im Network and Sharing Center auf "on" setzen, um diesen Pfad zu definieren.

Der Release-Bereich wird nun erstellt, einige Dateien werden hinein kopiert und ein Shortcut mit dem Namen **sitedefs.lnk** wird erstellt.

- Wenn alle Dateien kopiert wurden, fahren Sie fort, indem Sie vom Windows Explorer aus auf den Shortcut klicken.
Ein neues Setup-Programm beginnt mit der Installation des Client.
- Wenn das Setup-Programm beginnt, wählen Sie die Option "Install IBM Rational ClearCase LT".
- Klicken Sie auf "Weiter", akzeptieren Sie das "Software License Agreement" und starten Sie die Installation.

In **Vista** könnte das zweite Setup-Programm den internen Fehler: 2739 ausgeben. Starten Sie in diesem Fall den Windows Explorer und gehen Sie zu C:\Windows\System32.

- Klicken Sie auf die rechte Maustaste und führen Sie "cmd.exe" "As Administrator" aus. Daraufhin erscheint ein Eingabefenster.
- Geben Sie "regsvr32 jscript.dll" ein.
- Starten Sie das Setup-Programm wieder.

Um mit in ClearCase gespeicherten Dateien zu arbeiten, sollten Sie eine Ansicht erstellen, die auf Ihr ClearCase-Projekt verweist.

Jalindi Igloo 1.0.3

<http://www.jalindi.com/igloo/>

Um Jalindi Igloo mit Altova-Produkten zu verwenden, muss nur das Setup-Programm ausgeführt werden, um Jalindi Igloo zu installieren.

Bitte beachten Sie:

Wenn Sie Jalindi Igloo deinstallieren, werden alle anderen installierten SCC Provider Windows Registrierschlüssel (falls vorhanden) ebenfalls gelöscht und stehen nicht mehr zur Verfügung.

Wenn Sie mit Altova-Produkten arbeiten, wird empfohlen den Modus "Auto Commit" einzustellen.

- Der Modus "Auto Commit" befindet sich in den erweiterten Source Control-Optionen.
- Nachdem Sie einen Arbeitsbereich definiert haben, können Sie mit der Arbeit beginnen.

March-Hare CVS Suite Client 2008 (3321)

<http://www.march-hare.com/cvsnt/en.asp>

Eine "typical"-Installation funktioniert mit Altova-Produkten ordnungsgemäß.

Mercurial

siehe [Sergey Antonovs HgScc 1.0.1](#)

Microsoft SourceSafe 2005 with CTP

<http://msdn.microsoft.com/en-us/vstudio/aa718670.aspx>

Eine Standard-Installation von Microsoft Source Safe 2005 funktioniert mit Altova-Produkten ordnungsgemäß.

Microsoft Team Foundation Server 2008 MSSCCI Provider

<http://www.microsoft.com/downloads>

Voraussetzungen: Visual Studio 2008 Team Explorer oder Visual Studio 2008 mit Team Explorer 2008. Eine Standard-Installation funktioniert mit allen Altova-Produkten ordnungsgemäß.

Perforce P4V 2008.1

<http://www.perforce.com/>

Beim Perforce Visual Client (P4V) haben Sie die Wahl:

- alle Client-Funktionen zu installieren (Standardverhalten)
- nur die Funktion "SCC Plug-in (P4SCC)" zu installieren.

Eine Standard-Installation funktioniert mit allen Altova-Produkten ordnungsgemäß.

Wenn die Funktion "SCC Plug-in (P4SCC)" ausgewählt wird:

- funktionieren zwei SCC-Funktionen "Show differences" und "Source control manager" nicht.
- Die Funktion "Show differences" und die Möglichkeit, den Source Control Manager zu starten, funktionieren nicht, weil sie auf den nicht installierten Funktionen "Visual Merge Tool" bzw. "Visual Client (P4V)" basieren.
- Für die Vergleichsfunktion wird Drittanbietersoftware benötigt, während der Source Control Manager nur gestartet werden kann, nachdem der "Visual Client (P4V)" explizit installiert wurde.
- Nachdem Sie die Perforce Visual Client Installation gestartet haben, definieren Sie Ihre eigenen Client-Konfigurationseinstellungen (Server Name, Textbearbeitungsapplikation, Benutzername).
- Vergessen Sie nach Abschluss der Installation nicht, einen neuen Arbeitsbereich zu erstellen oder einen vorhandenen auszuwählen.

PureCM Client 2008/3a

<http://www.purecm.com/>

Eine Standard-Installation funktioniert mit allen Altova-Produkten ordnungsgemäß. Starten Sie nach der Installation den PureCM Client, um einen Server zu registrieren.

PushOK CVS SCC NT 2.1.2.5

http://www.pushok.com/soft_cvs.php

Eine Standard-Installation genügt, um PushOK CVS SCC NT zu verwenden.

- Stellen Sie nach Abschluss der Installation sicher, dass Ihre Version des CVS Proxy Plug-in korrekt registriert ist.
- Nach Definition eines Arbeitsbereichs können Sie anfangen, damit zu arbeiten.

<%AC-PUSHOKCVSX64%>

http://www.pushok.com/soft_cvs.php

Eine Standard-Installation genügt, um PushOK CVS SCC zu verwenden.

- Stellen Sie nach Abschluss der Installation sicher, dass Ihre Version des CVS Proxy Plug-in korrekt registriert ist.
- Nach Definition eines Arbeitsbereichs können Sie anfangen, damit zu arbeiten.

PushOK SVN SCC 1.5.1.1

http://www.pushok.com/soft_svn.php

Eine Standard-Installation genügt, um PushOK CVS SCC mit Altova-Produkten zu verwenden. Bei Installation unter Vista kann es vorkommen, dass die COM-Bibliothek **svncom.dll** nicht registriert werden kann. Stellen Sie in diesem Fall die Installation fertig und registrieren Sie die Bibliothek folgendermaßen manuell:

1. Starten Sie ein Eingabefenster über die Option "Run as administrator".
2. Geben Sie: `cd "C:\Program Files\PushOK Software\SVNSCC\svn"` ein
3. Geben Sie den Befehl `> regsvr32 svncom.dll` ein.

<%AC-PUSHOKSVNX64%>

http://www.pushok.com/soft_svn.php

Eine Standard-Installation genügt, um PushOK SVN SCC mit Altova-Produkten zu verwenden. Bei Installation unter Vista kann es vorkommen, dass die COM-Bibliothek **svncom.dll** nicht registriert werden kann. Stellen Sie in diesem Fall die Installation fertig und registrieren Sie die Bibliothek folgendermaßen manuell:

1. Starten Sie ein Eingabefenster über die Option "Run as administrator".
2. Geben Sie: `cd "C:\Program Files\PushOK Software\SVNSCC\svn"` ein
3. Geben Sie den Befehl `> regsvr32 svncom.dll` ein.

QSC Team Coherence Client 7.1.4.30

<http://www.teamcoherence.com>

Eine Standard-Installation funktioniert mit allen Altova-Produkten ordnungsgemäß.

- Wenn der Server auf dem Client-Rechner installiert ist, wird nach der Client-Installation einer Standardverbindung erstellt.
- Wenn der Server auf einem anderen Rechner installiert ist, müssen Sie im Dialogfeld "Connection Properties" des Team Coherence Client die Eigenschaft "HOSTNAME" in den Pfad zum entsprechenden Rechner ändern.

Qumasoft QVCS Enterprise 2.1.18

<http://www.qumasoft.com/>

Voraussetzungen: J2SE 1.5 oder höher <http://www.oracle.com/technetwork/java/index.html>

Um den Qumasoft QVCS-Enterprise Client zu installieren, starten Sie den Installer. Wenn Ihr Betriebssystem **Vista** ist, müssen Sie das Installationsverzeichnis vom Standardwert "C:\Program Files\QVCS-Enterprise Client" in "C:\QVCS-Enterprise Client" ändern. Der Grund dafür ist, dass Vista Applikationen nicht in den Bereich C:\Program Files schreiben lässt.

Bearbeiten Sie die Datei "**setEnv.cmd**" im Installationsverzeichnis, sodass die Umgebungsvariable `JAVA_HOME` auf den Ordner Ihrer JVM verweist.

Eventuell kann es in **Vista** beim Speichern der Datei zu einem Problem kommen.

2. Wenn das der Fall ist, starten Sie den Windows Explorer und gehen Sie zu `C:\Windows\System32`.
3. Klicken Sie auf die rechte Maustaste und führen Sie "`cmd.exe`" "As Administrator" aus. Daraufhin erscheint ein Eingabefenster.
4. Geben Sie "`cd <Installationsordner des QVCS –Enterprise Client>`" ein.
5. Geben Sie "`Notepad setEnv.cmd`" ein und bearbeiten Sie anschließend die Datei und speichern Sie sie.
6. Führen Sie vom Installationsverzeichnis des Qumasoft QVCS-Enterprise die Batch-Datei "`gui.bat`" aus.
7. Fügen Sie einen Server aus dem "Server-Menü" hinzu. Geben Sie den erforderlichen Namen, die IP-Adresse und Ports und den Log-in-Namen ein und definieren Sie einen lokalen Arbeitsbereich.

Qumasoft QVCS Pro 3.10.18

<http://www.qumasoft.com/>

Um Qumasoft QVCS-Pro zu installieren, führen Sie den Installer aus.

- Wenn Ihr Betriebssystem **Vista** ist, müssen Sie das Installationsverzeichnis vom Standardwert "C:\Program Files\QVCSBin" in "C:\QVCSBin" ändern.

Der Grund dafür ist, dass Vista Applikationen nicht in den Bereich C:\Program Files schreiben lässt.

- Starten Sie nach Fertigstellung der Installation den QVCS 3.10 Client, erstellen Sie einen neuen Benutzer und aktivieren Sie die **Ide Integration** durch Auswahl des Untermenüs "Ide Integration" im Menü "Admin" und Hinzufügen von QVCS als Versionskontroll-Tool.
- Erstellen Sie ein Projekt und definieren Sie einen Arbeitsbereich.

Reliable Software Code Co-Op 5.1a

http://www.relisoft.com/co_op/index.htm

Seapine Surround SCM Client 2009.0.0

<http://www.seapine.com/surroundscm.html>

Eine Standard-Installation funktioniert mit allen Altova-Produkten ordnungsgemäß.

- Nach der Installation muss eine Verbindung zu einem Server hergestellt werden.

Serena Dimensions 10.1.3 for Win32 Client

<http://www.serena.com/Products/dimensions/>

- Führen Sie eine "Typical" Installation des Serena Dimension Client durch.
- Definieren Sie den WEB Client Hostnamen und die Port-Nummer, wenn Sie dazu aufgefordert werden.

Sergey Antonovs HgSCC 1.0.1 for Mercurial SCS

http://www.newsupaplex.pp.ru/hgscs_news_eng.html

Eine Standard-Installation funktioniert mit allen Altova-Produkten ordnungsgemäß

Softimage Alienbrain Essentials/Advanced Client 8.1.0.7300

<http://www.alienbrain.com/>

- Führen Sie eine "Typical" Installation der Alienbrain Client Software aus und installieren Sie anschließend die Alienbrain Microsoft Visual Studio Integration.

Um mit Altova-Produkten zu arbeiten, müssen Sie Microsoft Visual Studio nicht installieren.

- Beim ersten Öffnen eines Projekts von der Versionskontrolle aus oder beim Hinzufügen eines Projekts zur Versionskontrolle, werden Sie aufgefordert einige Benutzereinstellungen zu definieren, z.B. Ihren Server und die Projektdatenbank, zu der eine Verbindung hergestellt werden soll.

SourceGear Fortress 1.1.4 Client

<http://www.sourcegear.com/fortress>

Eine Standard-Installation von SourceGear Fortress Client funktioniert mit Altova-Produkten.

SourceGear SourceOffsite Client 4.2.0 (Windows)

<http://www.sourcegear.com/sos/>

Eine Standard-Installation von SourceOffsite Client funktioniert mit Altova-Produkten.

SourceGear Vault 4.1.4 Client

<http://www.sourcegear.com/vault>

Eine Standard-Installation von SourceGear Vault Client funktioniert mit Altova-Produkten.

TamTam CVS SCC 1.2.36

<http://www.daveswebsite.com/software/tamtam>

Voraussetzungen: CVSNT 2.5 (client) unter <http://www.cvsnt.org>

Eine Standard-Installation funktioniert mit allen Altova-Produkten ordnungsgemäß.

- Um eine Verbindung zur CVS-Speicherschnittstelle herzustellen, müssen Sie CVSNT installieren.
- Öffnen Sie im Altova-Produkt die erweiterten Optionen der "Versionskontrolle" und geben Sie den Pfad zur Datei **cvns.exe** ein.

TamTam SVN SCC 1.2.21

<http://www.daveswebsite.com/software/tamtamsvn>

Voraussetzungen: Subversion Command Line Client 1.5.4 unter <http://subversion.tigris.org>

Eine Standard-Installation funktioniert mit allen Altova-Produkten ordnungsgemäß.

- Um eine Verbindung zur SVN-Speicherschnittstelle herzustellen, müssen Sie den Subversion Command Line Client installieren und in den Altova-Versionskontrolloptionen den Pfad zur ausführbaren Datei **svn.exe** definieren.
- Nach Start von XMLSpy müssen Sie den SCC Provider registrieren.

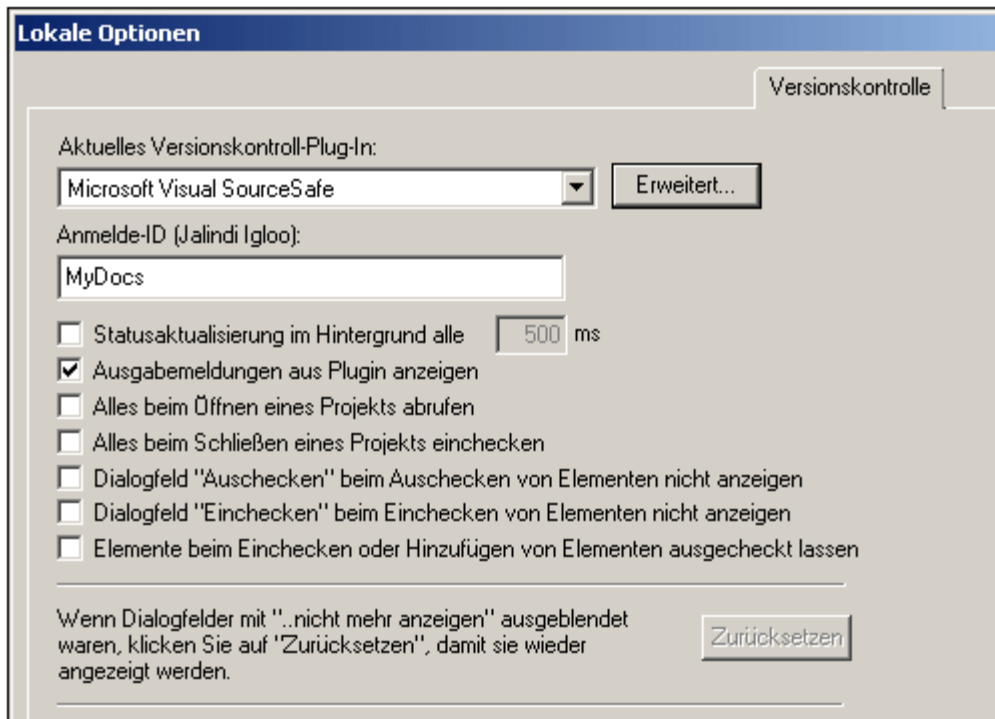
Auf einem **Vista** Rechner kann die CSS-Registrierung fehlschlagen.

- Falls dies der Fall ist, suchen Sie im Windows Explorer das Verzeichnis, das die ausführbare Altova-Applikationsdatei enthält.
- Klicken Sie auf die rechte Maustaste und führen Sie die Datei "As Administrator" aus. Die SCC-Registrierung sollte jetzt klappen.

15.3 Versionskontrolle und Vergleich mit Altova DiffDog

Sie können bestimmte Versionskontrollsysteme für die Verwendung von Altova DiffDog als Versionskontroll-Vergleichstool konfigurieren. Eine Liste davon sowie eine Anleitung zum Konfigurieren der einzelnen Systeme finden Sie weiter unten. In XMLSpy können Sie dies über das Dialogfeld "Optionen" (**Extras | Optionen**, *Abbildung unten*) auf dem Register "Versionskontrolle" tun.

Bei Verwendung einer 64-Bit-Version von XMLSpy mit einem 32-Bit-Versionskontrollprogramm ist die Option "Statusaktualisierungen alle ...ms durchführen" ([Extras | Optionen](#)) automatisch **ausgegraut** und kann nicht ausgewählt werden.



Das Kontrollkästchen "Statusaktualisierung im Hintergrund alle xxx ms" ist standardmäßig deaktiviert, d.h. es werden keine Statusaktualisierungen durchgeführt. Aktivieren Sie das Kontrollkästchen und geben Sie in das Feld einen Wert ein, wenn alle xx ms eine Statusaktualisierung durchgeführt werden soll. Bei 64-Bit-Versionen, die mit 32-Bit-Versionskontroll-Plugins arbeiten, hat diese Option keine Auswirkung.

Wählen Sie das gewünschte Versionskontrollsystem auf dem Register "Versionskontrolle" aus und klicken Sie auf die Schaltfläche "Erweitert". Das Dialogfeld, das sich daraufhin öffnet, ist je nach Versionskontrollsystem unterschiedlich. Beachten Sie bei der Konfiguration folgende Punkte:

- Wenn Sie eine Standardinstallation von Altova DiffDog durchgeführt haben, so lautet der Dateipfad zur ausführbaren Altova DiffDog Datei:
`c:\program files\altova\diffdog2012\DiffDog.exe`
Wenn Altova DiffDog in einem anderen Ordner auf Ihrem System installiert ist, fügen Sie den entsprechenden Wert ein, wo dies erforderlich ist.

Agenta Unified SCC 1.0.6

<http://agenta.com/products/UnifiedScc.aspx>

Mit den folgenden Schritten wird Altova DiffDog in Aigenta Unified SCC integriert:

1. Klicken Sie auf dem Register "Versionskontrolle" auf die Schaltfläche "Erweitert".
2. Wählen Sie das Register "Comparison and merging" und definieren Sie den vollständigen DiffDog-Dateipfad als Vergleichstool.

Borland StarTeam Cross-Platform Client 2008 R2

<http://www.borland.com/us/products/starteam>

Mit den folgenden Schritten wird Altova DiffDog in Borland Start Team integriert:

1. Verwenden Sie die "Personal Options" des StarTeam Client (Tools - Personal options – File - Alternate applications)
2. Compare utility: Geben Sie den vollständigen Pfad zu DiffDog ein
3. Compare utility options: \$file1 \$file2

ComponentSoftware CS-RCS (PRO) 5.1

<http://www.componentsoftware.com/Products/RCS>

Mit den folgenden Schritten wird Altova DiffDog in ComponentSoftware CS-RCS (Pro) integriert:

1. Gehen Sie zu den ComponentSoftware CS-RCS Properties.
2. Wählen Sie auf dem Register "File Types" eine Dateierweiterung aus und bearbeiten Sie sie.
3. Geben Sie den Wert "Custom Tool" für das "Difference Analysis Tool" ein bzw. wählen Sie ihn aus und durchsuchen Sie die Struktur, um den vollständigen Pfad zu DiffDog einzufügen.

Dynamsoft SourceAnywhere for VSS 5.3.2 Client

http://www.dynamsoft.com/Products/SAW_Overview.aspx

Mit den folgenden Schritten wird Altova DiffDog in Dynamsoft SourceAnywhere for VSS integriert:

1. Gehen Sie zu den Dynamic SourceAnywhere For VSS Client-Optionen.
2. Definieren Sie mit den Argumenten "%FIRST_FILE%" "%SECOND_FILE%" den vollständigen DiffDog Pfad als externe Applikation für diff/merge.

Warnung:

Führen Sie diese Einstellungen nicht über die Altova-Produktoptionen durch, da Sie hier keine externen Applikationsparameter einsetzen können.

Dynamsoft SourceAnywhere Hosted Client (22252)

<http://www.dynamsoft.com/Products/SourceAnywhere-Hosting-Version-Control-Source-Control.aspx>

Dynamsoft SourceAnywhere Standalone 2.2 Client

<http://www.dynamsoft.com/Products/SourceAnywhere-SourceSafe-VSS.aspx>

Mit den folgenden Schritten wird Altova DiffDog in Dynamsoft SourceAnywhere Hosted und Dynamsoft SourceAnywhere Standalone integriert:

1. Klicken Sie auf dem Register "Versionskontrolle" auf die Schaltfläche "Erweitert...".
2. Definieren Sie mit den Argumenten "%FIRST_FILE%" "%SECOND_FILE%" den vollständigen DiffDog-Pfad als externe Programmapplikation für diff/merge.
- 3.

Jalindi Igloo 1.0.3

<http://www.jalindi.com/igloo/>

Mit den folgenden Schritten wird Altova DiffDog in Jalindi Igloo integriert:

1. Wählen Sie in XMLSpy den Befehl "Unterschiede anzeigen".
2. Öffnen Sie den Bereich "Show differences or Merge Files".
3. Definieren Sie den "External Diff Command" durch Eingabe des vollständigen DiffDog-Pfads als externe Diff EXE.

Warnung: Bei Verwendung des Standard-Diff-Editors CvsConflictEditor kann es zu Problemen beim Vergleichen von Dateien mit besonders langen Zeilen kommen. Wir empfehlen Ihnen, an allen Dateien (v.a. ump-Dateien) ein "pretty print" durchzuführen, bevor Sie diese in der Speicherschnittstelle speichern. Dadurch wird die Zeilenlänge limitiert, sodass es zu keinen Problemen mit dem CVSConflictEditor kommt.

March-Hare CVS Suite Client 2008 (3321)

<http://www.march-hare.com/cvsnt/en.asp>

Mit den folgenden Schritten wird Altova DiffDog in Marc-Hare CVS Suite 2008 integriert:

1. Gehen Sie zu den TortoiseCVS Preferences und wählen Sie das Register "Tools".
2. Definieren Sie den vollständigen DiffDog-Pfad als Diff-Applikation und die Parameter "%1" "%2" als Parameter für den Vergleich in beide Richtungen.

Mercurial

siehe [Sergey Antonovs HgScc 1.0.1](#)

Microsoft SourceSafe 2005 with CTP

<http://msdn.microsoft.com/en-us/vstudio/aa718670.aspx>

Mit den folgenden Schritten wird Altova DiffDog in Microsoft SourceSafe 2005 integriert:

1. Klicken Sie auf dem Register "Versionskontrolle" auf die Schaltfläche "Erweitert".
2. Klicken Sie auf das Register "Custom Editors" und geben Sie die folgende Befehlszeile in das Befehlszeilenfeld ein: "C:\Program Files\Altova\DiffDog2012\DiffDog.exe %1 %2"
3. Wählen Sie über die Auswahlliste "Operation" den Befehl "File Difference".

Microsoft Team Foundation Server 2008 MSSCCI Provider

<http://www.microsoft.com/downloads>

Voraussetzungen: Visual Studio 2008 Team Explorer oder Visual Studio 2008 mit Team Explorer 2008. Mit den folgenden Schritten wird Altova DiffDog in Microsoft Visual Studio Team System 2008 Team Foundation Server MSSCCI Provider integriert:

1. Konfigurieren Sie in den Manager-Optionen (Visual Studio 2008 Team Explorer oder Visual Studio 2008) Altova DiffDog als neues Benutzertool
2. Wählen Sie als Plug-in Visual Studio Team Foundation Server Source.
3. Konfigurieren Sie durch Angabe der Erweiterungen die Dateien, die Sie mit DiffDog vergleichen möchten, sowie den vollständigen Dateipfad zu DiffDog.

Perforce P4V 2008.1

<http://www.perforce.com/>

Mit den folgenden Schritten wird Altova DiffDog in Perforce 2008 integriert:

1. Klicken Sie auf dem Register "Versionskontrolle" auf die Schaltfläche "Erweitert".
2. Wählen Sie im Bereich "Preferences" das Register "Diff".
3. Aktivieren Sie als Standardvergleichstool das Feld "Other application" und geben Sie den vollständige Pfad zu DiffDog ein.

**PushOK CVS SCC NT 2.1.2.5,
PushOK SVN SCC 1.5.1.1**

<%AC-PUSHOKCVSX64%>

<%AC-PUSHOKSVNX64%>

http://www.pushok.com/soft_cvs.php

Mit den folgenden Schritten wird Altova DiffDog in PushOK CVS NT und PushOK SVN SCC integriert:

1. Klicken Sie auf dem Register "Versionskontrolle" auf die Schaltfläche **"Erweitert..."**
2. Wählen Sie das Register "CVS Executables".
3. Wählen Sie im Feld "Diff/merge" den Wert "External merge/compare tool".
4. Fügen Sie den vollständigen DiffDog-Dateipfad ein.
5. Bearbeiten Sie im Feld "2 way diff cmd" den Wert "%first %second".

Warnung: Bei Verwendung des Standard-Vergleichseditors CvsConflictEditor kann es zu Problemen beim Vergleich von Dateien mit sehr langen Zeilen kommen. Wir empfehlen Ihnen, an allen Dateien (v.a. ump-Dateien) ein "pretty print" durchzuführen, bevor Sie diese in der Speicherschnittstelle speichern. Dadurch wird die Zeilenlänge limitiert, sodass es zu keinen Problemen mit dem CVSConflictEditor kommt.

QSC Team Coherence Client 7.1.4.30

<http://www.teamcoherence.com>

Mit den folgenden Schritten wird Altova DiffDog in Team Coherence Version Manager integriert:

1. Gehen Sie zu den Team Coherence Client-Optionen "Difference Viewer"
2. Geben Sie den vollständigen DiffDog-Dateipfad als Default Difference Viewer-Applikation an.
3. Definieren Sie als Parameter: "\$LF" "\$RF".

Warnung: Unter Umständen werden die neuen Einstellungen erst nach einem Windows Logoff angewendet.

Qumasoft QVCS Enterprise 2.1.18

<http://www.qumasoft.com/>

Mit den folgenden Schritten wird Altova DiffDog in Qumasoft QVCS-Enterprise integriert:

1. Fügen Sie das Qumasoft QVCS-Enterprise Installationsverzeichnis zur Umgebungsvariablen "Path" hinzu.
2. Gehen Sie zu den QVCS Enterprise User Preferences.
3. Aktivieren Sie unter "Utilities" das Kontrollkästchen "Use External Visual Compare Tool"
4. Definieren Sie als "Visual Compare Command Line":
<vollständiger DiffDog-Pfad> "file1Name file2Name"

Qumasoft QVCS Pro 3.10.18

<http://www.qumasoft.com/>

Mit den folgenden Schritten wird Altova DiffDog in Qumasoft QVCS-Pro integriert:

1. Gehen Sie zu den QVCS 3.10 Client Preferences.
2. Definieren Sie unter "Utilities" den vollständigen Pfad zu DiffDog als visuelles Vergleichstool mit den Parametern "%s %s"

Seapine Surround SCM Client 2009.0.0

<http://www.seapine.com/surroundscm.html>

Mit den folgenden Schritten wird Altova DiffDog in Seapine Surround SCM integriert:

1. Gehen Sie zu den Benutzeroptionen des Surround SCM Client und dem Abschnitt (Diff/Merge).

2. Bearbeiten Sie die Diff/Merge-Einstellungen, damit der Vergleich mit der ausgewählten Applikation durchgeführt wird.
3. Geben Sie mit den Parametern "%1" "%2" den vollständigen DiffDog-Pfad ein.
4. Starten Sie den Surround SCM Client und die Altova-Produkte erneut.

Sergey Antonovs HgSCC 1.0.1

http://www.newsupaplex.pp.ru/hgscs_news_eng.html

Mit den folgenden Schritten wird Altova DiffDog in Mercurial integriert:

1. Klicken Sie auf dem Register "Versionskontrolle" auf das Register **Erweitert**.
2. Wählen Sie das Vergleichstool "custom" und definieren Sie den vollständigen Pfad zu DiffDog.

SourceGear Fortress 1.1.4 Client

<http://www.sourcegear.com/fortress>

SourceGear Vault 4.1.4 Client

<http://www.sourcegear.com/vault>

Mit den folgenden Schritten wird Altova DiffDog in SourceGear Fortress and SourceGear Vault integriert:

1. Klicken Sie auf dem Register "Versionskontrolle" auf die Schaltfläche "Erweitert".
2. Definieren Sie die Diff/Merge Vault-Optionen durch Angabe des vollständigen DiffDog-Pfads und Verwendung der Argumente:
/ro1 /ro2 /title1:"%LEFT_LABEL%" /title2:"%RIGHT_LABEL%" "%LEFT_PATH%"
"%RIGHT_PATH%"

SourceGear SourceOffsite Client 4.2.0 (Windows)

<http://www.sourcegear.com/sos/>

Mit den folgenden Schritten wird Altova DiffDog in SourceGear SourceOffsite integriert:

1. Klicken Sie auf dem Register "Versionskontrolle" auf die Schaltfläche "Erweitert".
1. Definieren Sie als "External Programs", "Application for comparing files" den vollständigen DiffDog-Pfad.

TamTam CVS SCC 1.2.36,**TamTam SVN SCC 1.2.21**

<http://www.daveswebsite.com/software/tamtamGo>

Mit den folgenden Schritten wird Altova DiffDog in TamTam CVS SCC und TamTam SVN SCC integriert:

1. Klicken Sie auf dem Register "Versionskontrolle" auf die Schaltfläche "Erweitert".
2. Definieren Sie den vollständigen DiffDog-Pfad als externes Tool für Diff/Merge und Conflict.

Warnung: Bei Verwendung des Standard-Vergleichseditors CvsConflictEditor kann es zu Problemen beim Vergleich von Dateien mit sehr langen Zeilen kommen. Wir empfehlen Ihnen, an allen Dateien (v.a. ump-Dateien) ein "pretty print" durchzuführen, bevor Sie diese in der Speicherschnittstelle speichern. Dadurch wird die Zeilenlänge limitiert, sodass es zu keinen Problemen mit dem CvsConflictEditor kommt.

16 XMLSpy in MS Visual Studio

Sie können XMLSpy in die Microsoft Visual Studio IDE-Versionen 2005, 2008 und 2010 integrieren. Auf diese Weise werden die komplexen XML-Bearbeitungsfunktionen mit den umfangreichen Entwicklungsumgebung von Visual Studio kombiniert.

In diesem Abschnitt wird Folgendes beschrieben:

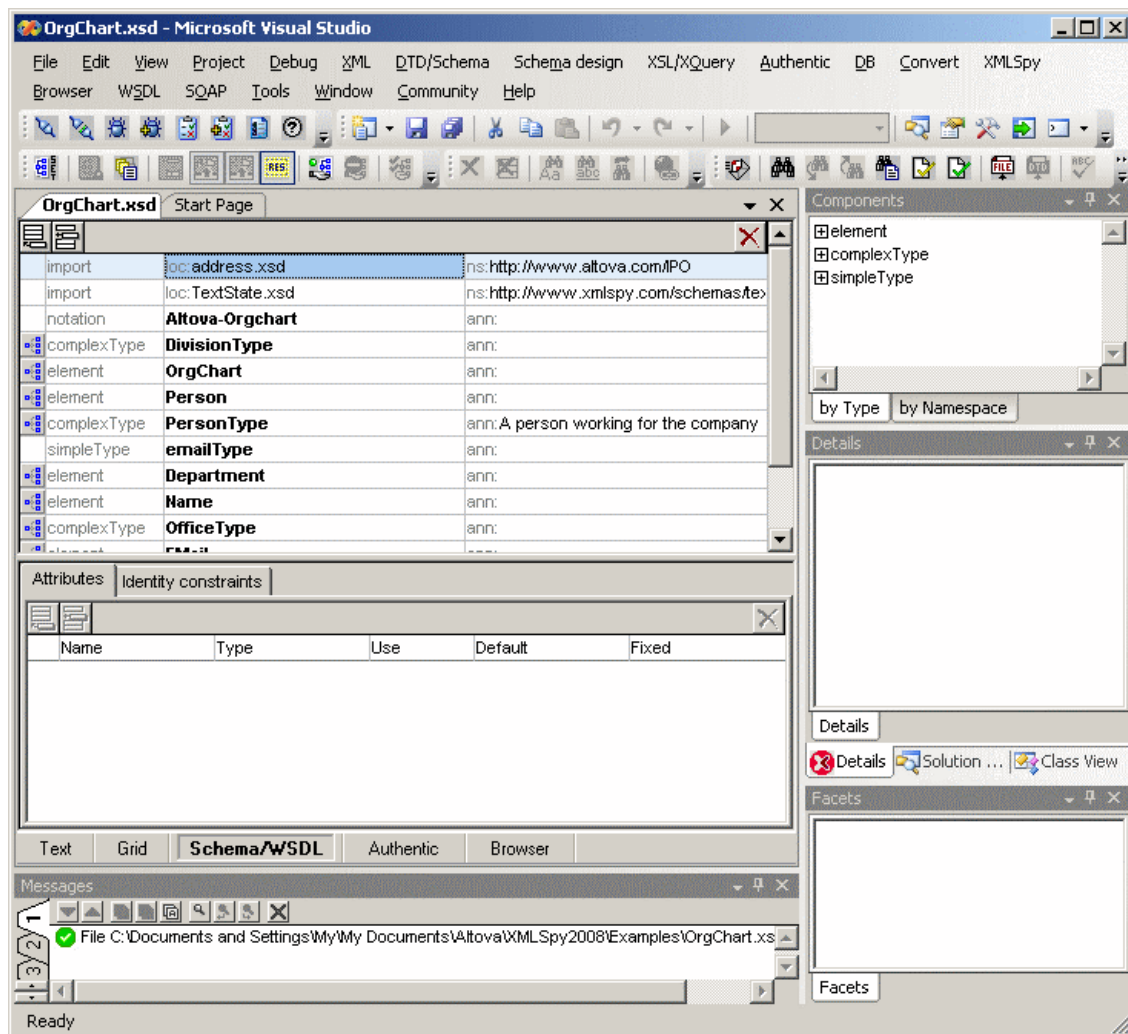
- der [allgemeine Installationsvorgang](#) und die Integration des XMLSpy Plugin in Visual Studio.
- [Unterschiede](#) zwischen der Visual Studio-Version und der Standalone-Version.
- [Die Debugger von XMLSpy](#) in Visual Studio.
- [Bekannte Probleme](#).

Anmerkung: Die Abbildungen in diesem Abschnitt stammen aus der Visual Studio Version 2005. Wenn Sie eine andere Version verwenden, kann es eventuell Unterschiede dazu geben.

16.1 Installieren des XMLSpy Plugin

Um das XMLSpy Plugin für Visual Studio zu installieren, gehen Sie folgendermaßen vor:

- Installieren Sie Microsoft Visual Studio
- Installieren Sie XMLSpy (Enterprise oder Professional Edition)
- Laden Sie das XMLSpy Integrationspaket für Microsoft Visual Studio herunter und starten Sie es. Sie finden das Installationspaket auf der XMLSpy (Enterprise und Professional Edition) Download-Seite unter www.altova.com. (**Bitte beachten Sie:** Sie müssen das Integrationspaket für Ihre jeweilige Version von XMLSpy (die aktuelle Version ist 2012) und Edition (Enterprise oder Professional) verwenden).



Nach Installation des Integrationspakets können Sie XMLSpy innerhalb der Visual Studio-Umgebung verwenden.

Aktivierung des Plug-In

Unter Umständen wurde das Plug-In bei der Installation nicht automatisch aktiviert. In diesem Fall gehen Sie vor wie folgt:

1. Navigieren Sie zu dem **Verzeichnis**, in dem die ausführbare Visual Studio IDE-Datei installiert wurde, z.B. C:\Program Files\MS Visual Studio\Common7\IDE.

2. Geben Sie in die Befehlszeile den folgenden Befehl ein `devenv.exe /setup`.
3. Warten Sie, bis der Prozessor fertig ist, bevor Sie die Applikation in Visual Studio starten.

Anmerkung: Die Abbildungen in diesem Abschnitt stammen aus der Visual Studio Version 2005. Möglicherweise bestehen Unterschiede zu der von Ihnen verwendeten Version.

16.2 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

Die folgenden XMLSpy Funktionen stehen in Visual Studio nicht zur Verfügung:

- Die Skripting-Umgebung (**Extras | XMLSpy Optionen | Skripting**) wird derzeit nicht unterstützt.
- Separates Browser-Fenster (eine Option auf dem Register "Ansicht" (**Extras | Optionen**) wird nicht unterstützt. Das bedeutet, dass sich die Text- und die Browser-Ansicht immer im selben Fenster befinden
- Die Textstatussymbole der [Authentic-Ansicht](#) werden nicht unterstützt.
- 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 Befehle des Menüs "Globale Ressourcen")
- **Extras | XMLSPY-Optionen**

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 [Einführung](#))

- 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 Funktion, unterschiedlicher Befehl

Einige Funktionalitäten von XMLSpy stehen in Visual Studio unter einem anderen Namen zur Verfügung. Diese sind:

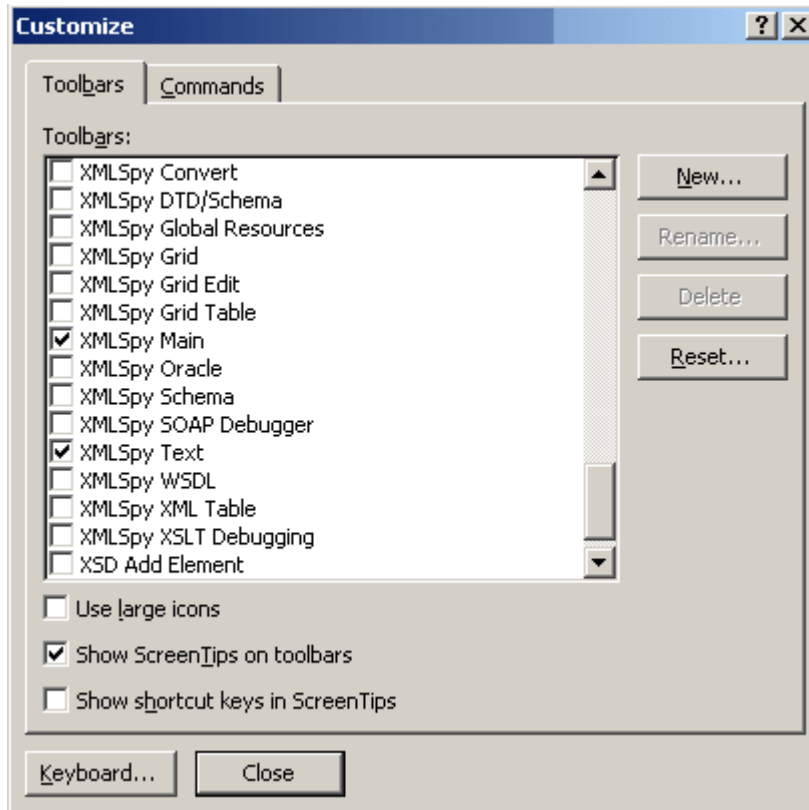
XMLSpy	Visual Studio	Funktionalität
Datei Öffnen Zu URL wechseln	Datei Öffnen Website	Öffnet die Datei über eine URL
Zu URL wechseln URL speichern	Datei XMLSpy-Datei speichern unter My Network Places	Speichert die Datei unter einer URL-Adresse

XMLSpy Befehle als Visual Studio-Befehle

Einige **XMLSpy** Befehle stehen auf der Benutzeroberfläche von Visual Studio als Visual

Studio-Befehle zur Verfügung:

- **Rückgängig, Wiederholen:** 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" (**Extras | Anpassen**) enthalten sowohl Befehle aus Visual Studio als auch **XMLSpy**.



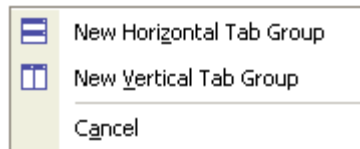
- **Ansichten:** Im Menü **Ansicht** enthalten die beiden Befehle **XMLSpy Tool-Fenster** und **XMLSpy-Ansicht** Optionen zum Ein- und Ausblenden der Eingabehilfenfenster und anderer Seitenleisten und zum Wechseln zwischen den Bearbeitungsansichten und zum Ein- und Ausblenden von Bearbeitungshilfen.
- **XMLSpy Hilfe:** Dieses **XMLSpy** Menü wird als Untermenü im Hilfemenü von Visual Studio angezeigt.

Anmerkung: Die Abbildungen in diesem Abschnitt stammen aus der Visual Studio Version 2005. Wenn Sie eine andere Version verwenden, kann es eventuell Unterschiede dazu geben.

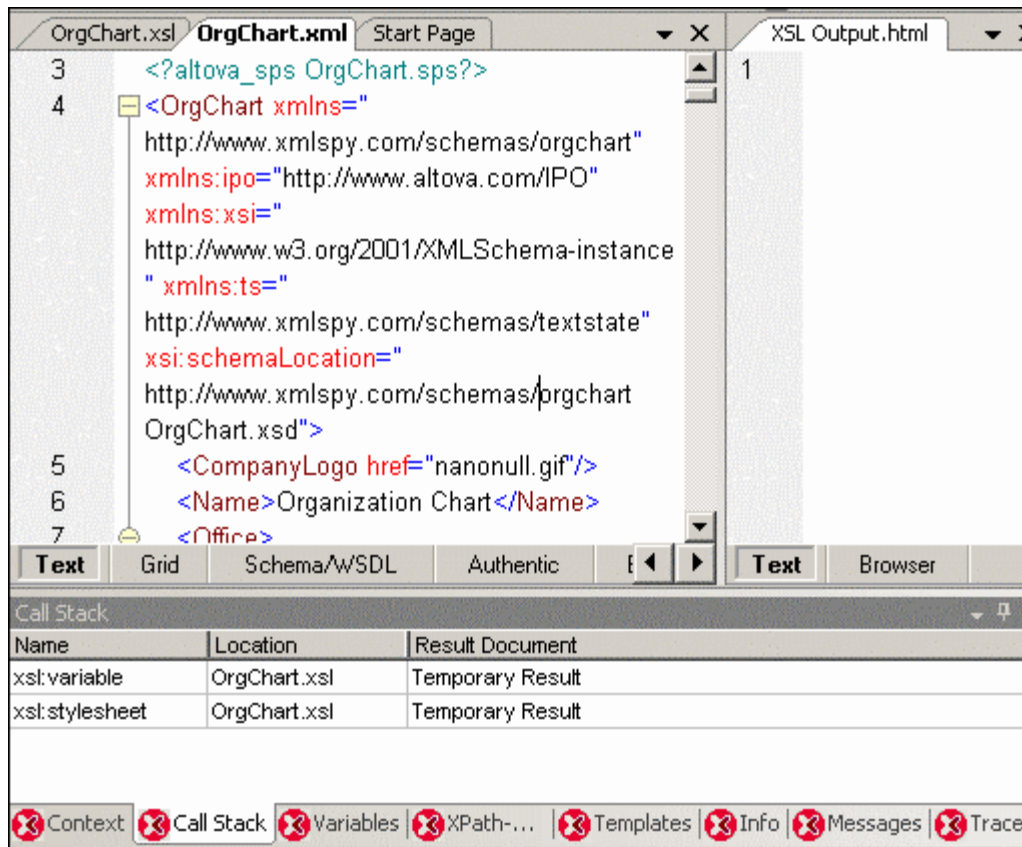
16.3 Die XMLSpy Debugger in Visual Studio

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).



Anmerkung: Die Abbildungen in diesem Abschnitt stammen aus der Visual Studio Version 2005. Wenn Sie eine andere Version verwenden, kann es eventuell Unterschiede dazu geben.

16.4 Bekannte Probleme

In diesem Abschnitt werden einige bekannte Probleme, die im Zusammenhang mit den verschiedenen Editionen von MS Visual Studio und XMLSpy auftreten, beschrieben.

17 XMLSpy in Eclipse

Eclipse 3.x ist eine Open-Source-Umgebung, in die verschiedene Arten von Applikationen in Form von Plug-Ins integriert werden können.

Das XMLSpy Plugin für die Eclipse-Plattform ist ein Eclipse Plug-In, das Ihnen gestattet, auf die Funktionalitäten von XMLSpy über die Eclipse 3.5 / 3.6 / 3.7-Plattform zuzugreifen. Es steht auf Windows Plattformen zur Verfügung. In diesem Abschnitt wird beschrieben, wie man das XMLSpy Plugin für Eclipse [installiert](#) und die [XMLSpy-](#) und die [XMLSpy-Debugger Perspektive einrichtet](#). Anschließend stehen die Komponenten der Benutzeroberfläche von XMLSpy und die Menübefehle von XMLSpy über die Benutzeroberfläche von Eclipse zur Verfügung.

Anmerkung:

- Die Versionskontrollfunktionen, die in der Standalone-Version verfügbar sind, werden in der Eclipse-Version nicht unterstützt.

17.1 Installation des XMLSpy Plugin für Eclipse

Stellen Sie vor Installation des XMLSpy Plugin für Eclipse sicher, dass die folgenden Komponenten bereits installiert sind:

- XMLSpy Enterprise oder Professional Edition.
- Java Runtime Environment (JRE) Version 1.5 oder höher, welches für Eclipse benötigt wird. Empfohlen wird JRE5. Nähere Informationen dazu finden Sie auf der [Eclipse Website](#).
- Eclipse-Plattform 3.5 / 3.6 / 3.7

Nachdem Sie diese Komponenten installiert haben, können Sie das XMLSpy Plugin für Eclipse, das imXMLSpy Integrationspaket (*siehe unten*) enthalten ist, installieren. **Bitte beachten Sie:** Sie müssen das entsprechende Integrationspaket für Ihre XMLSpy-Version (die aktuelle Version ist 2012) installieren.

Anmerkung zu JRE

Wenn beim Öffnen eines Dokuments die unten stehende Fehlermeldung angezeigt wird,

```
java.lang.UnsupportedClassVersionError: com/altova/....  
(Unsupported major.minor version 49.0)
```

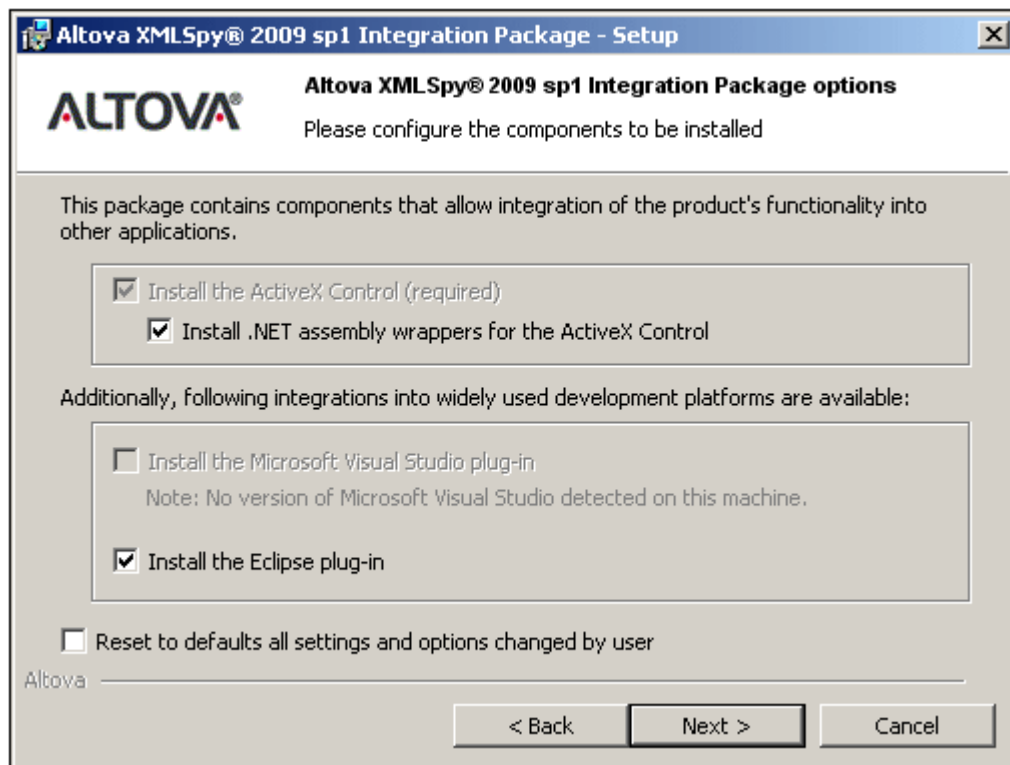
dann weist das darauf hin, dass Eclipse ein älteres JRE verwendet. Eclipse verwendet zum Suchen der Datei `javaw.exe` die Umgebungsvariable `PATH`.

Das Problem kann gelöst werden, indem Sie die `PATH`-Umgebungsvariable ändern, sodass zuerst eine neuere Version gefunden wird. Alternativ dazu können Sie Eclipse auch mit dem Befehlszeilenparameter `-vm` starten und den Pfad zu einer `javaw.exe` Version 1.5 oder höher angeben.

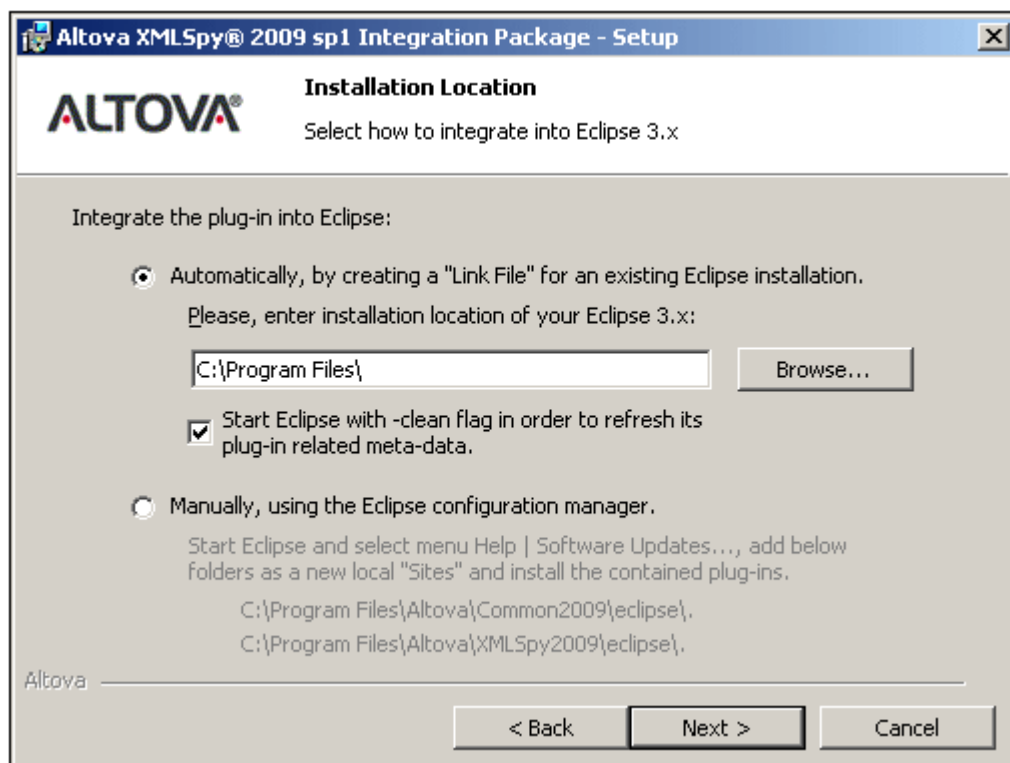
XMLSpy Integrationspaket

Das XMLSpy Plugin für Eclipse ist im XMLSpy-Integrationspaket enthalten und wird mit diesem installiert. Führen Sie die Installation folgendermaßen durch:

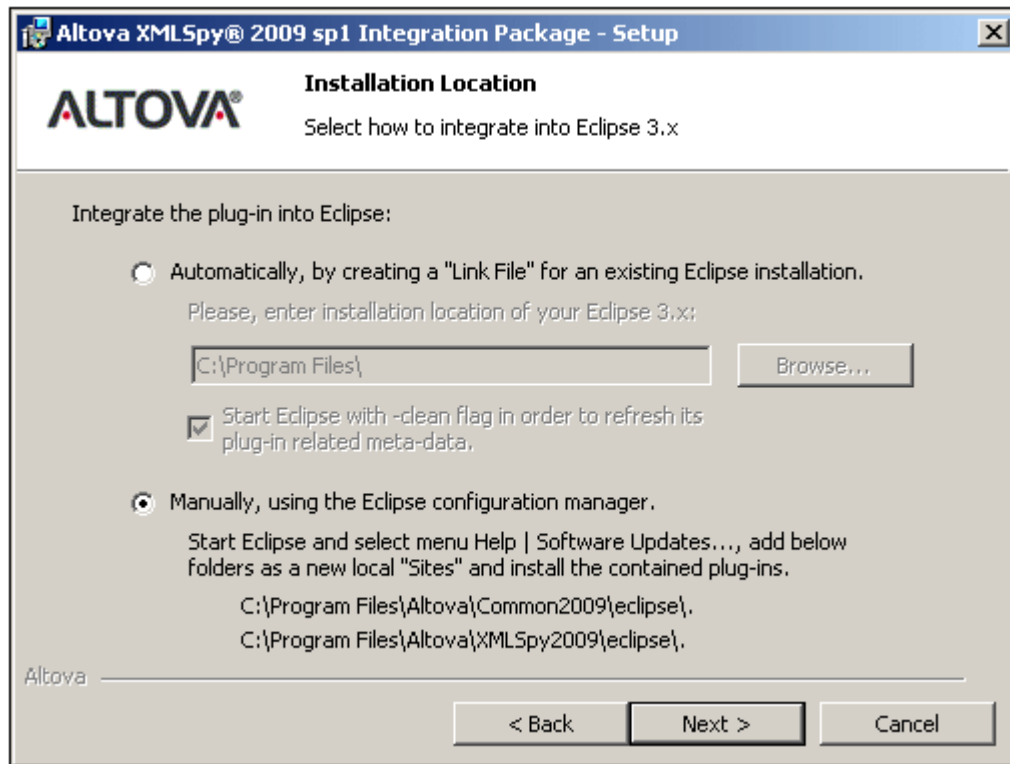
1. Stellen Sie sicher, dass XMLSpy, JRE und Eclipse bereits installiert sind (*siehe oben*).
2. Laden Sie das XMLSpy Integrationspaket von der [Komponenten-Download-Seite](#) der [Altova Website](#) herunter und installieren Sie es. Bei der Installation gibt es zwei wichtige Schritte, die weiter unten in Schritt 3 und 4 näher beschrieben sind.
3. Während der Installation des XMLSpy Integrationspakets erscheint ein Dialogfeld, in dem Sie gefragt werden, ob das XMLSpy Plugin für Eclipse installiert werden soll (*siehe Abbildung unten*). Überprüfen Sie die Option und klicken Sie auf **Weiter**.



4. Im nächsten Dialogfeld (Eclipse)-Installationspfad (*Abbildung unten*), können Sie auswählen ob der Installationsassistent das XMLSpy Plugin während der Installation in Eclipse integrieren soll (Option "Automatisch") oder ob Sie das Plug in später (über die Benutzeroberfläche von Eclipse) installieren möchten.



Wir empfehlen die Integration über den Installationsassistenten vorzunehmen. Aktivieren Sie dazu die Option "Automatisch" und suchen Sie nach dem Ordner, in dem sich die ausführbare Datei von Eclipse befindet (`eclipse.exe`). Klicken Sie anschließend auf **Weiter**. Wenn Sie sich für eine manuelle Integration des XMLSpy Plug-in für Eclipse in Eclipse entscheiden, wählen Sie die Option "manuell". Eine genauere Anleitung dazu finden Sie weiter unten.

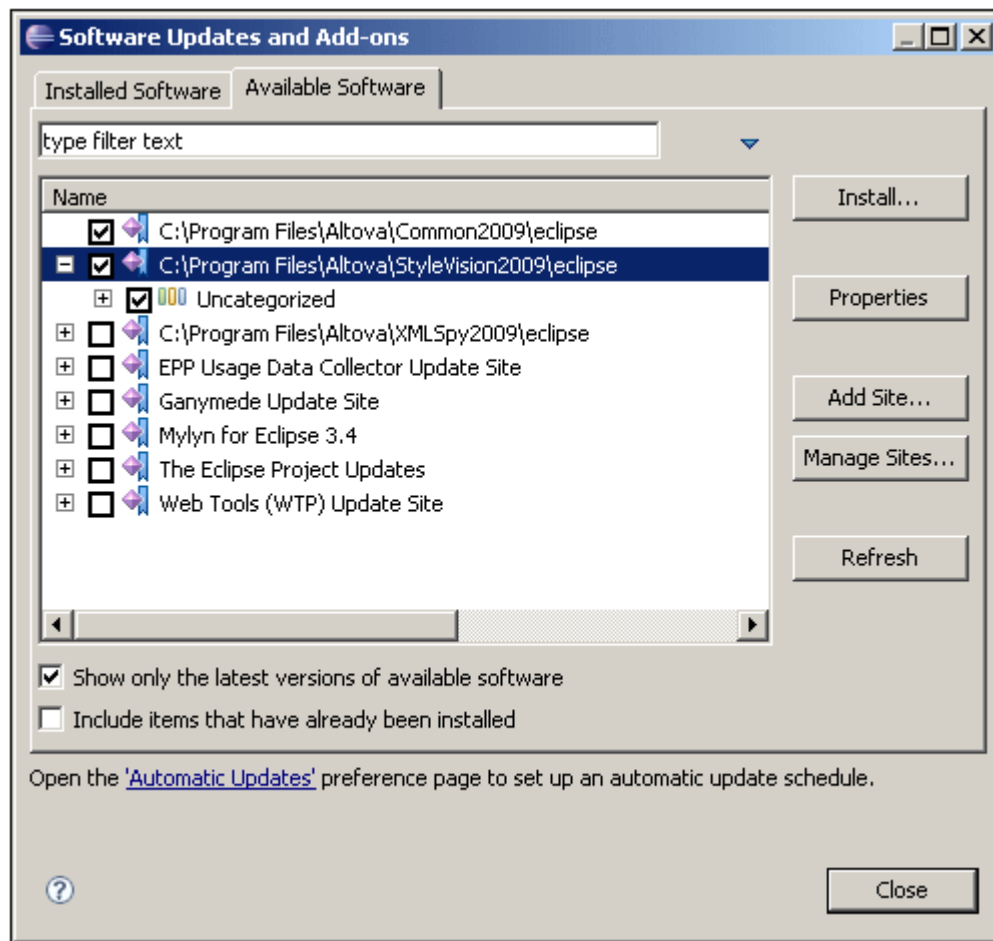


5. Stellen Sie die Installation fertig. Wenn Sie die automatische Integration wählen, wird das XMLSpy Plugin für Eclipse in Eclipse integriert und steht beim nächsten Start von Eclipse bereits zur Verfügung. Andernfalls müssen Sie das Plug-in manuell integrieren (siehe unten).

Manuelle Integration des XMLSpy Plugin in Eclipse

Um das XMLSpy Plugin für Eclipse manuell zu integrieren, gehen Sie folgendermaßen vor:

1. Klicken Sie in Eclipse auf den Menübefehl **Help | Install New Software**.
2. Klicken Sie im Dialogfeld "Software Updates and Add-Ons", das daraufhin angezeigt wird, auf das Register "Available Updates" (*Abbildung unten*).
3. Klicken Sie auf die Schaltfläche **Add Site**.
4. Klicken Sie im Dialogfeld "Add Site" auf die Schaltfläche **Local**.
5. Navigieren Sie zum Ordner `c:\Program Files\Altova\Common2012\eclipse`, wählen Sie ihn aus und klicken Sie auf **OK**.
6. Wiederholen Sie die Schritte 3 bis 5 und wählen Sie diesmal den Ordner `c:\Program Files\Altova\StyleVision2012\eclipse` aus.
7. Die beiden hinzugefügten Ordner werden auf dem Register "Available Software" (*Abbildung unten*) angezeigt. Aktivieren Sie bei jedem Ordner das Kontrollkästchen auf der obersten Ebene, um die Plug-ins auszuwählen und klicken Sie auf die Schaltfläche **Install**.



8. Daraufhin wird ein Dialogfeld geöffnet, in dem Sie überprüfen können, ob die angehakten Komponenten installiert werden.
9. Klicken Sie zum Fortfahren auf **Next**. Daraufhin wird das Dialogfeld "Review License" angezeigt.
10. Lesen Sie die Lizenzbedingungen und klicken Sie auf *I accept the terms....*, wenn Sie damit einverstanden sind. Klicken Sie anschließend auf **Finish**, um die Installation fertig zu stellen.

Falls es Probleme mit dem Plug-in gibt (z.B. fehlende Symbole), starten Sie Eclipse mit dem `-clean` Flag.

Derzeit installierte Version

Um zu sehen, welche Version des XMLSpy Plugin für Eclipse installiert ist, wählen Sie die Eclipse-Menüoption **Help | About Eclipse Platform**. Wählen Sie anschließend das Symbol XMLSpy.

17.2 XMLSpy-Einstiegspunkte in Eclipse

Um Zugriff auf die Funktionalitäten von XMLSpy zu erhalten, stehen Ihnen in Eclipse die folgenden Einstiegspunkte zur Verfügung:

- [XMLSpy Perspektive](#): Hier stehen Ihnen die Funktionen der Benutzeroberfläche von XMLSpy auf der Benutzeroberfläche von Eclipse zur Verfügung.
- [XMLSpy Symbolleisten-Schaltflächen](#): Diese Schaltflächen bieten Zugriff auf die XMLSpy Hilfe und die Funktion "Neues Dokument erstellen".

XMLSpy-Perspektive

Eine Perspektive ist in Eclipse eine konfigurierte Ansicht der Benutzeroberfläche mit Funktionalitäten aus verschiedenen Applikationen. Wenn das XMLSpy Plugin für Eclipse in Eclipse integriert wird, wird automatisch eine XMLSpy-Standardperspektive erstellt. Diese Perspektive ist eine Benutzeroberfläche, die die Elemente der Benutzeroberfläche von XMLSpy enthält: Bearbeitungsansichten, Eingabehilfen und andere Seitenleisten.

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 bei jeder aktiven Datei wechseln und so die Datei in einer anderen Umgebung bearbeiten bzw. verarbeiten. Perspektiven haben daher zwei wichtige Vorteile:

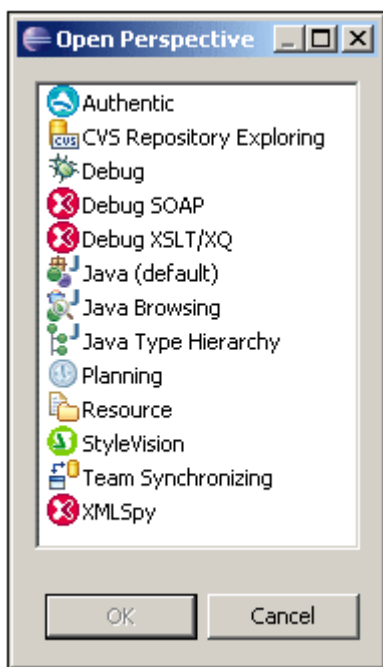
1. Die Arbeitsumgebung für die aktive Datei kann schnell gewechselt werden und
2. Sie können zwischen Dateien wechseln, ohne eine neue Entwicklungsumgebung öffnen zu müssen (die mit der Datei verknüpfte Umgebung steht in einer Perspektive zur Verfügung).

Um mit der XMLSpy-Perspektive zu arbeiten, müssen Sie Folgendes tun:

- 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 | Open Perspective | Other**. Wählen Sie im Dialogfeld, das jetzt angezeigt wird (*Abbildung unten*), XMLSpy aus und klicken Sie auf **OK**.



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.

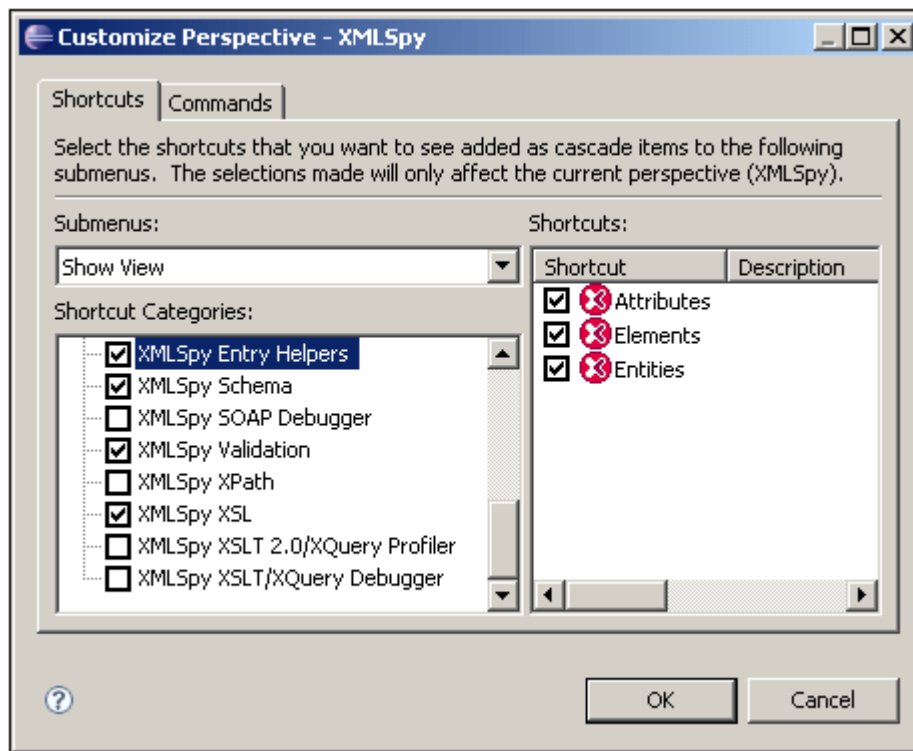
Perspektiven können auch gewechselt werden, wenn eine Datei geöffnet oder zur aktiven gemacht wird. Die Perspektive der Applikation, die mit dem Dateityp einer Datei verknüpft ist, wird automatisch geöffnet, wenn diese Datei das erste Mal geöffnet wird. Vor dem Wechseln der Perspektive wird ein Dialogfeld 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**.

Festlegen der Einstellungen für die XMLSpy-Perspektive

Zu den Einstellungen einer Perspektive gehören: (i) eine Einstellung, um die Perspektive automatisch zu ändern, wenn eine Datei eines verknüpften Dateityps geöffnet wird (*siehe oben*) und (ii) Optionen zum Inkludieren oder Ausblenden einzelner XMLSpy-Symboleisten. 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**.

Anpassen der XMLSpy Perspektive

Mit Hilfe der Anpassungsoptionen können Sie festlegen, welche Shortcuts und Befehle in der Perspektive enthalten sein sollen. Um dieses Dialogfeld aufzurufen (*in der Abbildung unten sehen Sie das Dialogfeld für die XMLSpy-Perspektive*) machen Sie die Perspektive (in diesem Fall die XMLSpy-Perspektive) aktiv und wählen Sie den Befehl **Window | Customize Perspective** aus.



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.

Auf dem Register "Commands" können Sie Befehlsgruppen hinzufügen. Um die Befehle in einer Befehlsgruppe anzuzeigen, wählen Sie aus den verfügbaren Befehlsgruppen (die im Bereich "Command Groups" angezeigt werden) die gewünschte Gruppe aus. Die Befehle in dieser Gruppe werden auf der rechten Seite hierarchisch geordnet im Menü, in dem sie zu sehen sein werden, angezeigt. Damit die Befehlsgruppe inkludiert wird, aktivieren Sie ihr Kontrollkästchen.

Klicken Sie auf **OK**, um die Anpassung abzuschließen und damit die Änderungen wirksam werden.

XMLSpy-Symbolleisten-Schaltflächen

Es werden automatisch zwei XMLSpy-Schaltflächen in der Symbolleiste angelegt (*Abbildung unten*).

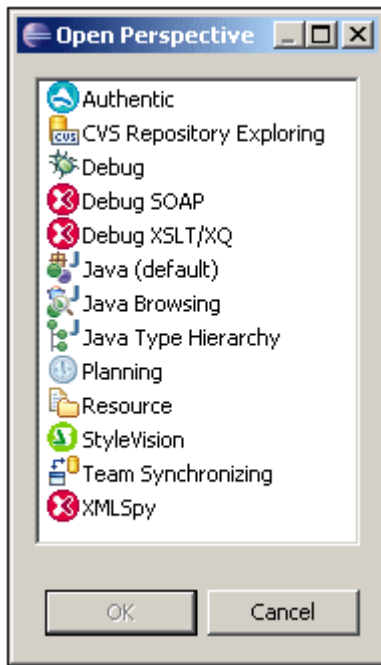


Diese dienen zum: (i) Öffnen der XMLSpy Hilfe und (ii) Erstellen neuer XMLSpy-Dokumente.

17.3 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.

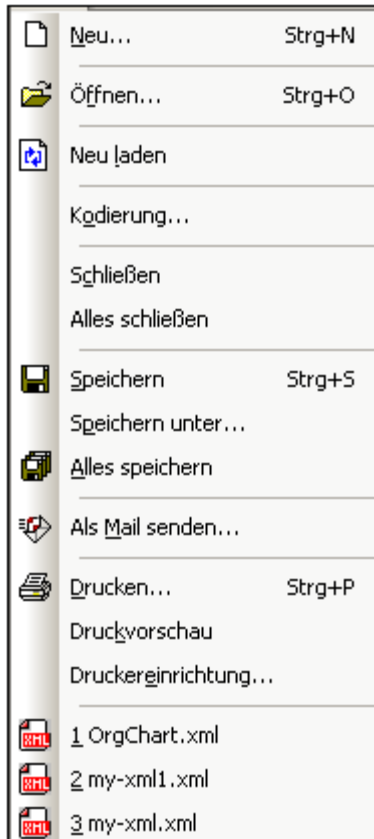
18 Benutzerreferenz

Im Referenzteil der Dokumentation finden Sie eine Beschreibung aller XMLSpy Menübefehle und deren Verwendung. Wir haben versucht, das Handbuch so umfassend wie möglich zu gestalten. Sollten Sie trotzdem Fragen zu Themen haben, die in der vorliegenden Dokumentation nicht behandelt werden, werfen Sie bitte einen Blick in die FAQs und die Diskussionsforen auf der Altova Website. Sollten Ihre Fragen damit noch immer nicht beantwortet sein, kontaktieren Sie uns bitte über unser [Support Center](#) auf der Altova Website.

Bitte beachten Sie, dass das Menü [Datei](#) und [Bearbeiten](#) 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.

18.1 Menü "Datei"

Das Menü "Datei" enthält Befehle, die zum Arbeiten mit Dateien benötigt werden. Die Befehle sind in der aus Windows Software-Produkten bekannten Reihenfolge angeordnet.



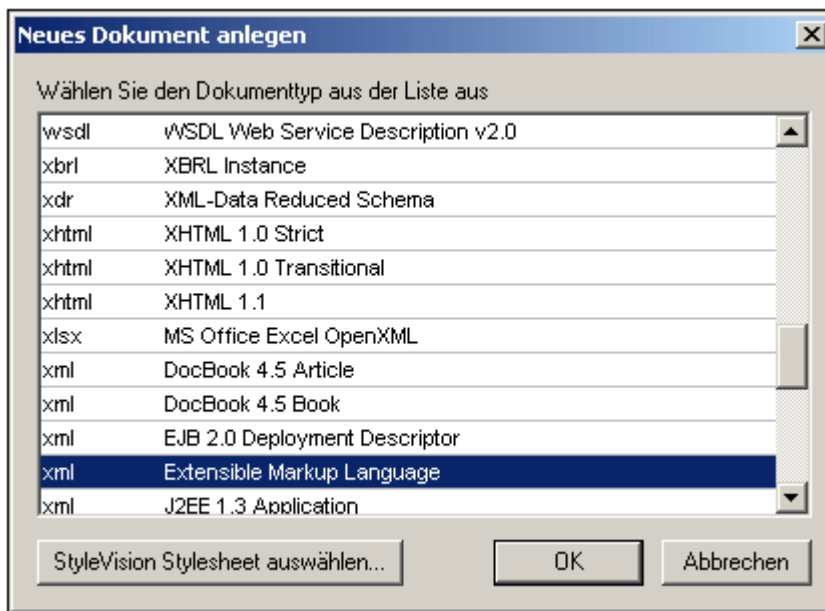
Zusätzlich zu den Standardbefehlen [Neu](#), [Öffnen](#), [Speichern](#), [Drucken](#), [Druckereinarbeitung](#), und [Beenden](#) bietet XMLSpy eine Reihe von XML- und applikationsspezifischen Befehlen.

18.1.1 Neu



Strg+N

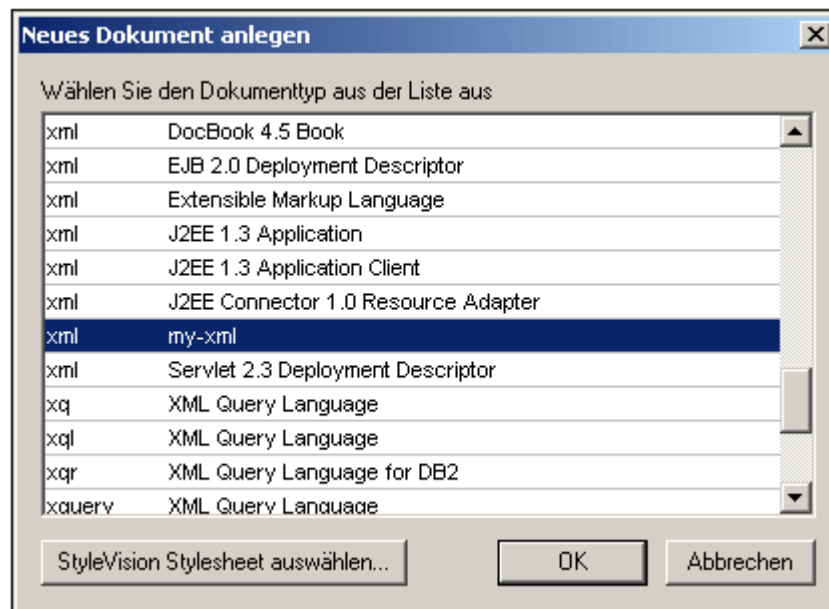
Der Befehl **Neu** wird verwendet, um ein neues Dokument zu erstellen. Wenn Sie auf **Neu** klicken, wird das Dialogfeld "Neues Dokument anlegen" geöffnet, in dem Sie auswählen können, welche Art von Dokument erstellt werden soll. Falls das Dokument, das Sie erstellen wollen, nicht aufgelistet ist, wählen Sie "XML", und ändern Sie die Erweiterung des Dokuments beim Speichern. Beachten Sie, dass Sie neue Dateiartern in die Liste einfügen können, indem Sie diese unter [Extras | Optionen auf dem Register "Dateiartern"](#) eintragen.



Erstellen von Vorlagen für neue Dokumente

Sie können mehrere Vorlagen für verschiedene Dateiartern erstellen, die Sie anschließend direkt über das Dialogfeld "Neues Dokument anlegen" öffnen und bearbeiten können. Um Ihre eigene Vorlage zu erstellen, sodass Sie in diesem Dialogfeld in der Liste der Dokumente angezeigt wird, müssen Sie zuerst ein Vorlagendokument erstellen und dieses dann in dem Ordner, der alle Vorlagen enthält, speichern. Gehen Sie vor wie folgt:

1. Öffnen Sie den Ordner **XMLSpy\Template** 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. my-xml.xml zu speichern. Sie haben nun eine Vorlage mit dem Namen `my-xml.xml`, die im Dialogfeld "Neues Dokument anlegen" in der Dateiliste angezeigt wird.



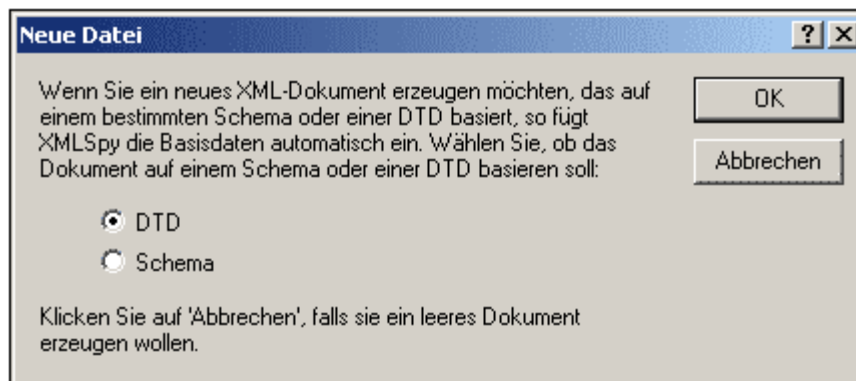
4. Zum Öffnen der Vorlage wählen Sie **Datei | Neu** und anschließend die Vorlage (in diesem Fall `my-xml`).

Bitte beachten Sie: Um eine Vorlage zu löschen, öffnen Sie das Vorlagen-Verzeichnis und löschen Sie dort die gewünschte Vorlage.

DTD / XML-Schema einem neuen XML-Dokument zuweisen

Wenn Sie ein neues Dokument einer bestimmten Art erstellen, das auf einem Standardschema (DTD oder XML-Schema) basiert, 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 eine neue Datei erstellen, für die das Schema nicht bekannt ist (z.B. eine XML-Datei), werden Sie aufgefordert, ein Schema (DTD oder XML-Schema) mit dem zu erstellenden Dokument zu verknüpfen.

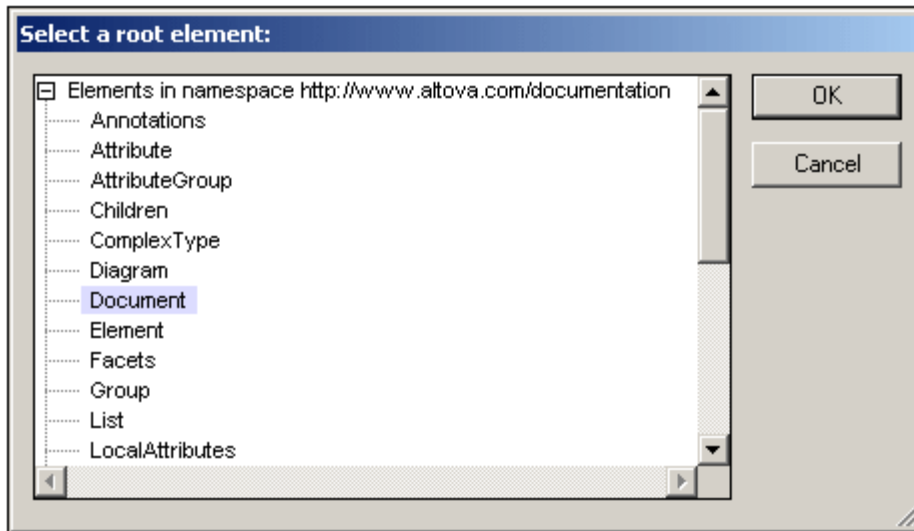


Wenn Sie eine DTD oder ein XML-Schema mit Ihrem Dokument verknüpfen, klicken Sie im Dialogfeld "Neue Datei" auf OK, um den Pfad zum Schema anzugeben. Wenn Sie in diesem Dialogfeld auf "Abbrechen" klicken, wird eine neue Datei erstellt, die mit keinem Schema

verknüpft ist.

Angabe des Dokumentsymbols eines neuen XML-Dokuments

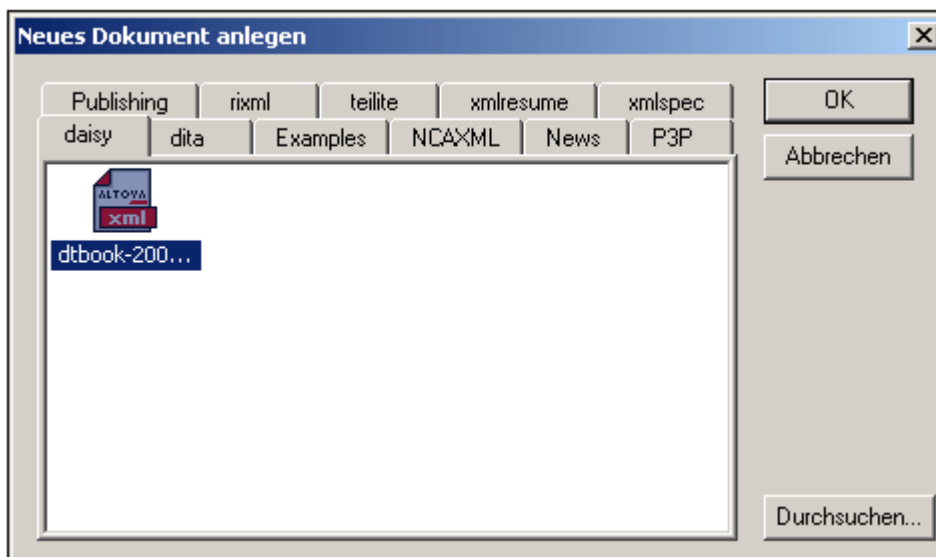
Wenn Sie ein XML-Schema auswählen, kann dieses mehr als ein globales Element enthalten. Jedes davon kann potenziell ein Dokument- (oder Root) Element sein. Im Dialogfeld "Wählen Sie das Root-Element" können Sie das Root-Element für das XML-Dokument auswählen. Dieses Dialogfeld wird angezeigt, wenn Sie im Dialogfeld "Neue Datei" Schema auswählen und wenn das XML-Schema mehrere globale Elemente enthält.



Das neue XML-Dokument wird mit diesem Element als Dokument-Element erstellt.

Zuweisen eines StyleVision Power Stylesheet beim Erstellen eines neuen Dokuments

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 Abbildung oben) 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 StyleVision Power Stylesheets zur Verfügung. Alternativ dazu können Sie das gewünschte StyleVision Power Stylesheet auch über die Schaltfläche **Durchsuchen...** suchen. Die Register in diesem Dialogfeld entsprechen den Ordnern im Ordner `sps/Template` im Programmordner der Applikation.

18.1.2 Öffnen

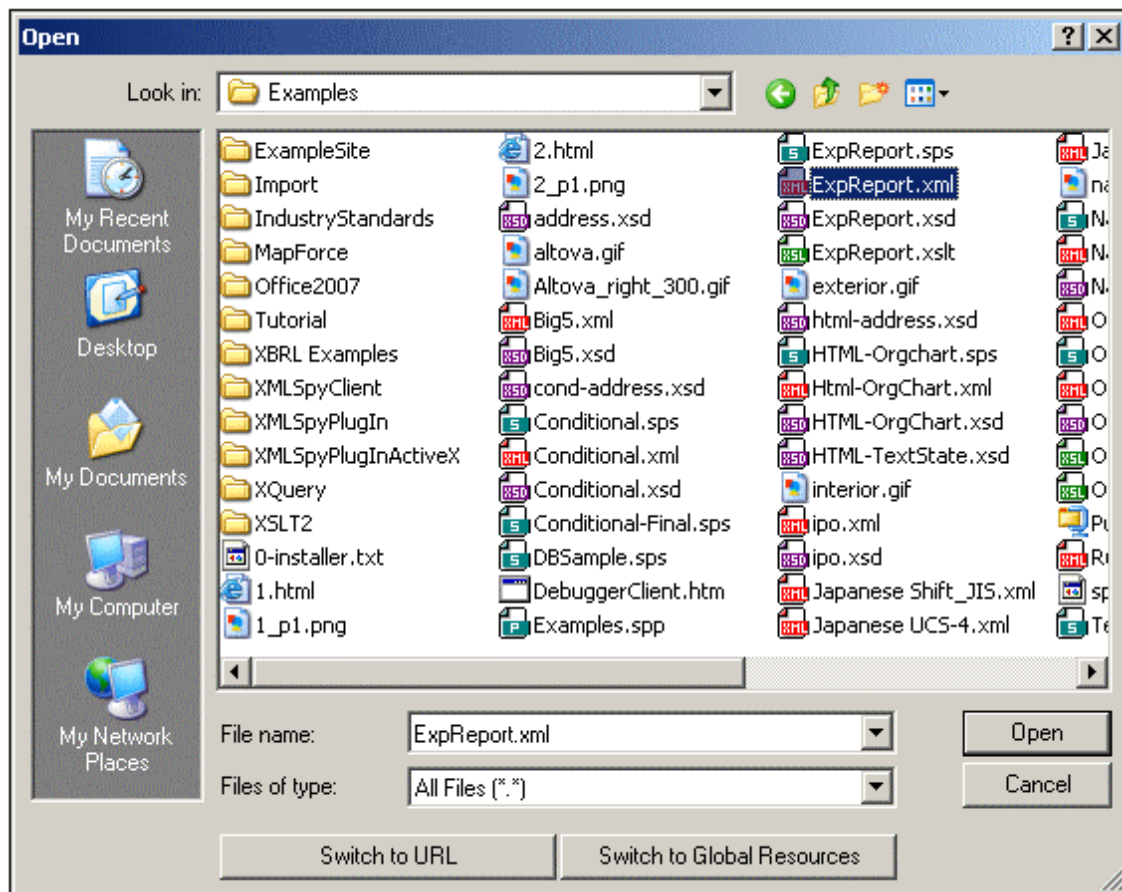


Strg+O

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 Dateitypen kann auf dem [Register "Dateitypen" unter Extras | Optionen](#) 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 **Wohlgeformtheit prüfen (F7)**, um die Datei erneut zu überprüfen. Wenn Sie die Option [Validierung beim Öffnen der Datei](#) aktiviert haben und die Datei ungültig ist, wird ebenfalls eine Fehlermeldung angezeigt. Beheben Sie den Fehler und klicken Sie auf **XML | XML validieren (F8)**, um das Dokument erneut zu validieren.

Auswählen von Dateien über URLs und globale Ressourcen

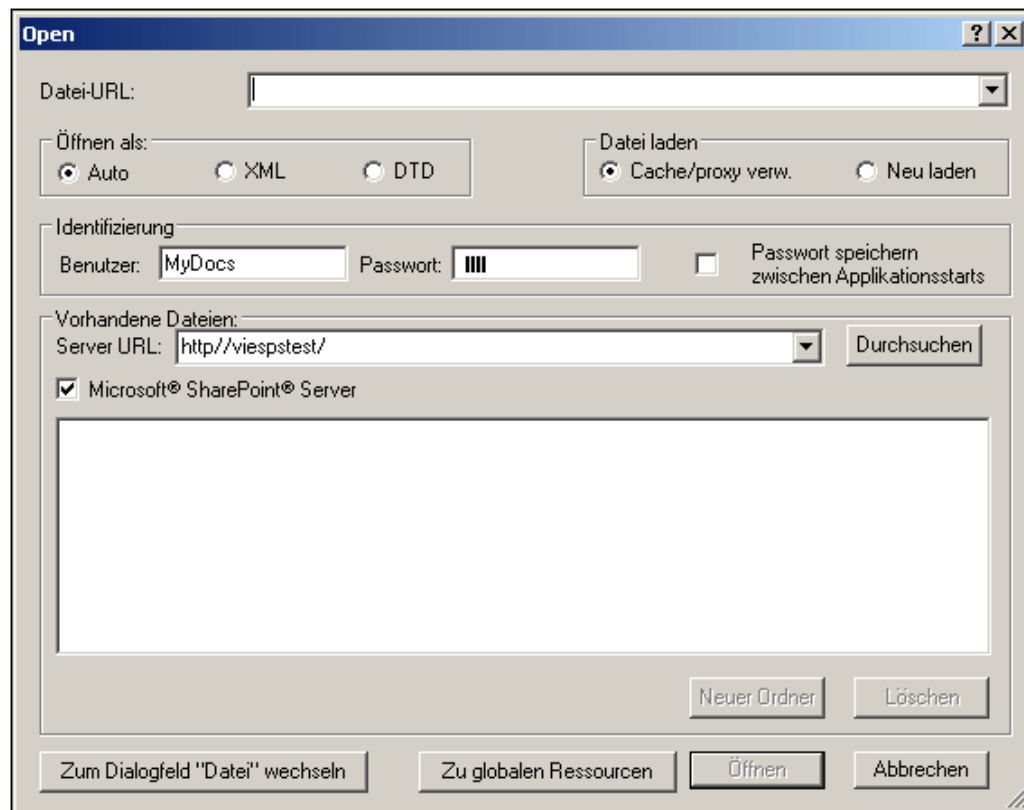
In einigen Dialogfeldern zum Öffnen und Speichern von Dateien können Sie auswählen, ob Sie die gewünschte Datei auswählen oder eine Datei über eine URL oder eine globale Ressource (siehe *Abbildung unten*) speichern möchten. Wählen Sie den Befehl **Zu URL wechseln** oder **Zu globaler Ressource wechseln** um eine dieser Optionen auszuwählen.



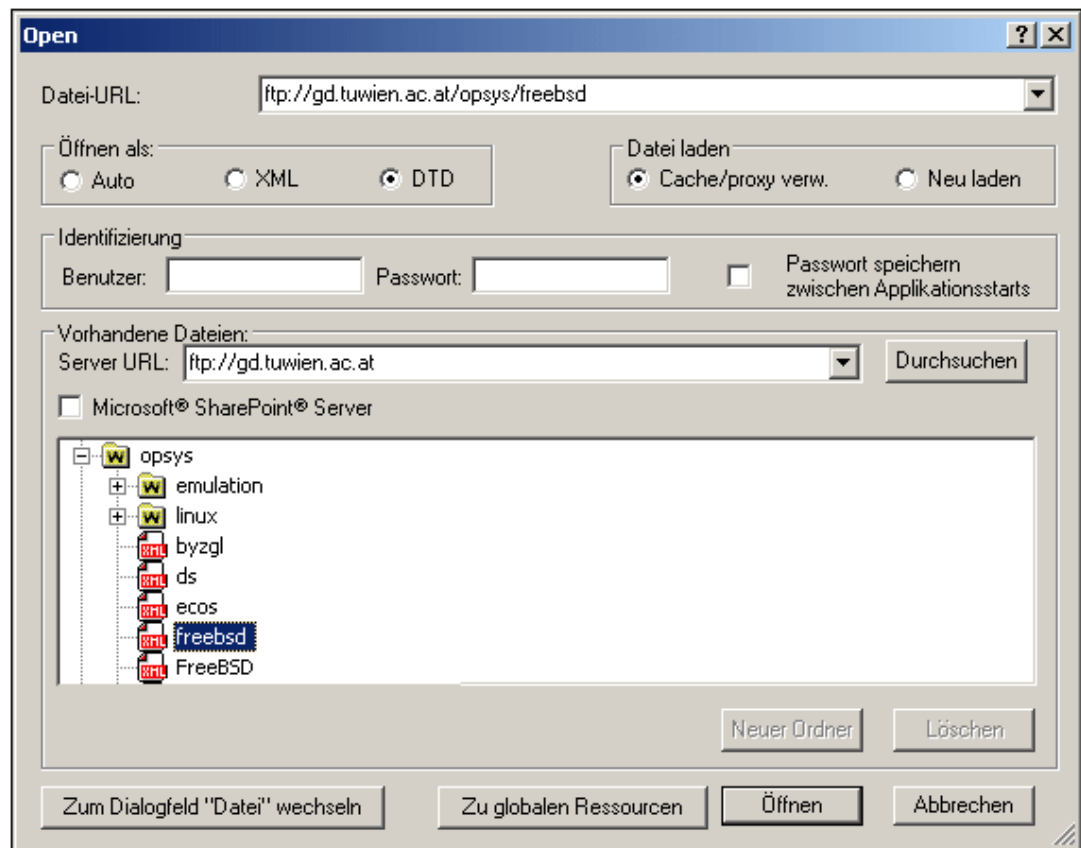
Auswählen von Dateien über URLs

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" in den URL-Modus (Abbildung unten).



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 anzuzeigen und darin zu navigieren.
5. Klicken Sie in der Ordnerstruktur auf die gewünschte Datei.



Die Datei-URL wird im Feld "Datei-URL" (*Abbildung oben*) angezeigt. Die Schaltfläche "Öffnen" wird erst jetzt aktiv.

6. Klicken Sie auf "Öffnen", um die Datei zu laden. Die Datei wird im Hauptfenster angezeigt.

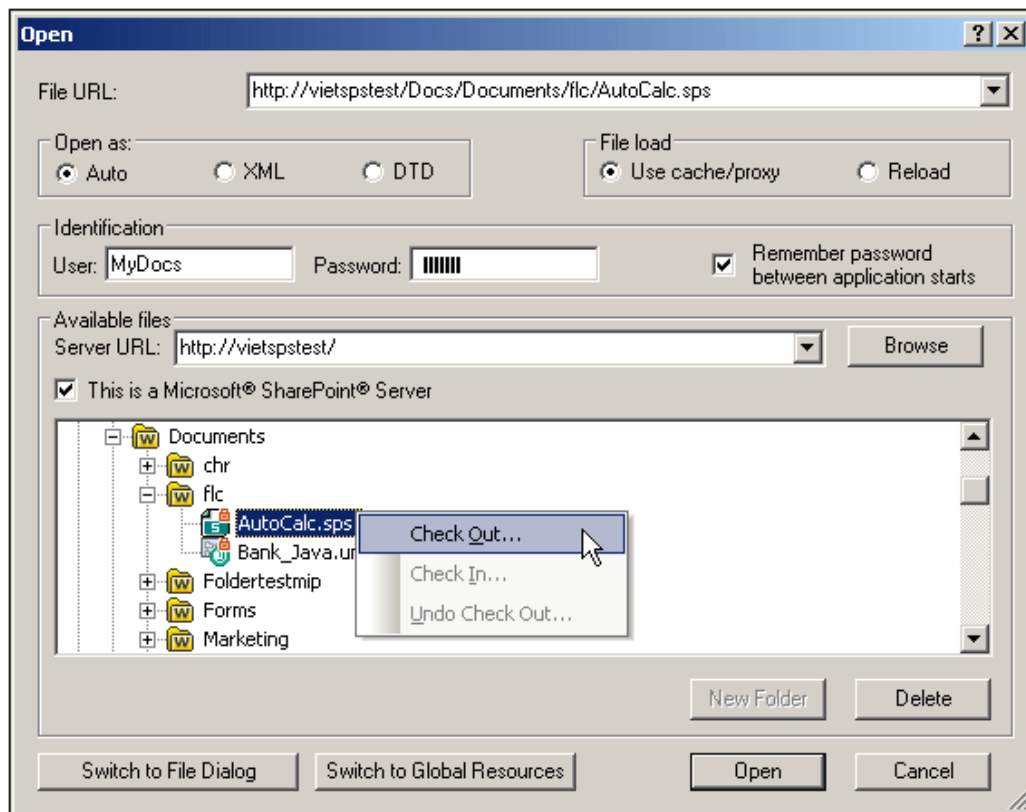
Anmerkung: Die Durchsuchen-Funktion steht nur auf Servern zur Verfügung, die WebDAV und Microsoft SharePoint Server unterstützen. Die unterstützten Protokolle sind FTP, HTTP und HTTPS.

Hinweis: Um mehr Kontrolle über den Ladevorgang zu haben, können Sie die Datei über den lokalen Cache oder einen Proxy-Server laden (wodurch der Vorgang erheblich schneller geht, wenn die Datei bereits zuvor einmal geladen wurde). Alternativ dazu können Sie die Datei auch neu laden, wenn Sie mit einem EPS-System oder einem Datenbank-System 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.

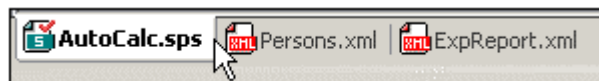


Wenn Sie mit der rechten Maustaste auf eine Datei klicken, erscheint ein Kontextmenü mit Befehlen, die für diese Datei zur Verfügung stehen (*Abbildung oben*).

- Im Folgenden sehen Sie die verschiedenen Dateisymbole:

	Eingecheckt. Die Datei kann ausgecheckt werden.
	Die Datei wurde von einem anderen Benutzer ausgecheckt und kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Die Datei 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 auf das Dateiregister im Hauptfenster Ihrer Applikation klicken (*Abbildung unten*), einchecken.



- Wenn eine Datei von einem anderen Benutzer ausgecheckt wurde, kann sie nicht ausgecheckt 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.

Öffnen und Speichern von Dateien über globale Ressourcen

Um eine Datei über eine globale Ressource zu öffnen oder zu speichern, klicken Sie auf **Zu globaler Ressource wechseln**. Daraufhin wird ein Dialogfeld angezeigt, in dem Sie die globale Ressource auswählen können. Diese Dialogfelder werden im Abschnitt Verwenden globaler Ressourcen näher beschrieben. Eine allgemeine Beschreibung globaler Ressourcen finden Sie in der Dokumentation im Abschnitt [Globale Ressourcen](#)

18.1.3 Neu laden

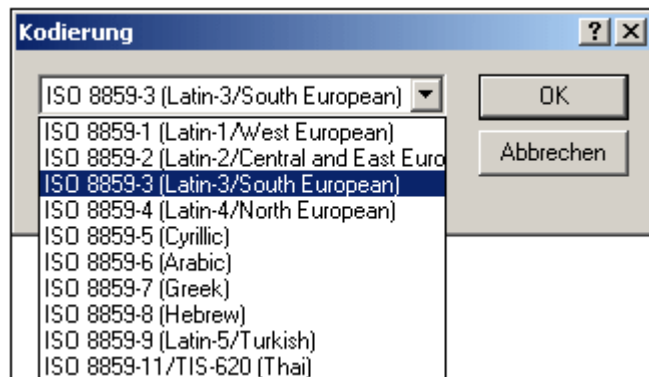


Mit **Neu laden** können Sie geöffnete Dokumente erneut laden.

Dieser Befehl ist nützlich, wenn ein geöffnetes Dokument außerhalb von XMLSpy geändert wurde. In diesem Fall 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. Sie können diese Option im Dialogfeld "Optionen" ([Extras | Optionen](#)) ändern.

18.1.4 Kodierung

Mit **Kodierung...** können Sie die Kodierung der in Verwendung befindlichen Datei (XML oder nicht XML) ansehen und 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 die Kodierungsspezifikation 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" auf dem Register "Kodierung"](#) festgelegt werden.

Anmerkung: Beim Speichern eines Dokumentes prüft XMLSpy automatisch die

Kodierungsspezifikation und öffnet ein Dialogfeld, falls es den vom Benutzer angegebenen Kodierungsnamen nicht erkennt. Sollte Ihr Dokument Zeichen enthalten, die in der gewählten Kodierung nicht angezeigt werden können, erhalten Sie eine Warnmeldung beim Speichern der Datei.

18.1.5 Schließen, Alle schließen, Alle inaktiven 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.

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.

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.

18.1.6 Speichern, Speichern unter, Alles speichern

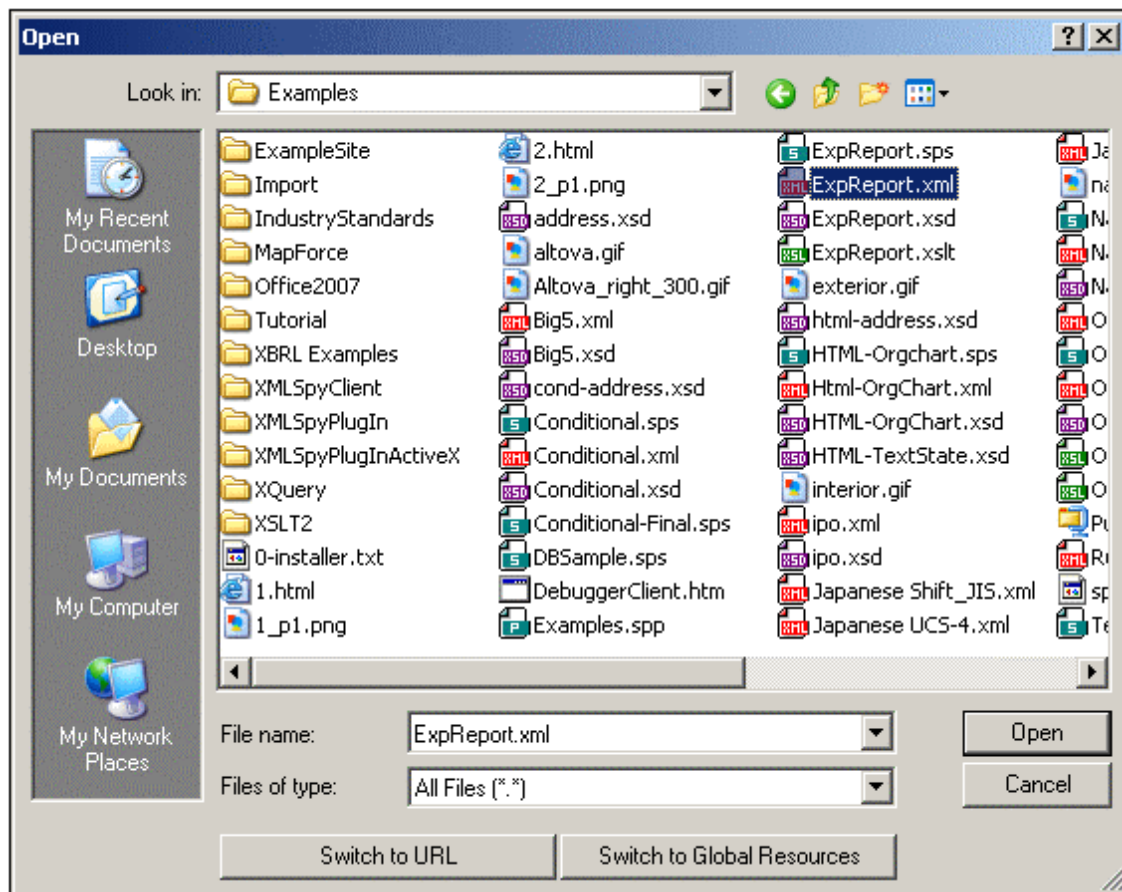
Mit dem Befehl **Speichern**  (**Strg+S**) wird der Inhalt des aktiven Dokuments in der geöffneten Datei gespeichert. Beim Speichern wird das Dokument automatisch auf [Wohlgeformtheit geprüft](#). Außerdem wird das Dokument auch automatisch validiert, wenn diese Option im Dialogfeld "Optionen" ([Extras | Optionen](#)) auf dem Register "Datei" aktiviert wurde. Die XML Deklaration wird auch auf [Kodierungsspezifikationen](#) geprüft. Diese Kodierung wird beim Speichern des Dokuments verwendet.

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**.

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 von Dateien über URLs und globale Ressourcen

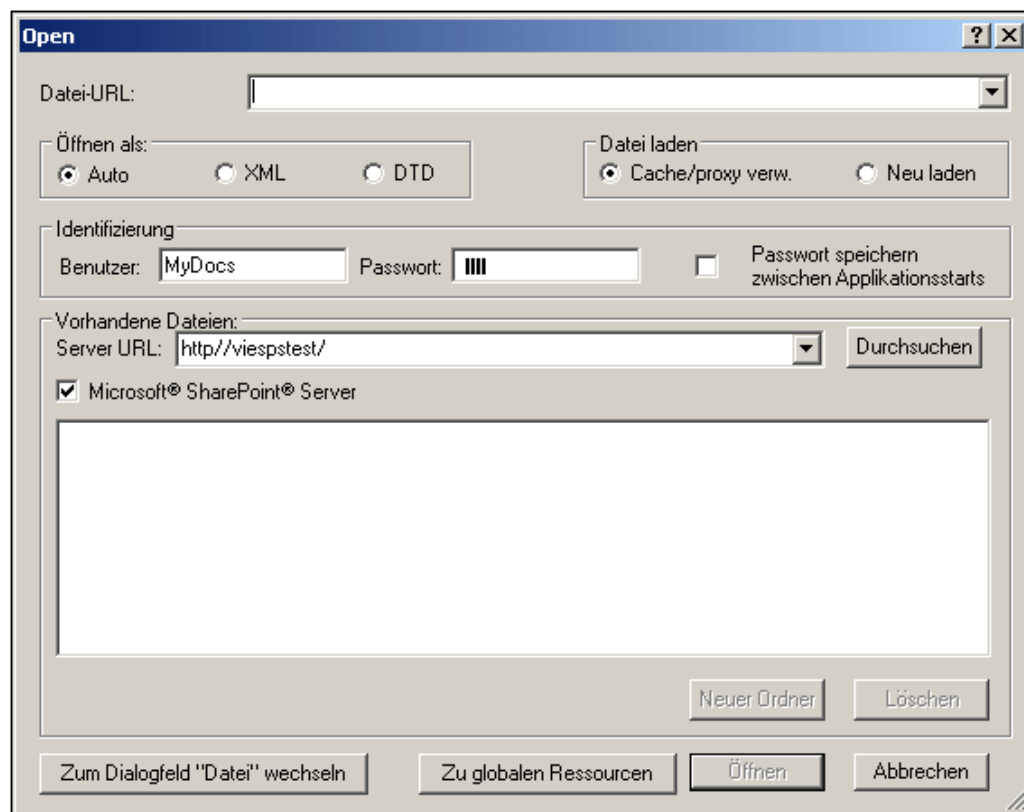
In einigen Dialogfeldern zum Öffnen und Speichern von Dateien können Sie auswählen, ob Sie die gewünschte Datei auswählen oder eine Datei über eine URL oder eine globale Ressource (siehe *Abbildung unten*) speichern möchten. Wählen Sie den Befehl **Zu URL wechseln** oder **Zu globaler Ressource wechseln** um eine dieser Optionen auszuwählen.



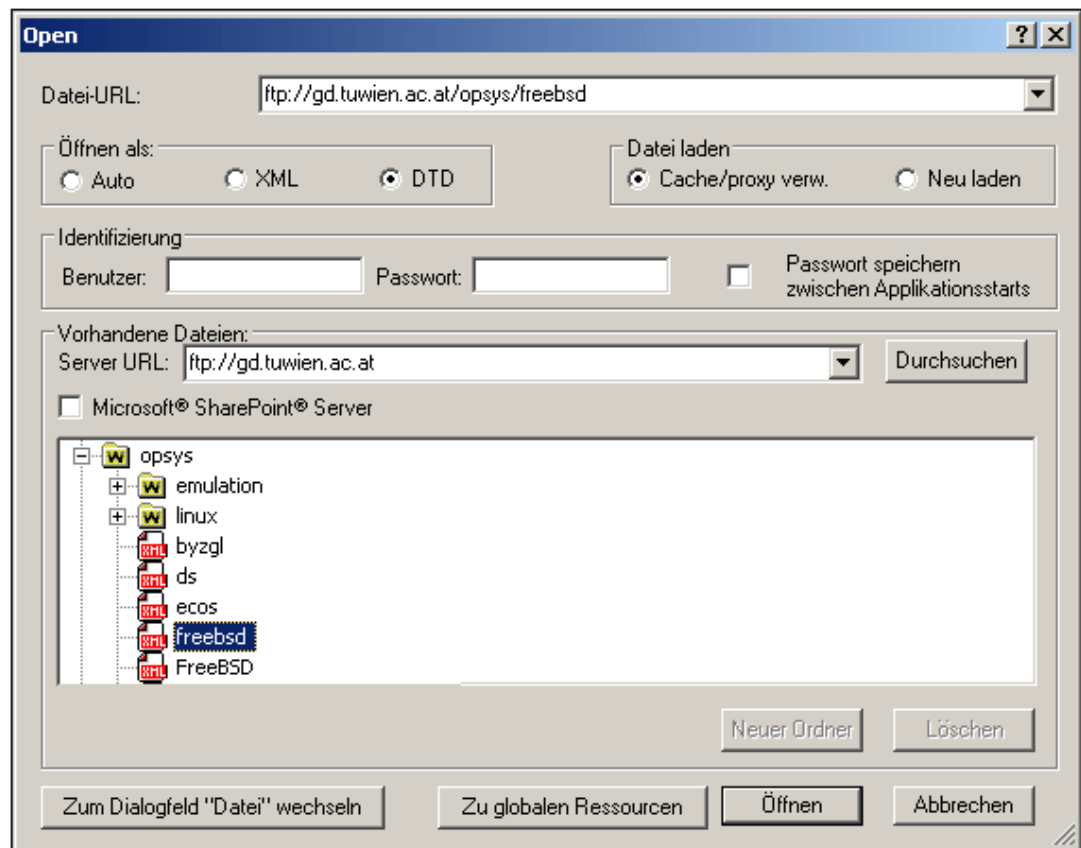
Auswählen von Dateien über URLs

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" in den URL-Modus (Abbildung unten).



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 anzuzeigen und darin zu navigieren.
5. Klicken Sie in der Ordnerstruktur auf die gewünschte Datei.



Die Datei-URL wird im Feld "Datei-URL" (Abbildung oben) angezeigt. Die Schaltfläche "Öffnen" wird erst jetzt aktiv.

6. Klicken Sie auf "Öffnen", um die Datei zu laden. Die Datei wird im Hauptfenster angezeigt.

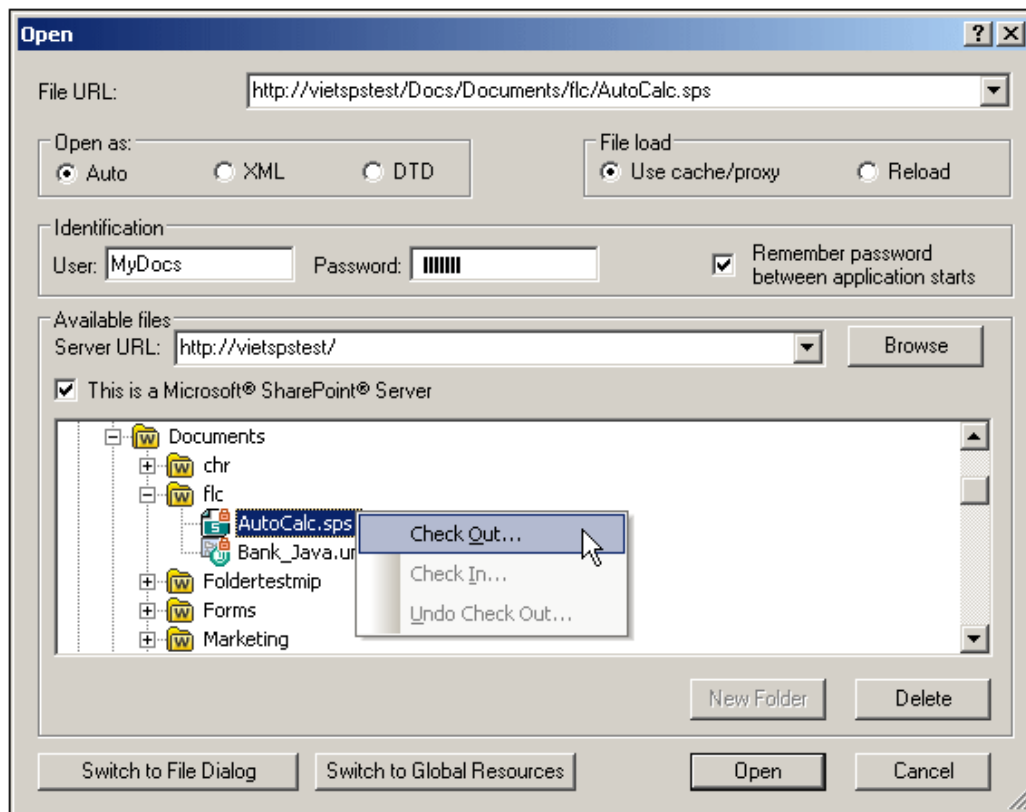
Anmerkung: Die Durchsuchen-Funktion steht nur auf Servern zur Verfügung, die WebDAV und Microsoft SharePoint Server unterstützen. Die unterstützten Protokolle sind FTP, HTTP und HTTPS.

Hinweis: Um mehr Kontrolle über den Ladevorgang zu haben, können Sie die Datei über den lokalen Cache oder einen Proxy-Server laden (wodurch der Vorgang erheblich schneller geht, wenn die Datei bereits zuvor einmal geladen wurde). Alternativ dazu können Sie die Datei auch neu laden, wenn Sie mit einem EPS-System oder einem Datenbank-System 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.

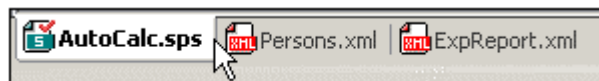


Wenn Sie mit der rechten Maustaste auf eine Datei klicken, erscheint ein Kontextmenü mit Befehlen, die für diese Datei zur Verfügung stehen (*Abbildung oben*).

- Im Folgenden sehen Sie die verschiedenen Dateisymbole:

	Eingecheckt. Die Datei kann ausgecheckt werden.
	Die Datei wurde von einem anderen Benutzer ausgecheckt und kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Die Datei 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 auf das Dateiregister im Hauptfenster Ihrer Applikation klicken (*Abbildung unten*), einchecken.



- Wenn eine Datei von einem anderen Benutzer ausgecheckt wurde, kann sie nicht ausgecheckt 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.

Öffnen und Speichern von Dateien über globale Ressourcen

Um eine Datei über eine globale Ressource zu öffnen oder zu speichern, klicken Sie auf **Zu globaler Ressource wechseln**. Daraufhin wird ein Dialogfeld angezeigt, in dem Sie die globale Ressource auswählen können. Diese Dialogfelder werden im Abschnitt Verwenden globaler Ressourcen näher beschrieben. Eine allgemeine Beschreibung globaler Ressourcen finden Sie in der Dokumentation im Abschnitt [Globale Ressourcen](#)

18.1.7 Als Mail senden



Mit dem Befehl **Als Mail senden...** können Sie jede XML-Datei oder Teile einer XML-Datei als Email versenden. Sie haben folgende Möglichkeiten:

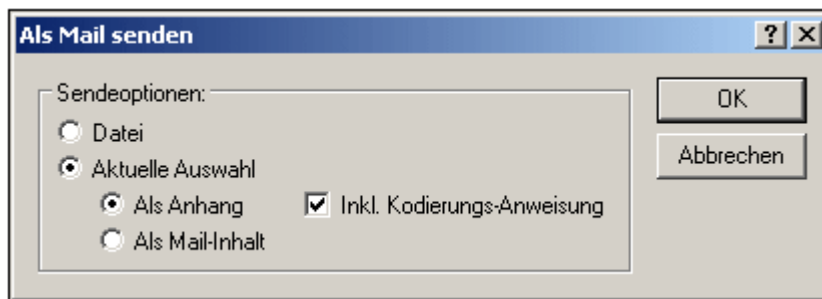
- Versenden eines XML-Dokuments als Anhang oder als Inhalt einer E-Mail.
- Versenden eines Teils eines XML-Dokuments als Anhang oder Inhalt einer E-Mail.
- Versenden einer Gruppe von Dateien (die im Projektfenster ausgewählt wurden) als Anhang einer E-Mail.
- Versenden einer URL (die im Projektfenster ausgewählt wurde) als Anhang oder als Link.

Bitte beachten Sie: Zur Verwendung dieser Funktion benötigen Sie ein MAPI-kompatibles E-Mail-System.

Versenden von Dokumenten und Teilen eines Dokuments

Um ein XML-Dokument zu versenden,

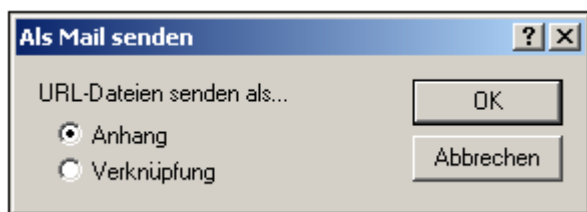
1. Machen Sie es im Hauptfenster zum aktiven Dokument. Wenn Sie nur einen Teil aus einem XML-Dokument oder eine Gruppe von Dateien eines Projekts versenden möchten, markieren Sie den Abschnitt im Dokument oder wählen Sie die gewünschten Dateien im Projekt aus.
2. Klicken Sie nun auf **Als Mail senden....** Daraufhin wird das folgende Dialogfenster angezeigt.



3. Aktivieren Sie die gewünschten Optionen und klicken Sie auf **OK**. Das/die ausgewählte(n) Dokument(e)/der ausgewählte Inhalt/die ausgewählten URLs werden an die E-Mail angehängt bzw. der Inhalt wird in die E-Mail eingefügt.

URL per Mail versenden

Zum Versenden einer oder mehrerer URLs wählen Sie die URLs im Projektfenster aus und klicken Sie auf **Als Mail senden....** Es wird das folgende Dialogfenster angezeigt.



Wählen Sie aus, in welcher Form die URL versendet werden soll und bestätigen Sie mit **OK**.

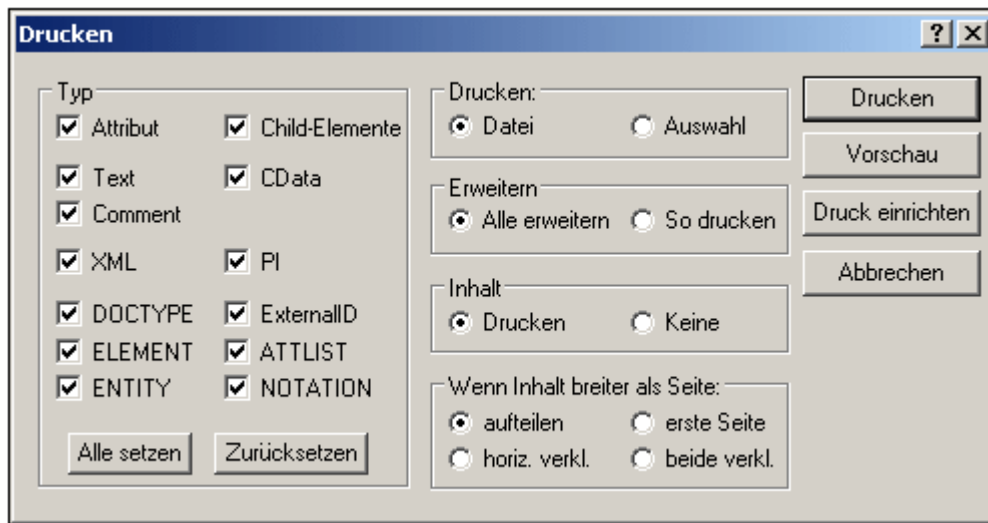
18.1.8 Drucken



Strg+P

Der Befehl **Drucken** öffnet das Dialogfeld "Drucken". Hier können Sie festlegen, was die Druckausgabe enthalten soll. Anschließend kann das aktive Dokument, das Sie in der aktuellen Ansicht sehen, gedruckt werden.

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 waagrecht). 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](#)".

Bitte beachten Sie: Sie können die Spaltenbreite in der Grid-Ansicht anpassen, um die Druckausgabe zu optimieren.

Programmlogo

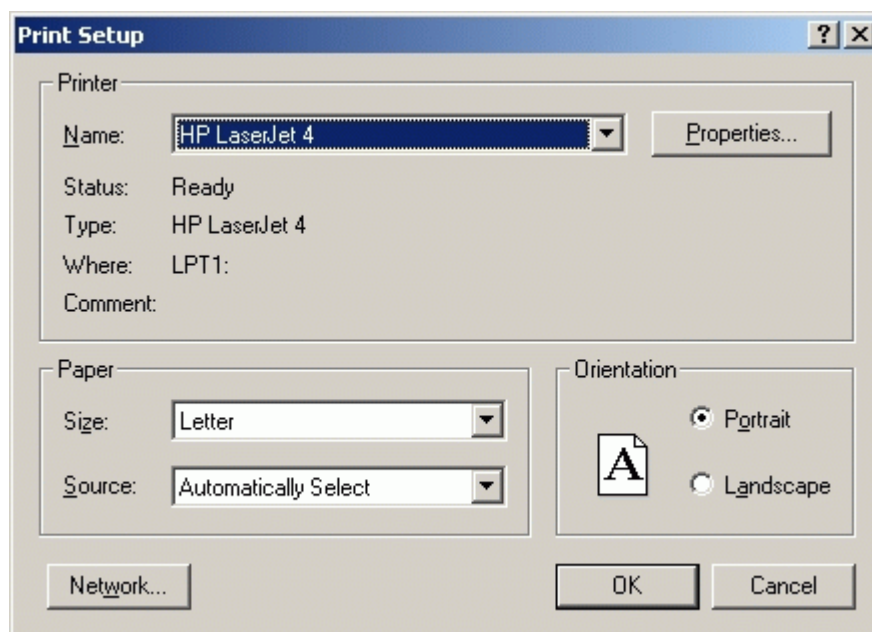
Wenn Sie eine Lizenz erworben haben, können Sie das Programmlogo, die Copyright-Informationen und die Registrierungsinformationen deaktivieren, wenn Sie ein Dokument von XMLSpy aus drucken. Diese Option steht im [Dialogfeld "Optionen" auf dem Register "Ansicht"](#) zur Verfügung.

18.1.9 Druckvorschau, Druckereinrichtung

Mit dem Befehl **Druckvorschau** öffnen Sie das Dialogfeld "Drucken". Klicken Sie auf die Schaltfläche "Vorschau", um eine Druckvorschau des aktiven Dokuments zu sehen. Im Modus "Druckvorschau" finden Sie in der Druckvorschau-Symbolleiste 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.

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.



Das obige Bild zeigt das Dialogfeld, bei dem ein Drucker des Typs HP Laserjet 4 am parallelen Port LPT1 angeschlossen ist.

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**.

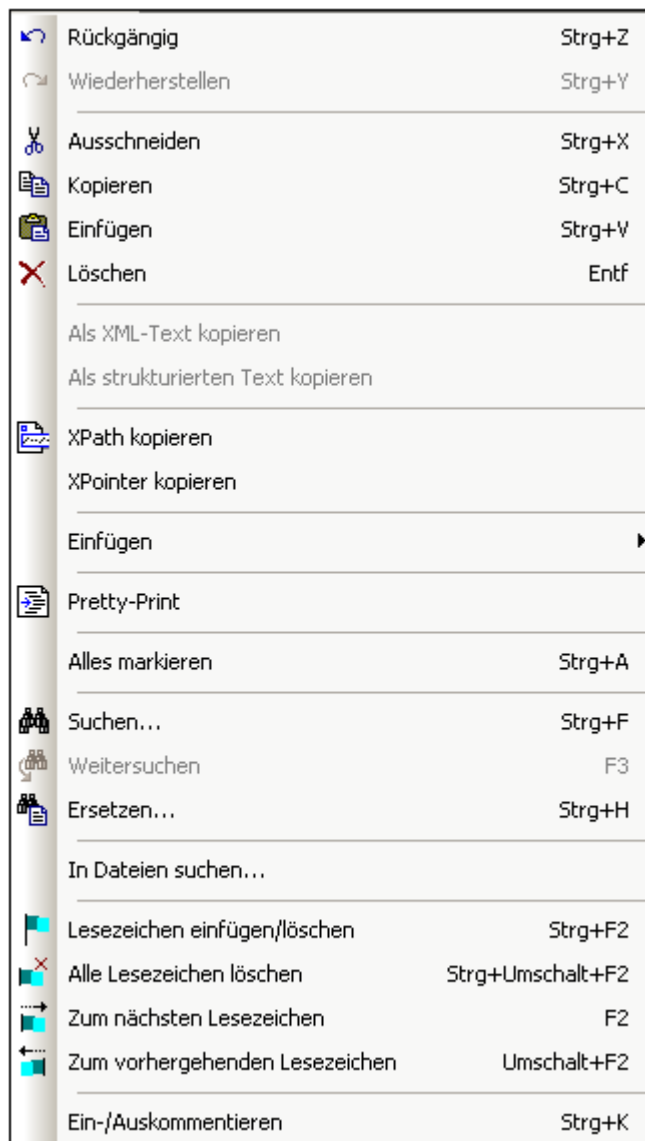
18.1.10 Zuletzt verwendete Dateien, Beenden

Im Menü **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+A** das Menü **Datei** aufrufen, und dann die der Datei entsprechende Nummer eintippen.

Mit dem Befehl **Beenden** beenden Sie XMLSpy, nachdem Sie Ihre Arbeit an Ihren Dokumenten abgeschlossen haben. 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.

18.2 Menü "Bearbeiten"

In diesem Menü befinden sich Befehle, die beim Editieren mit XMLSpy verwendet werden.



Zusätzlich zu den Standardbefehlen [Rückgängig](#), [Wiederherstellen](#), [Ausschneiden](#), [Kopieren](#), [Einfügen](#), [Löschen](#), [Alles markieren](#), [Suchen](#), [Weitersuchen](#) und [Ersetzen](#), bietet XMLSpy Sonderbefehle für folgende Funktionen:

- [Auswahl in Zwischenablage als XML-Text kopieren](#),
- [Als strukturierten Text kopieren](#), oder
- [XPath Selector für das ausgewählte Objekt in die Zwischenablage kopieren](#).
- Einfügen und Entfernen von Lesezeichen und Gehe zu Lesezeichen

18.2.1 Rückgängig, Wiederherstellen

Der Befehl **Rückgängig (Strg+Z)**  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 **Wiederherstellen (Strg+Y)**  (in Visual Studio **Wiederholen**) können Sie zuvor rückgängig gemachte Aktionen wiederherstellen. Sie können den Verlauf des Dokuments sowohl zurück als auch nach vorne verfolgen.

18.2.2 Ausschneiden, Kopieren, Einfügen, Löschen

Mit dem Befehl **Ausschneiden (Umschalt + Entf oder Strg + X)**  wird der markierte Text bzw. das markierte Objekt in die Zwischenablage kopiert und aus der aktuellen Position im Dokument gelöscht.

Mit dem Befehl **Kopieren (Strg+C)**  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 Befehle um Elemente in Textform in die Zwischenablage zu kopieren: [Als XML-Text kopieren](#) und [Als strukturierten Text kopieren](#). Auf dem [Register "Bearbeiten" unter Extras | Optionen](#) können Sie festlegen, welche dieser beiden Operationen bei Auswahl des Befehls "Kopieren" durchgeführt werden soll.

Mit dem Befehl **Einfügen (Strg+V)**  fügen Sie den Inhalt der Zwischenablage an der aktuellen Cursorposition ein.

Mit dem Befehl **Löschen (Entf)**  wird der markierte Text/das markierte Objekt gelöscht, ohne dass es in die Zwischenablage kopiert wird.

18.2.3 Als XML-Text kopieren

Mit dem Befehl **Als XML-Text kopieren** können Sie auf einfache Weise Daten aus anderen Produkten, die Datenmanipulation auf XML-Ebene erlauben, kopieren. Beim Editieren Ihres Dokuments in der Grid-Ansicht können Sie Elemente in ihrer XML-Textdarstellung in die Zwischenablage kopieren:

```
<row>
  <para align="left">
    <bold>Check the FAQ</bold>
  </para>
  <para>
    <link mode="internal">
      <link_section>support</link_section>
      <link_subsection>faq30</link_subsection>
      <link_text>XMLSPY 4.0 FAQ</link_text>
    </link>
    <link mode="internal">
      <link_section>support</link_section>
      <link_subsection>faq25</link_subsection>
      <link_text>XMLSPY 3.5 FAQ</link_text>
    </link>
  </para>
</row>
```

```

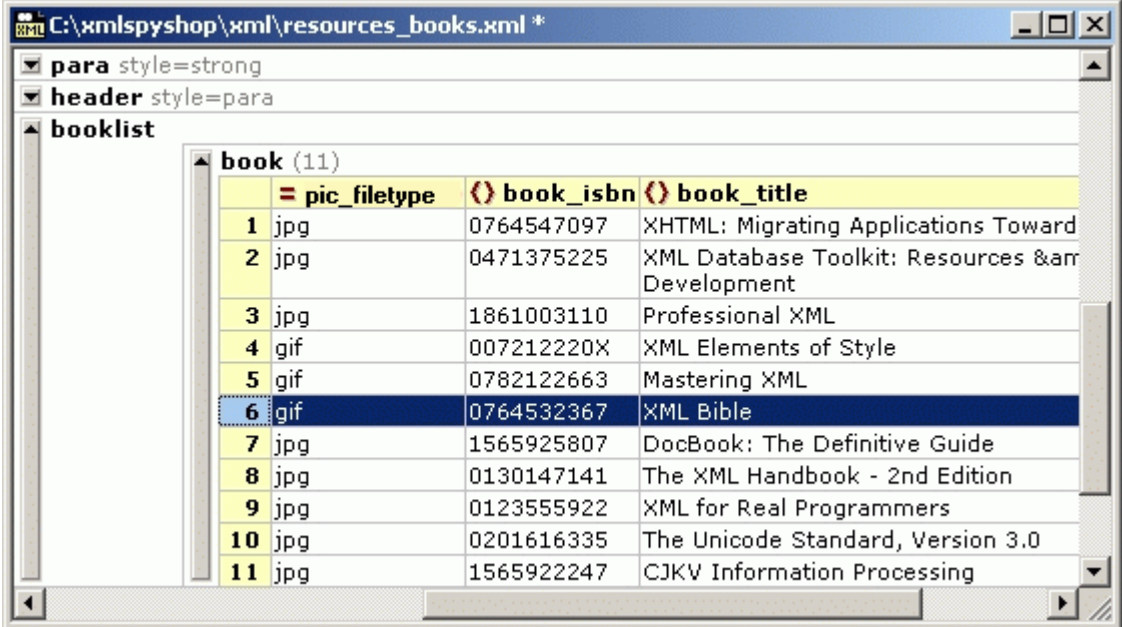
        </link>
    </para>
</row>

```

Der Befehl formatiert den Text automatisch nach den gültigen Einstellungen für das Speichern der Datei. Diese Einstellungen können im Dialogfeld "Optionen" (**Extras | Optionen**) auf dem [Register "Datei"](#) im Bereich "Datei speichern" vorgenommen werden.

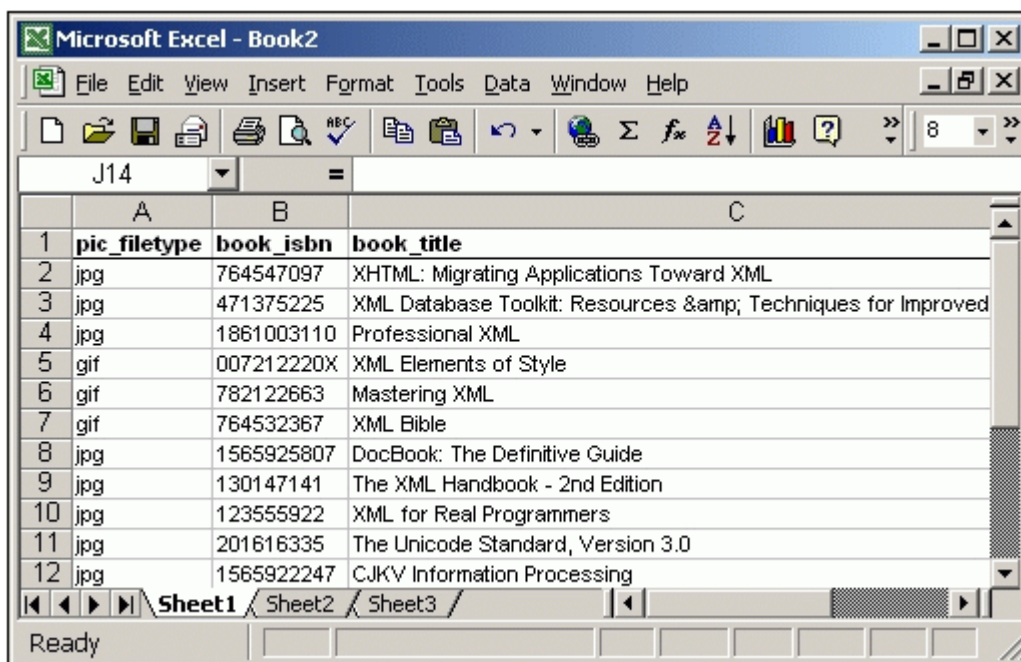
18.2.4 Als strukturierten Text kopieren

Mit diesem Befehl werden Elemente so in die Zwischenablage kopiert, wie sie am Bildschirm angezeigt werden. Dieser Befehl ist beim Kopieren von tabellenförmigen Daten in der Grid-Ansicht sehr nützlich. Die kopierten Daten können in XMLSpy sowie in Produkten von Drittanbietern verwendet werden. So können Sie z.B. XML-Daten in Spreadsheet-Applikationen wie Microsoft Excel kopieren.



	pic_filetype	book_isbn	book_title
1	jpg	0764547097	XHTML: Migrating Applications Toward
2	jpg	0471375225	XML Database Toolkit: Resources & Development
3	jpg	1861003110	Professional XML
4	gif	007212220X	XML Elements of Style
5	gif	0782122663	Mastering XML
6	gif	0764532367	XML Bible
7	jpg	1565925807	DocBook: The Definitive Guide
8	jpg	0130147141	The XML Handbook - 2nd Edition
9	jpg	0123555922	XML for Real Programmers
10	jpg	0201616335	The Unicode Standard, Version 3.0
11	jpg	1565922247	CJKV Information Processing

Wenn Sie diese Tabelle nach Excel kopieren, werden Ihre Daten wie folgt angezeigt:



Bitte beachten Sie: Das Ergebnis dieses Befehls hängt davon ab, wie die Daten am Bildschirm angezeigt werden. Wenn das sich das XML-Fragment, das als Beispiel für den Befehl [Als XML-Text kopieren](#) verwendet wurde, in der Tabellen-Ansicht der Grid-Ansicht befindet, sieht das Resultat des Kopiervorgangs aus wie folgt:

```
row
  para
    align    bold    link
    left     Check the FAQ
    link
      mode      link_section  link_subsection  link_text
      internal  support      faq30           XMLSPY 3.5 FAQ
      internal  support      faq25           XMLSPY 2.5 FAQ
```

In der normalen Grid-Ansicht werden die Daten wie folgt in die Zwischenablage kopiert:

row	
para	
align	left
bold	Check the FAQ
para	
link	
mode	internal
link_section	support
link_subsection	faq30
link_text	XMLSPY 3.5 FAQ
link	
mode	internal
link_section	support
link_subsection	faq25
link_text	XMLSPY 2.5 FAQ

18.2.5 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) auswählt und den Ausdruck in die Zwischenablage kopiert, sodass Sie ihn dann in ein Dokument (z.B. ein XSLT-Dokument) einfügen können. Alle Ausdrücke beginnen beim Dokument-Root.

Der XPath-Ausdruck wird in der Grid-Ansicht und der Textansicht unterschiedlich aufgelöst. Wenn in der Grid-Ansicht ein einzelnes Element markiert ist, werden mit dem XPath-Ausdruck nicht dieses Element, sondern alle Elemente dieses Namens auf dieser hierarchischen Ebene des Dokuments ausgewählt. In der Textansicht wird dieses spezifische Element ausgewählt. Wenn z.B. ein Element des dritten `Person`-Elements des zweiten `Company`-Elements ausgewählt wird, das den Namen `LastName` hat, würde der XPath-Ausdruck folgendermaßen lauten:

- Grid-Ansicht: `/Companies/Company/Person/LastName`
- Textansicht: `/Companies/Company[ 2 ]/Person[ 3 ]/LastName`

Anmerkung: In der Grid-Ansicht kann der Befehl **XPath kopieren** auch über das Kontextmenü aufgerufen werden.

18.2.6 XPointer kopieren

Der Befehl **XPointer kopieren** steht in der Text- und der Grid-Ansicht zur Verfügung. Er dient zum Erstellen eines `element()` Schema XPointer für den/die aktuell ausgewählten Node(s) und kopiert ihn in die Zwischenablage. Auf diese Art können Sie den XPointer in ein Dokument (z.B. in das `xpointer` Attribut eines `XInclude`-Elements in einem XML-Dokument) einfügen.

Das `element()` Schema von XPointer gibt die Ergebnisse in der Form `element( /1/3 )` zurück, womit das dritte Child des Dokument-Elements (oder Root-Elements) ausgewählt wird. Beachten Sie bitte die folgenden Punkte:

- Attribute können mit Hilfe des `element()` Schemas nicht dargestellt werden. Wenn ein Attribut ausgewählt wird, geschieht Folgendes: In der Grid-Ansicht ist der Befehl **XPointer kopieren** deaktiviert; in der Textansicht wird der XPointer des übergeordneten Elements dieses Attributs generiert.
- Es können nicht mehrere Elemente ausgewählt werden. Andernfalls ist der Befehl in der Grid-Ansicht deaktiviert. In der Textansicht wird der XPointer des übergeordneten

Elements der Auswahl generiert.

Anmerkung: In der Grid-Ansicht kann der Befehl **XPointer kopieren** auch über das Kontextmenü aufgerufen werden.

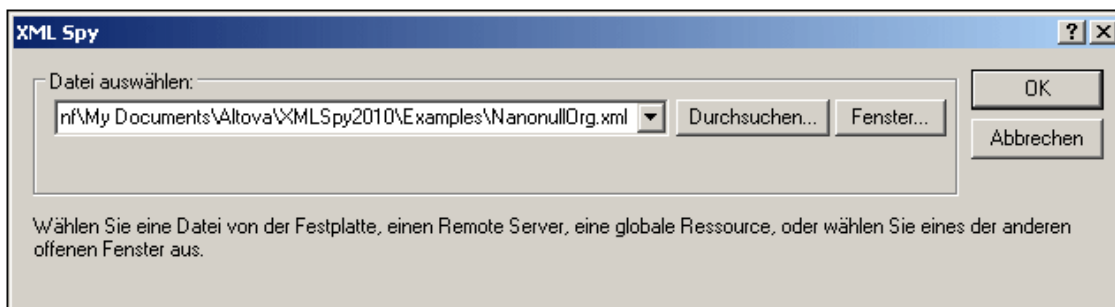
18.2.7 Einfügen

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](#)
- [XInclude einfügen](#)
- [Externe kodierte Datei einfügen](#)

Dateipfad einfügen

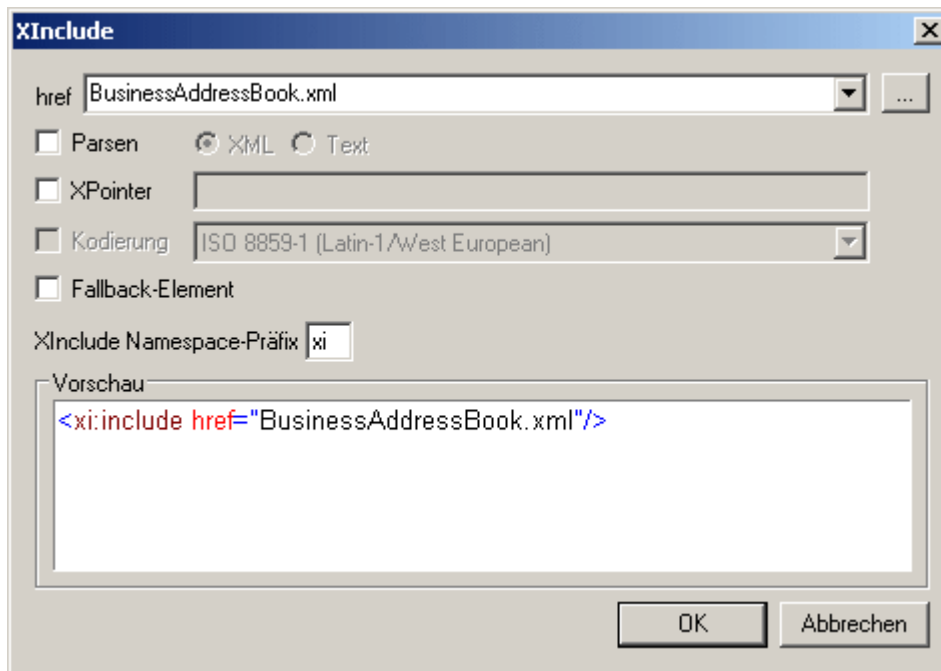
Der Befehl **Dateipfad einfügen** ist in der Text- und der Grid-Ansicht von Dokumenten jedes Dateityps aktiv. Sie können damit den Pfad zu einer Datei am Cursorpunkt einfügen. Wenn Sie auf den Befehl klicken, erscheint ein Dialogfeld (*Abbildung unten*), in dem Sie die gewünschte Datei auswählen können.



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**); (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*).



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 geparkt wird. (XML ist die Standardeinstellung und muss daher nicht extra angegeben werden.) Das Attribut `xpointer` identifiziert ein bestimmtes Fragment des Dokuments, das mit dem Attribut `href` lokalisiert wird; dieses Element wird inkludiert. Das Attribut `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-URLs 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 [Attribute mit dem Präfix :xml](#) des Abschnitts Schema-Ansicht.

XPointers

XMLSpy unterstützt XPointers in XInclude. Die relevanten W3C Recommendations dazu sind die [XPointer Framework](#) und die [XPointer element\(\) Scheme](#) 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.

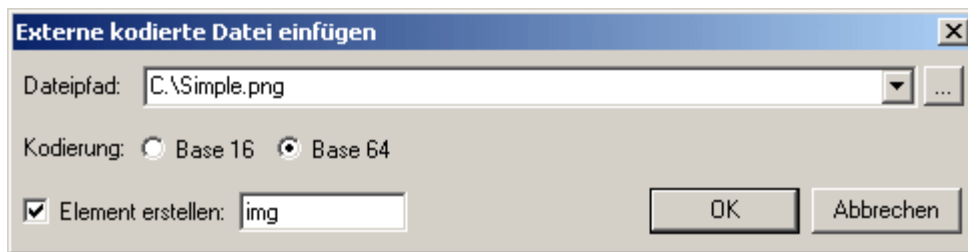
- Im ersten oben angeführten `xi:include` Element verwendet das `xpointer` Attribut das Elementschema mit dem NCName `usa`. Laut XPointer Framework identifiziert dieser NCName das Element, das die ID `usa` hat.
- Im zweiten oben angeführten `xi:include` Element identifiziert das `xpointer` Attribut mit 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.

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">
  iVBORw0KGgoAAAANSUhEUgAAABAAAAQMAAAAPW0iAAAAB1BMVEUAAAD/
  //+12Z/dAAAAM0lEQVR4nGP4/5/h/1+G/58ZDrAz3D/MCH8yw83NDDenge4U
  g9C9zwz3gVLMDA/A6P9/AFGGFyjoXZtQAAAAAE1FTksuQmCC
</img>
```

Die obige Auflistung zeigt den kodierten Text einer PNG-Bilddatei. Um den kodierten Text wurde ein `img` Element erstellt.

18.2.8 Pretty-Print



Mit diesem Befehl wird Ihr XML-Dokument in der Text-Ansicht neu formatiert. Zwei Formatierungsoptionen stehen zur Verfügung, abhängig davon, ob [im Dialogfeld "Optionen" auf dem Register "Ansicht" \(Extras | Optionen\)](#) das Kontrollkästchen *Einrückung verwenden* aktiviert ist oder nicht:

- *Einrückung verwenden* aktiviert: Das Dokument wird so neu formatiert, dass Sie eine strukturierte Ansicht erhalten. Jede untergeordnete Ebene in der Hierarchie wird um den definierten Einrückungsabstand weiter eingerückt. Dadurch erhalten Sie eine bessere Übersicht über die Dokumentstruktur.
- *Einrückung verwenden* deaktiviert: Das Dokument wird so neu formatiert, dass jede neue Zeile links ausgerichtet ist.

Um eine strukturierte eingerückte Ansicht des XML-Dokuments einzurichten, befolgen Sie die folgende Methode:

1. Aktivieren Sie im [Dialogfeld "Optionen" auf dem Register "Ansicht"](#) (**Extras | Optionen**) das Kontrollkästchen *Einrückung verwenden*.
2. Definieren Sie im [Dialogfeld "Einstellungen für Textansicht"](#) (**Ansicht | Einstellungen für Textansicht**) die Tabulatorgröße, die Sie für die Einrückung des mit Pretty-Print formatierten Texts verwenden möchten.
3. Geben Sie im [Dialogfeld "Optionen" auf dem Register "Datei"](#) (**Extras | Optionen**) die Elemente ein, für die keine Ausgabe-Formatierung (Einrückung) gewünscht wird.
4. Klicken Sie auf den Befehl **Pretty-Print** (diesen Befehl).

Um das Dokument so neu zu formatieren, dass alle Zeilen links ausgerichtet sind, deaktivieren Sie das Kontrollkästchen *Einrückung verwenden*.

Beachten Sie bitte die folgenden Punkte:

1. Damit dieser Befehl funktioniert, muss das XML-Dokument wohlgeformt sein.
2. Bei Anwendung von "Pretty-Print" werden beim Speichern des Dokuments Leerzeichen und Tabulatoren zum Dokument hinzugefügt.
3. Wenn die Option "Pretty Print" aktiviert ist (**Extras | Optionen | Ansicht | Einrückung verwenden**) und wenn Sie von der Textansicht in die Grid-Ansicht und wieder zurück in die Textansicht wechseln, so wird das Dokument automatisch mit Pretty Print dargestellt. Der Befehl **Pretty-Print** braucht dazu nicht ausgewählt zu werden.

18.2.9 Alles markieren

Strg + A

Mit dem Befehl **Alles markieren** (in Visual Studio **Alle auswählen**) wird der Inhalt des gesamten Dokuments markiert.

18.2.10 Suchen, Weitersuchen...

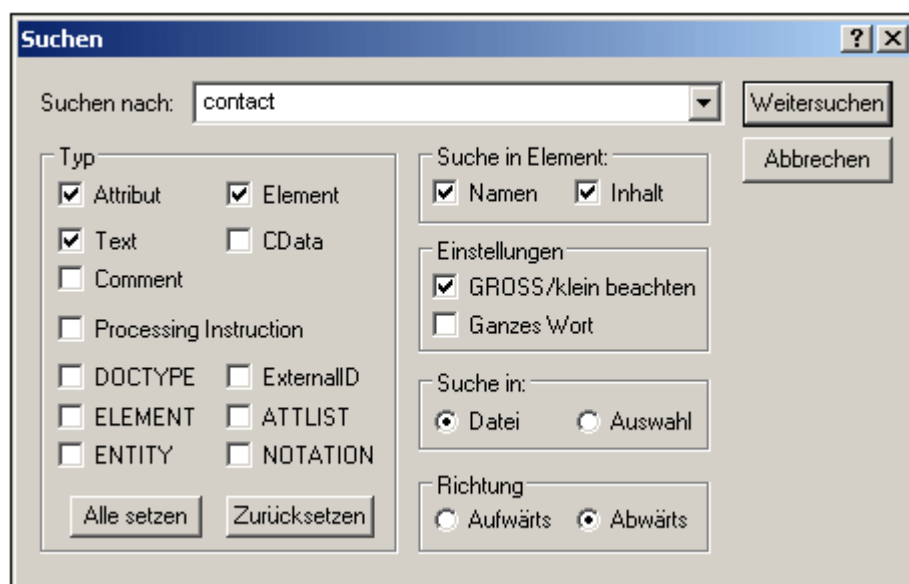
Mit dem Befehl **Suchen** (**Strg+F**)  (in Visual Studio **Suchen und Ersetzen**) rufen Sie das Dialogfeld "Suchen" auf, in dem Sie den zu suchenden String sowie weitere Suchoptionen spezifizieren können.

Abhängig von der ausgewählten Ansicht, gibt es im Dialogfeld "Suchen" verschiedene zur Auswahl stehende Optionen.

Um nach Text zu suchen, geben Sie diesen in das Listefeld "Suchen nach" ein oder wählen Sie einen der letzten 10 Einträge aus der Dropdown-Liste aus und legen Sie anschließend die Suchoptionen fest.

Grid-Ansicht

In der Grid-Ansicht erscheint das folgende Dialogfeld. Die verfügbaren Optionen sind unten beschrieben.

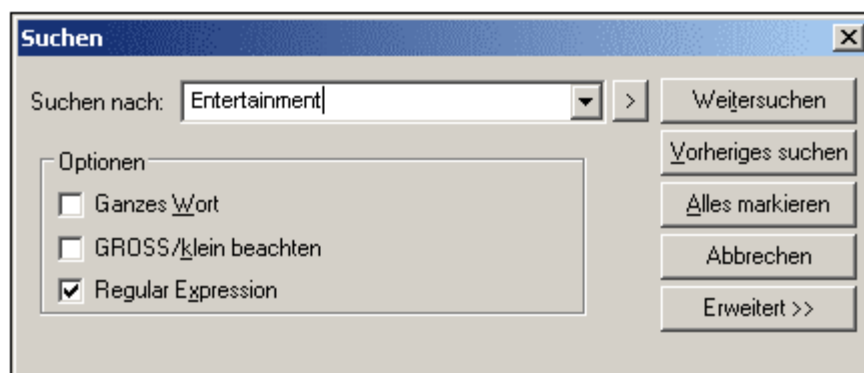


Aktivieren Sie die gewünschten Optionen oder klicken Sie auf ein Optionsfeld.

- 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. 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.
- Im Bereich "Richtung" definieren Sie, in welche Richtung die Suche erfolgen soll.

Text-Ansicht

In der Textansicht wird das folgende Dialogfeld angezeigt. Die verfügbaren Optionen sind im Folgenden beschrieben.



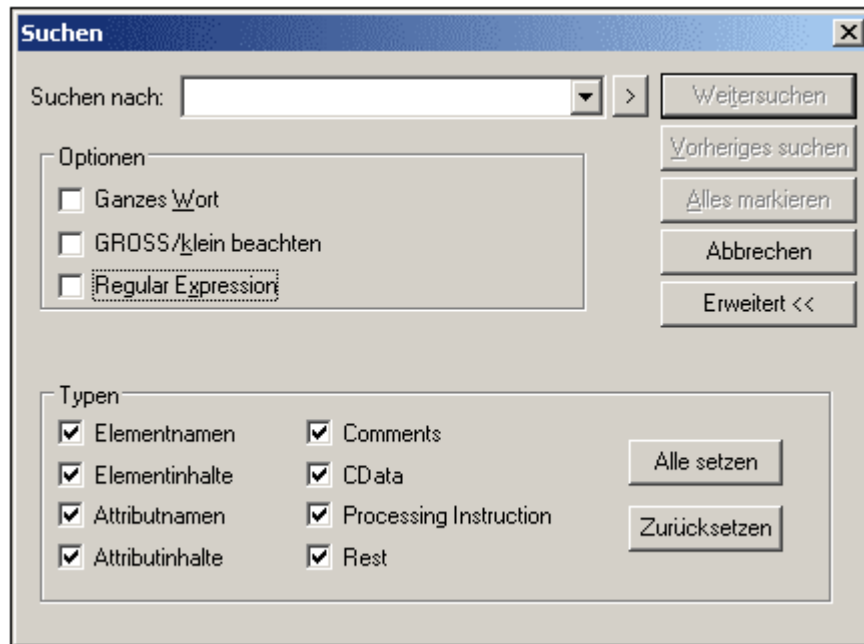
Die folgenden Suchoptionen stehen zur Verfügung:

- Ganzes Wort: Bei der Suche werden nur die Wörter im Text gefunden, die exakt mit dem gesuchten Text übereinstimmen. Bei Eingabe des Suchstrings `f` würde z.B. bei Aktivierung der Option "Ganzes Wort" nur das Wort `f` gefunden; `f` in `fess` würde

nicht gefunden werden.

- 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 Erläuterung zu Regular Expressions finden Sie unter [Regular Expressions](#).

Schaltfläche "Erweitert": Öffnet Gruppe **Typ** (siehe Screenshot unten), wo Sie angeben können, welche XML-Typen durchsucht werden sollen.

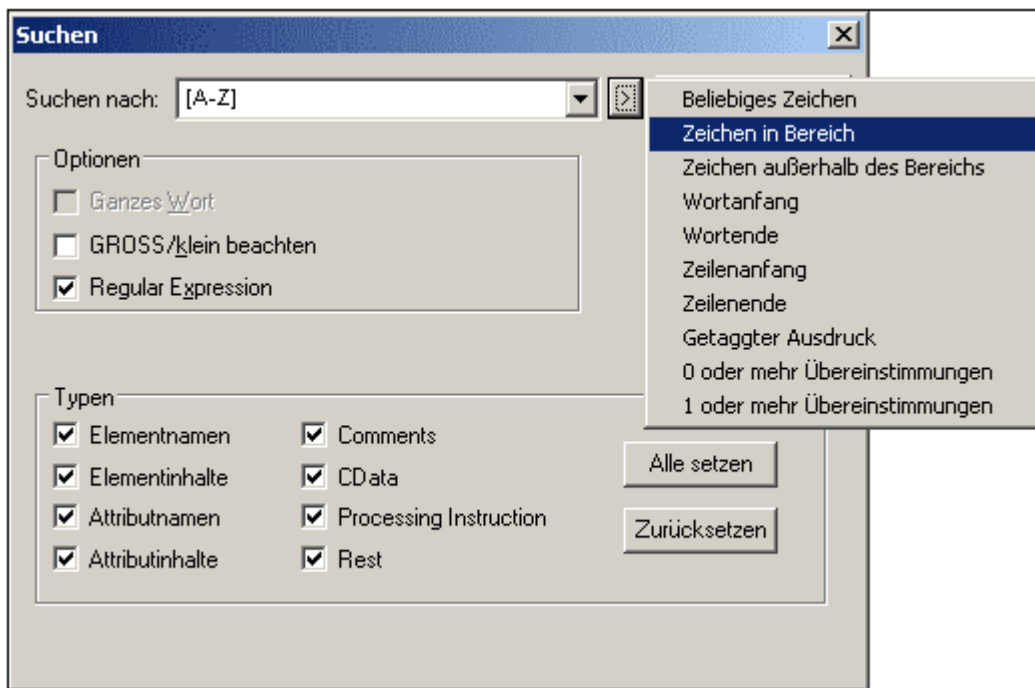


Bitte beachten Sie:

- 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](#) eingeben können.

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 >-Schaltfläche 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.

- .
 - \(
 - \)
 - \n
 - \<
 - \>
 - \x
 - [...]
 - [^...]
 - ^
 - \$
 - *
 - +
- 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 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.
- 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" nicht unterstützt.

Mit dem Befehl **Weitersuchen (F3)**  wird der letzte Suchbefehl wiederholt, um nach der nächsten Instanz des gesuchten Texts zu suchen.

Mit den Befehlen **Suchen** und **Weitersuchen** können Sie auch nach Datei- und Ordernamen suchen, wenn ein Projekt im Fenster "Projekt" ausgewählt ist.

18.2.11 Ersetzen...



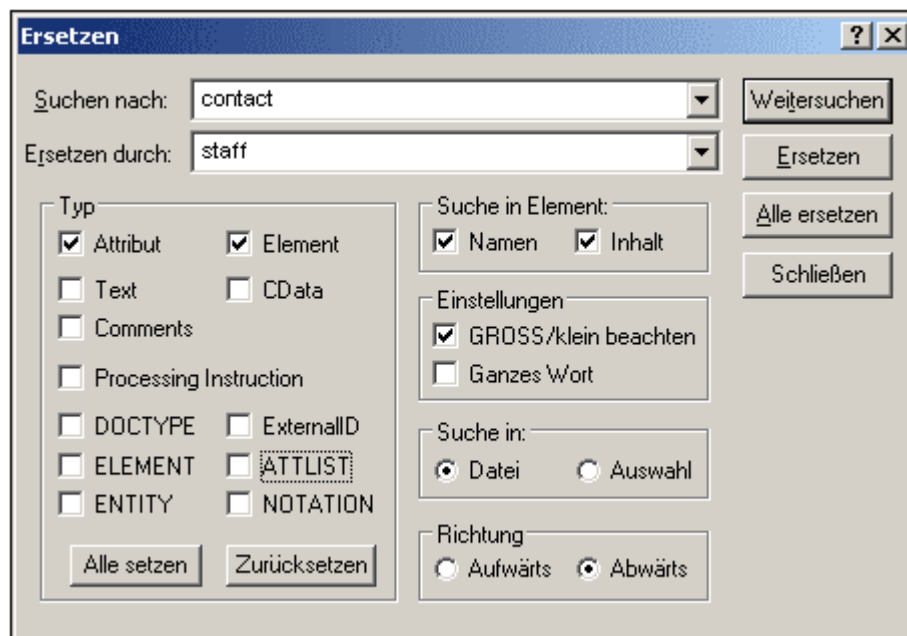
Strg + H

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...](#).

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

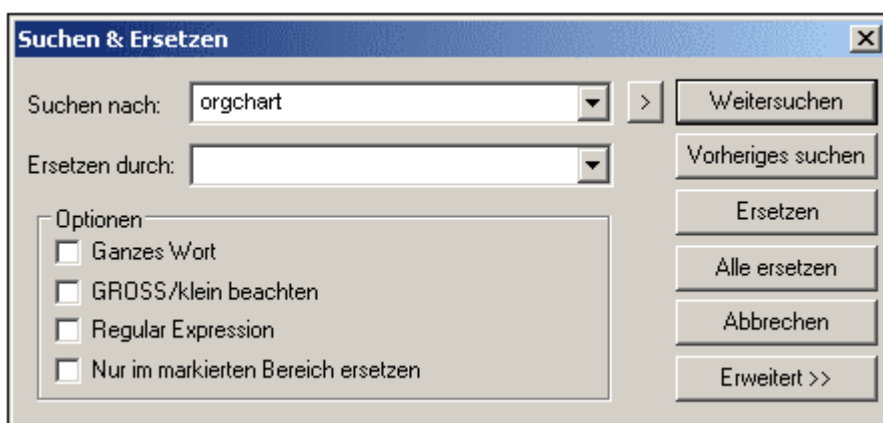
In der unten gezeigten Abbildung sehen Sie die verschiedenen Suchoptionen.



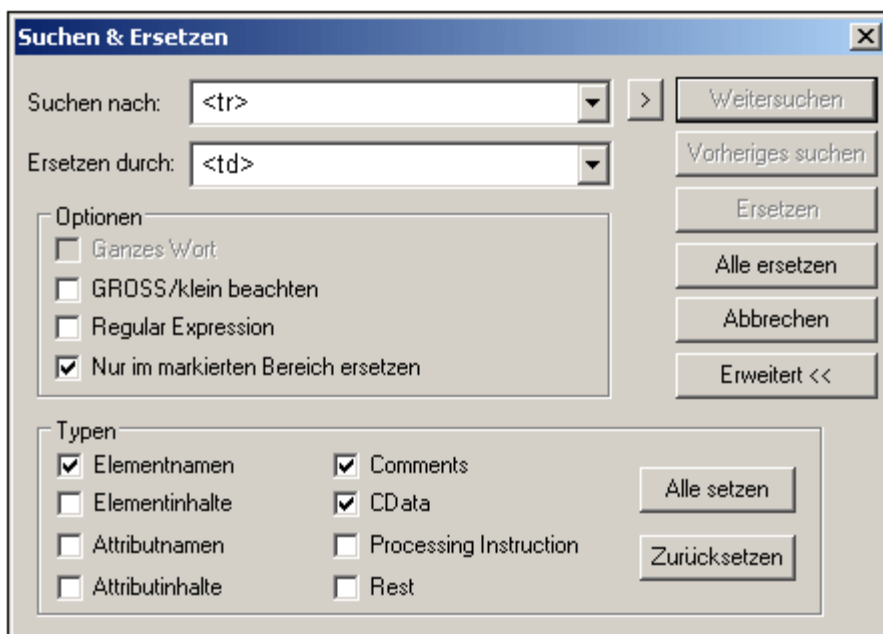
Diese Optionen sind im Abschnitt [Suchen...](#) beschrieben.

Text-Ansicht

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...](#). Bei Auswahl der Option "Nur im markierten Bereich ersetzen" werden die Ersetzungen nur im markierten Dokumentbereich vorgenommen.



Wenn Sie auf die Schaltfläche "Erweitert" klicken, werden die Typ-Optionen angezeigt (nähere Informationen dazu siehe [Suchen...](#)).



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.

18.2.12 In Dateien suchen

Das Dialogfeld **In Dateien suchen** ist eine ideale Methode, um 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 in der ["In Dateien suchen"-Ausgabeleiste](#) 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](#). 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 . Die Verwendung von Regular Expressions zur Suche wird unter [Suchen](#) erklärt.

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](#) 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 `.xml*`) gefiltert werden. Das Trennzeichen zwischen zwei Dateierweiterungen kann ein Komma oder ein Semikolon (`.xml*;` `.xml*` 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"-Ausgabeliste](#) 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"-Ausgabeliste](#) 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.

18.2.13 Lesezeichen-Befehle

Lesezeichen einfügen/löschen

Mit dem Befehl **Lesezeichen einfügen/löschen (Strg+F2)**  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 dem Befehl **Alle Lesezeichen löschen (Strg+Umschalt+F2)**  löschen Sie alle derzeit in der Textansicht definierten Lesezeichen. Dieser Befehl steht nur in der Textansicht zur Verfügung.

Bitte beachten Sie: Die Auswirkungen dieses Befehls können mit dem Befehl "Rückgängig" nicht rückgängig gemacht werden.

Zum nächsten Lesezeichen

Mit dem Befehl **Zum nächsten Lesezeichen (F2)**  setzen Sie den Cursor an den Beginn der nächsten mit einem Lesezeichen versehenen Zeile. Dieser Befehl ist nur in der Textansicht verfügbar.

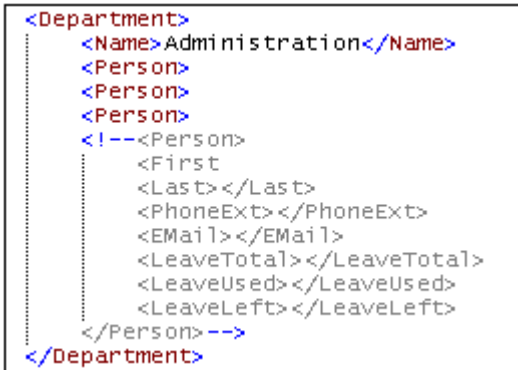
Zum vorhergehenden Lesezeichen

Mit dem Befehl **Zum vorhergehenden Lesezeichen (Umschalt+F2)**  setzen Sie den Cursor an den Beginn der vorhergehenden mit einem Lesezeichen versehenen Zeile. Dieser Befehl ist nur in der Textansicht verfügbar.

18.2.14 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 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*).

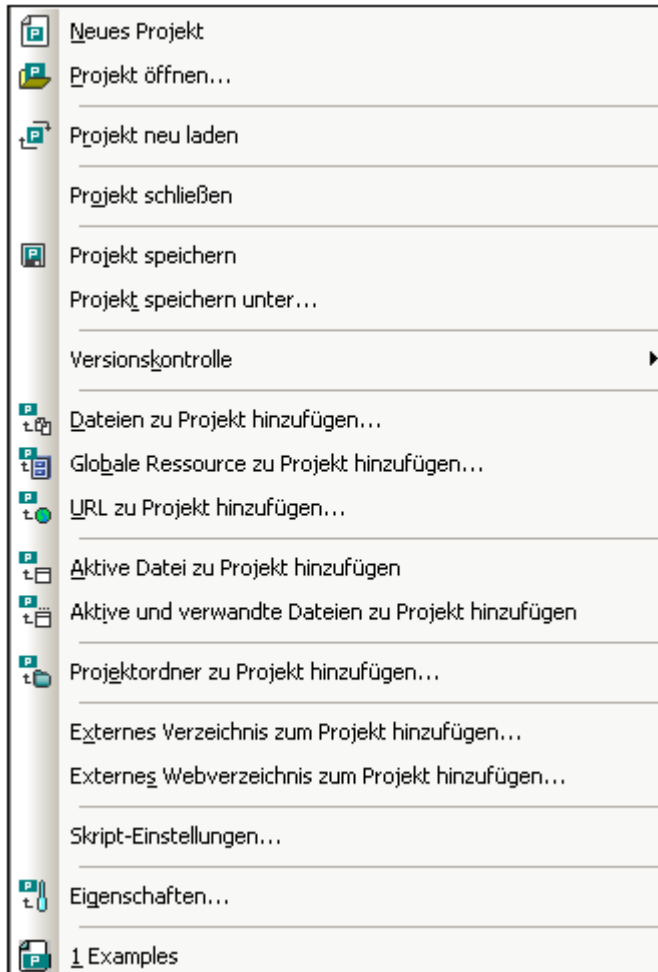


```
<Department>
  <Name>Administration</Name>
  <Person>
  <Person>
  <Person>
  <!--<Person>
    <First
    <Last></Last>
    <PhoneExt></PhoneExt>
    <EMail></EMail>
    <LeaveTotal></LeaveTotal>
    <LeaveUsed></LeaveUsed>
    <LeaveLeft></LeaveLeft>
  </Person>-->
</Department>
```

Um die Auskommentierung eines Textblocks rückgängig zu machen, wählen Sie die 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.

18.3 Menü "Projekt"

XMLSpy verwendet die bekannte Baumstruktur, um Dateien und URLs in einem XML-Projekt zu gliedern. [Dateien](#) und [URLs](#) mit gleicher Erweiterung können in [Ordern](#) 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
|   |
|   | -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.

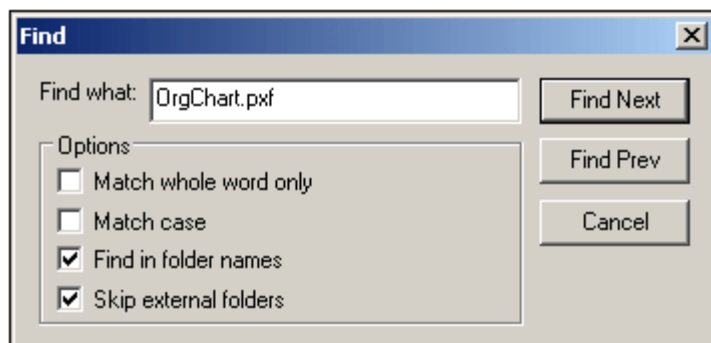
Drag-and-Drop

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. Ordernamen 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 Ordernamen übereinstimmen muss.
- Ordernamen können in die Suche einbezogen werden. Andernfalls werden nur Dateinamen durchsucht.
- [Externe Ordner](#) 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 innerhab 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 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.

18.3.1 Neues Projekt



Mit diesem Befehl 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.

18.3.2 Projekt öffnen.



Diese Funktion **öffnet** ein bereits vorhandenes Projekt inXMLSpy.
Falls Sie zuvor an einem anderen Projekt gearbeitet haben, wird dieses Projekt zuerst geschlossen.

18.3.3 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.

18.3.4 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.

18.3.5 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.

18.3.6 Versionskontrolle

XMLSpy unterstützt Microsoft Source-Safe und andere kompatible Speicherschnittstellen. Die von XMLSpy unterstützten Versionskontrollsysteme sind im Abschnitt [Unterstützte Versionskontrollsysteme](#) aufgelistet. Eine Beschreibung zur Installation finden Sie im Abschnitt [Installation von Versionskontrollsystemen](#). In diesem Abschnitt werden die Befehle aus dem Untermenü **Projekt | Versionskontrolle** beschrieben, mit denen Sie innerhalb von XMLSpy auf Ihr Versionskontrollsystem zugreifen können.

Übersicht über die Funktion "Versionskontrolle"

Um Dateien in einem XMLSpy-Projekt 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](#) 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 Untermenü](#) durchgeführt werden. Die Befehle in diesem Untermenü sind in der Benutzerreferenz im Abschnitt "Projektmenü" aufgelistet und beschrieben.

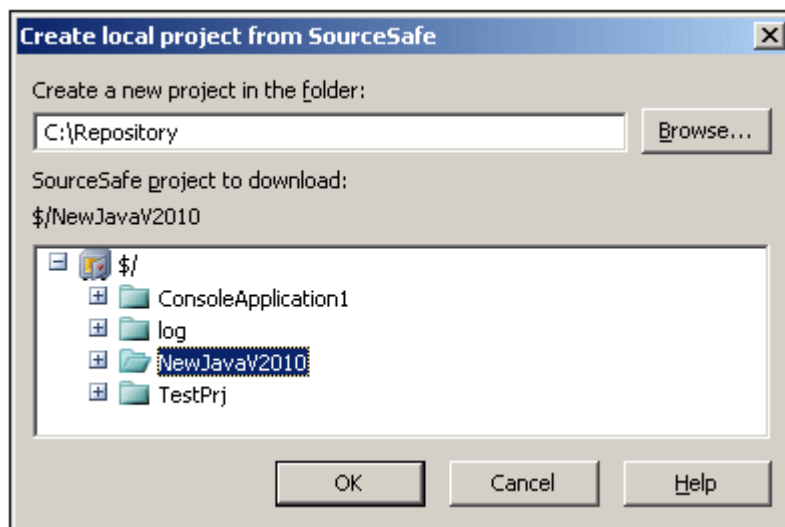
Anmerkung: Der aktuelle Versionskontroll-Provider kann auf zwei Arten gewechselt werden: (i) über die Versionskontrolloptionen ([Extras | Optionen | Versionskontrolle](#)) oder (ii) im Dialogfeld "Versionskontrolle wechseln" ([Projekt | Versionskontrolle | Versionskontrolle wechseln](#)).

Anmerkung: Beachten Sie, dass ein Versionskontrollprojekt nicht dasselbe wie ein XMLSpy-Projekt ist. Versionskontrollprojekte sind verzeichnisabhängig, während XMLSpy-Projekte logische Konstruktionen ohne direkte Verzeichnisabhängigkeit sind.

Aus Versionskontrolle öffnen

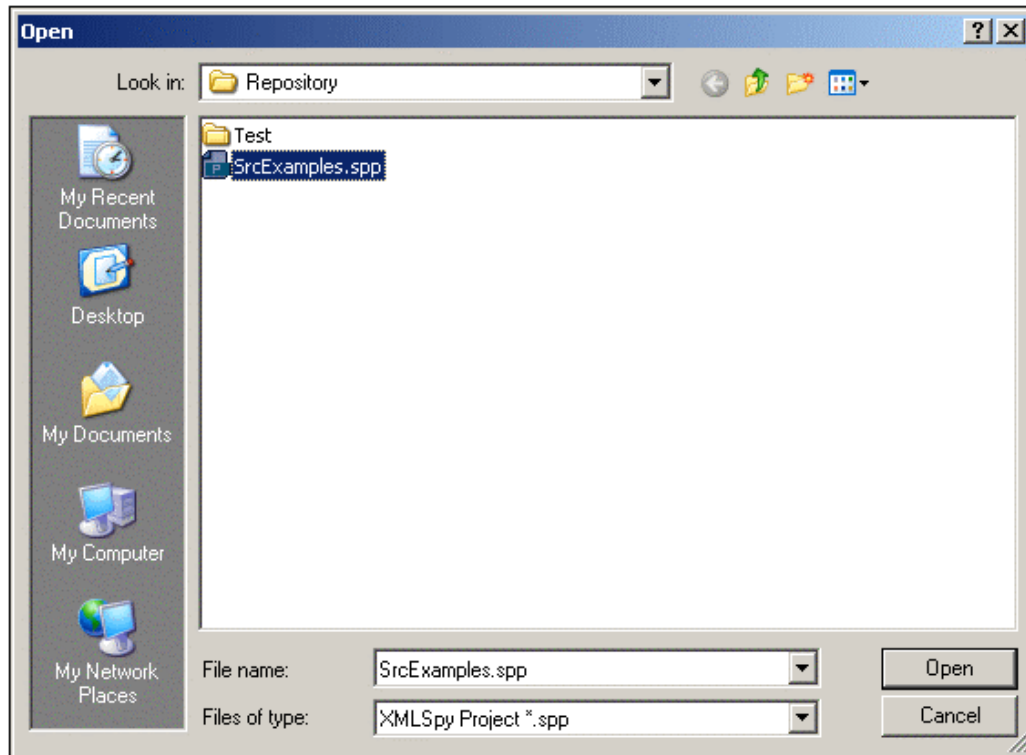
Mit dem Befehl "Aus Versionskontrolle öffnen" wird ein lokales Projekt anhand einer bestehenden Versionskontrolldatenbank erstellt. Dieser Befehl ist nur dann aktiv, wenn vorher über die Menüoption [Zu Versionskontrolle hinzufügen](#) ein XMLSpy Projekt zur Versionskontrolle hinzugefügt wurde.

1. Wählen Sie **Projekt | Versionskontrolle | Aus Versionskontrolle öffnen**. Daraufhin wird das Anmeldedialogfeld geöffnet, in dem Sie Ihre Anmeldedaten eingeben, bevor Sie fortfahren können.
2. Daraufhin wird das Dialogfeld "Create local project from SourceSafe" angezeigt (*Abbildung unten*). Definieren Sie das Verzeichnis, das das neue lokale Projekt enthalten soll. Dies wird zum neuen **Arbeitsverzeichnis** oder dem Auscheck-Ordner (in der Abbildung unten: C:\Repository)..



3. Wählen Sie das gewünschte Versionskontrollprojekt aus. In der Abbildung oben ist der Versionskontroll-Projektordner `NewJavaV2010` ausgewählt. Die Dateien in diesem Versionskontrollordner werden in den lokalen Ordner heruntergeladen. Wenn der hier von Ihnen definierte Ordner noch nicht existiert, erscheint ein Dialogfeld, in dem Sie aufgefordert werden, ihn zu erstellen. Klicken Sie auf **Ja**, um den neuen Ordner zu

- erstellen. Das Dialogfeld "Öffnen" wird angezeigt. (Abbildung unten).
 4. Daraufhin wird das Dialogfeld "Öffnen" (Abbildung unten) angezeigt.



4. Wählen Sie die gewünschte Applikationsprojektdatei aus und klicken Sie auf "Öffnen". Das ausgewählte Projekt wird nun in XMLSpy geöffnet und die Datei wird unter Versionskontrolle gestellt.

Symbole der Versionskontrolle:

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.

Versionskontrolle aktivieren

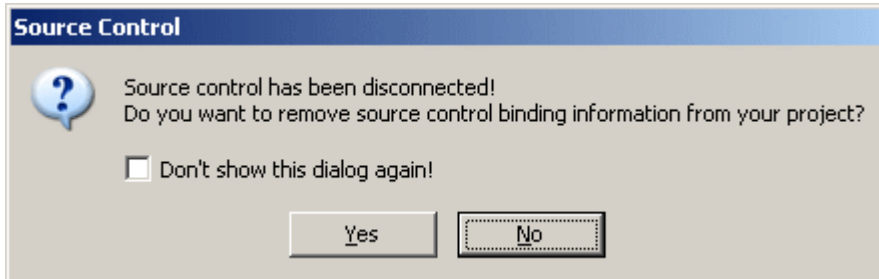
Mit Hilfe dieses Befehls können Sie die Versionskontrolle für ein XMLSpy-Projekt aktivieren oder deaktivieren. Bei Auswahl dieser Option für eine Datei oder einen Ordner, wird die Versionskontrolle für das gesamte Projekt aktiviert/deaktiviert.

So aktivieren Sie die Versionskontrolle für ein Projekt:

1. Klicken Sie auf jede beliebige Datei oder jeden beliebigen Ordner im Projektfenster.
2. Wählen Sie die Menüoption **Projekt | Versionkontrolle | Versionskontrolle aktivieren**. Der vorherige Ein-/Auscheck-Status der verschiedenen Dateien wird abgerufen und im Projektfenster angezeigt.

So deaktivieren Sie die Versionskontrolle für ein Projekt:

Wählen Sie die Menüoption **Projekt | Versionskontrolle | Versionskontrolle aktivieren**. Sie werden nun gefragt, ob Sie die Binding-Informationen aus dem Projekt entfernen möchten. (siehe Abbildung unten).



Um die Versionskontrolle **provisorisch** zu deaktivieren, wählen Sie **Nein**.

Um die Versionskontrolle für dieses Projekt **permanent** zu deaktivieren, wählen Sie **Ja**.

Aktuellste Version holen

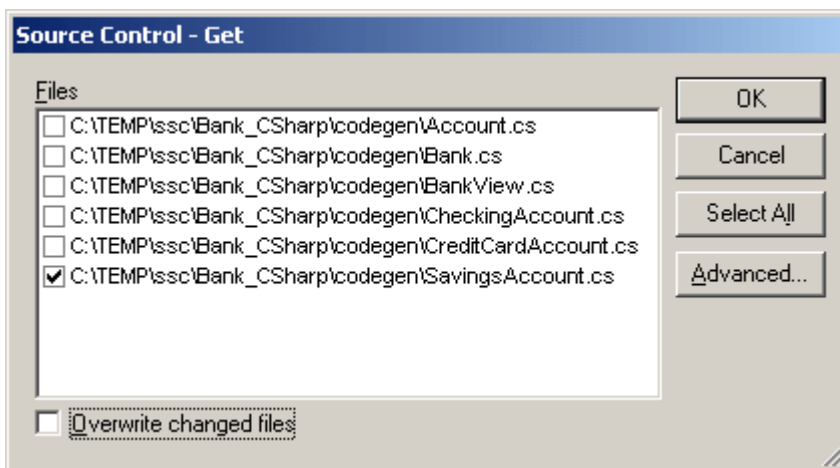
Mit diesem Befehl wird die aktuellste Version der ausgewählten Datei(en) aus dem Versionskontrollspeicher in das Arbeitsverzeichnis geholt. Die Dateien werden als schreibgeschützte Dateien und nicht ausgecheckt abgerufen. Bei Verwendung dieses Befehls stehen keine weiteren Optionen zur Verfügung.

So rufen Sie die neueste Version einer Datei ab:

1. Wählen Sie die Dateien aus, für die Sie die neueste Version holen möchten.
2. Wählen Sie **Projekt | Versionskontrolle | Aktuellste Version holen**.

Abrufen

Ruft eine schreibgeschützte Kopie der ausgewählten Dateien ab und platziert Sie in den Arbeitsordner. Standardmäßig sind die Dateien nicht zur Bearbeitung ausgecheckt. Um eine Datei abzurufen, wählen Sie sie im Projektfenster aus (es können auch mehrere Dateien ausgewählt werden) und wählen Sie dann den Befehl **Abrufen**. Daraufhin wird das Dialogfeld "Abrufen" (Abbildung unten) geöffnet. Versehen Sie die abzurufenden Dateien mit einem Häkchen.

**Kontrollkästchen "Geänderte Dateien überschreiben"**

Überschreibt lokal geänderte Dateien mit jenen aus der Versionskontrolldatenbank.

Alle auswählen

Wählt alle Dateien im Listenfeld aus.

Erweitert

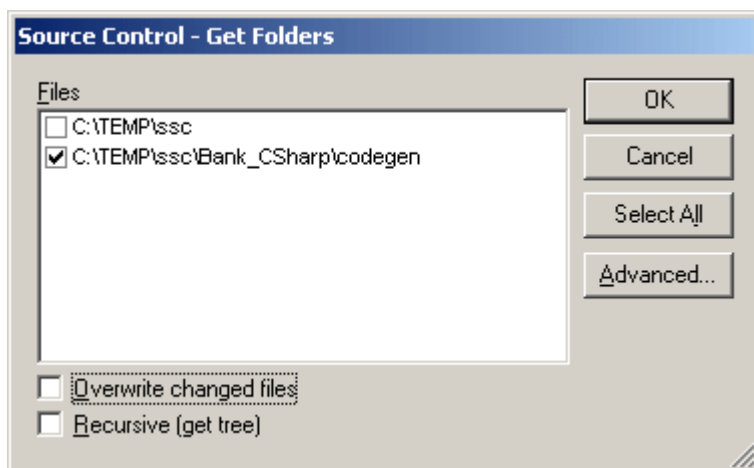
Mit Hilfe dieser Option können Sie die Optionen **Replace writable** und **Set timestamp** in den entsprechenden Auswahllisten auswählen.



Mit dem Kontrollkästchen "Make writable" wird das "read-only" Attribut der abgerufenen Dateien entfernt.

Ordner abrufen

Ruft schreibgeschützte Kopien von Dateien in den ausgewählten Ordnern ab und platziert Sie in den Arbeitsordner. Die Dateien werden standardmäßig nicht zur Bearbeitung ausgecheckt. Um einen Ordner abzurufen, wählen Sie ihn im Projektfenster aus und wählen Sie dann den Befehl **Ordner abrufen**. Daraufhin wird das Dialogfeld "Abrufen" (*Abbildung unten*) geöffnet. Versehen Sie die abzurufenden Ordner mit einem Häkchen.



Kontrollkästchen "Geänderte Dateien überschreiben"

Überschreibt lokal geänderte Dateien mit jenen aus der Versionskontrolldatenbank.

Kontrollkästchen Rekursiv (Struktur abrufen)

Ruft alle Dateien der Ordnerstruktur unterhalb des ausgewählten Ordners ab.

Alle auswählen

Wählt alle Dateien im Listenfeld aus.

Erweitert

Mit Hilfe dieser Option können Sie die Optionen **Replace writable** und **Set timestamp** in den entsprechenden Auswahllisten auswählen.



Mit dem Kontrollkästchen "Make writable" wird das "read-only" Attribut der abgerufenen Dateien entfernt.

Auschecken

Mit diesem Befehl checken Sie die aktuellste Version der ausgewählten Datei(en) aus und platzieren beschreibbare Kopien davon in das Arbeitsverzeichnis. Die Dateien werden für andere Benutzer als "ausgecheckt" markiert.

Um Dateien auszuchecken, gehen Sie folgendermaßen vor:

1. Wählen Sie die gewünschte Datei bzw. den gewünschten Ordner in der Modell-Struktur aus.
2. Wählen Sie **Projekt | Versionskontrolle | Auschecken**.
3. Wählen Sie im Dialogfeld "Auschecken" die gewünschten Dateien aus und klicken Sie auf **OK**.

Anmerkung:

- Sie können die Anzahl der auszucheckenden Dateien ändern, indem Sie die einzelnen Kontrollkästchen in der Liste der Dateien aktivieren.
- Bei Auswahl der Option *Lokale Version auschecken* wird nur die lokale Version der Dateien ausgecheckt, nicht die aus der Versionskontrolldatenbank.
- Folgende Objekte können ausgecheckt werden: (i) Einzelne Dateien - (klicken Sie im Projektfenster auf die Datei (oder mehrere Dateien mittels Strg + Klick); (ii) Ordner (klicken Sie im Projektfenster auf den Ordner, um ihn auszuwählen oder verwenden Sie Strg+Klick um mehrere Ordner zu selektieren)
- Ausgecheckte Dateien und Ordner werden mit einem roten Häkchen markiert.
-

Erweitert

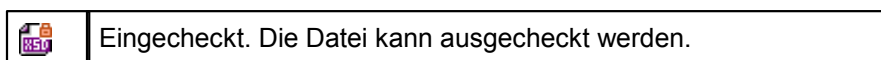
Mit Hilfe dieser Option können Sie die Optionen **Replace writable** und **Set timestamp** in den entsprechenden Auswahllisten auswählen.



Mit dem Kontrollkästchen "Make writable" wird das "read-only" Attribut der abgerufenen Dateien entfernt.

Symbole der Versionskontrolle:

Die Ordner und Dateien sind mit bestimmten Symbolen versehen, deren Bedeutung im Folgenden erklärt wird.



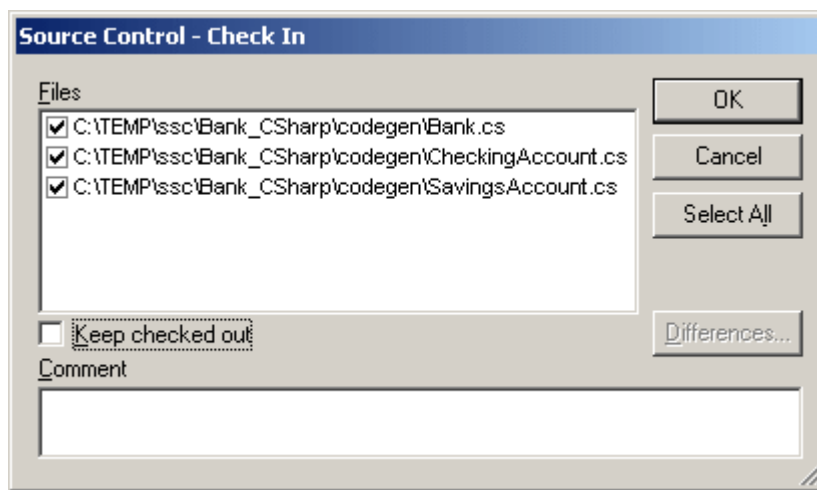
	Die Datei wurde von einem anderen Benutzer ausgecheckt und kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Die Datei kann bearbeitet und eingchecked werden.

Einchecken

Mit diesem Befehl checken Sie zuvor ausgecheckte Dateien, d.h. Ihre lokal aktualisierten Dateien, wieder ein und stellen die Dateien zurück in das Versionskontrollprojekt.

Um Dateien einzuchecken, gehen Sie folgendermaßen vor:

1. Wählen Sie die gewünschten Dateien in der Modell-Struktur aus.
2. Wählen Sie **Projekt | Versionskontrolle | Einchecken**.
3. Wählen Sie im Dialogfeld "Einchecken" die gewünschten Dateien aus und klicken Sie auf OK.



Anmerkung:

- Rechtsklicken Sie im Projektfenster auf ein ausgechecktes Objekt, und wählen Sie "Einchecken" im Kontextmenü.
- Folgende Objekte können eingchecked werden: (i) Einzelne Dateien - (klicken Sie im Projektfenster auf die Datei (oder mehrere Dateien mittels Strg + Klick); (ii) Ordner (klicken Sie im Projektfenster auf den Ordner, um ihn auszuwählen oder verwenden Sie Strg+Klick um mehrere Ordner zu selektieren)
- Das Vorhängeschloss-Symbol zeigt an, dass sich die Datei/der Ordner **unter Versionskontrolle** befindet, aber derzeit nicht ausgecheckt ist.
- Die Schaltfläche "Unterschiede" ist aktiv, wenn im Bereich "Dateien" eine Zeile ausgewählt ist. Wenn Sie auf diese Schaltfläche klicken, sehen Sie die Unterschiede zwischen den beiden Dateiversionen.

Symbole der Versionskontrolle:

Die Ordner und Dateien sind mit bestimmten Symbolen versehen, deren Bedeutung im Folgenden erklärt wird.

	Eingchecked. Die Datei kann ausgecheckt werden.
	Die Datei wurde von einem anderen Benutzer ausgecheckt und kann nicht ausgecheckt werden.



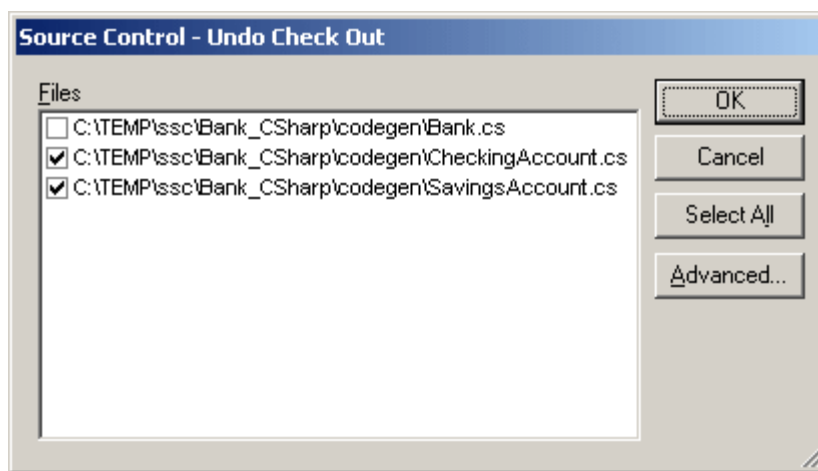
Lokal ausgecheckt. Die Datei kann bearbeitet und eingchecked werden.

Auschecken rückgängig...

Mit diesem Befehl werden Änderungen in zuvor ausgecheckten Dateien, also lokal aktualisierten Dateien **verworfen**. Es werden die alten Dateien aus der Versionskontrolldatenbank beibehalten.

Um das Auschecken rückgängig zu machen, gehen Sie folgendermaßen vor:

1. Wählen Sie die Dateien im Projektfenster aus
2. Wählen Sie **Projekt | Versionskontrolle | Auschecken rückgängig**.
3. Wählen Sie im Dialogfeld "Auschecken rückgängig" die gewünschten Dateien, für die das Auschecken rückgängig gemacht werden soll und klicken Sie auf OK.



Die Option "Auschecken rückgängig" kann auf die folgenden Dateien angewendet werden: (i) Einzelne Dateien - (klicken Sie im Projektfenster auf die Datei (oder mehrere Dateien mittels Strg + Klick); (ii) Ordner (klicken Sie im Projektfenster auf den Ordner, um ihn auszuwählen oder verwenden Sie Strg+Klick um mehrere Ordner zu selektieren)

Erweitert

Mit Hilfe dieser Option können Sie die Optionen **Replace writable** und **Set timestamp** in den entsprechenden Auswahllisten auswählen.



Mit dem Kontrollkästchen "Make writable" wird das "read-only" Attribut der abgerufenen Dateien entfernt.

Symbole der Versionskontrolle:

Die Ordner und Dateien sind mit bestimmten Symbolen versehen, deren Bedeutung im Folgenden erklärt wird.



Eingchecked. Die Datei kann ausgecheckt werden.

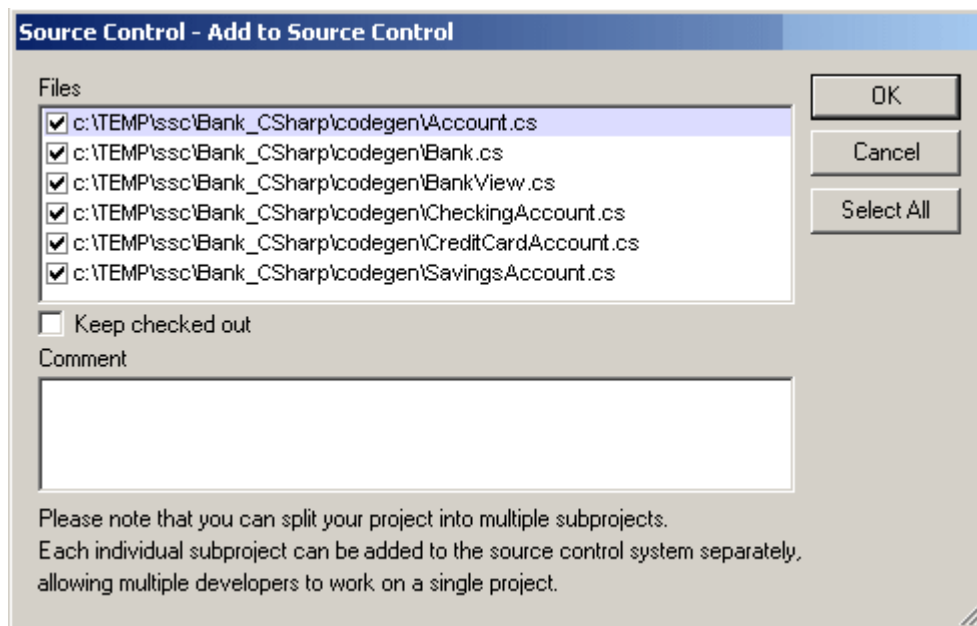
	Die Datei wurde von einem anderen Benutzer ausgecheckt und kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Die Datei kann bearbeitet und eingecheckt werden.

Zu Versionskontrolle hinzufügen

Fügt die ausgewählten Dateien oder Ordner in einem Projektordner zur Versionskontrolldatenbank hinzu und stellt sie unter Versionskontrolle.

Um Dateien zur Versionskontrolle hinzuzufügen, gehen Sie folgendermaßen vor:

1. Klicken Sie im Projektfenster auf eine Datei und wählen Sie die Menüoption **Projekt | Versionskontrolle | Zu Versionskontrolle hinzufügen**. Wenn Sie einen Ordner auswählen, werden die Dateien aus diesem Ordner zur Versionskontrolle hinzugefügt.
2. Stellen Sie sicher, dass die Dateien, die hinzugefügt werden sollen, im Dialogfeld "Zu Versionskontrolle hinzufügen", das angezeigt wird, angehakt sind. Wenn die hinzuzufügenden Dateien ausgecheckt bleiben sollen, aktivieren Sie das Kontrollkästchen "Ausgecheckt lassen" und klicken Sie anschließend auf OK.



Neben den einzelnen unter Versionskontrolle gestellten Dateien erscheint nun ein Vorhängeschloss. Wenn die Dateien ausgecheckt werden, werden sie mit einem roten Häkchen angezeigt.

Symbole der Versionskontrolle:

Die Ordner und Dateien sind mit bestimmten Symbolen versehen, deren Bedeutung im Folgenden erklärt wird.

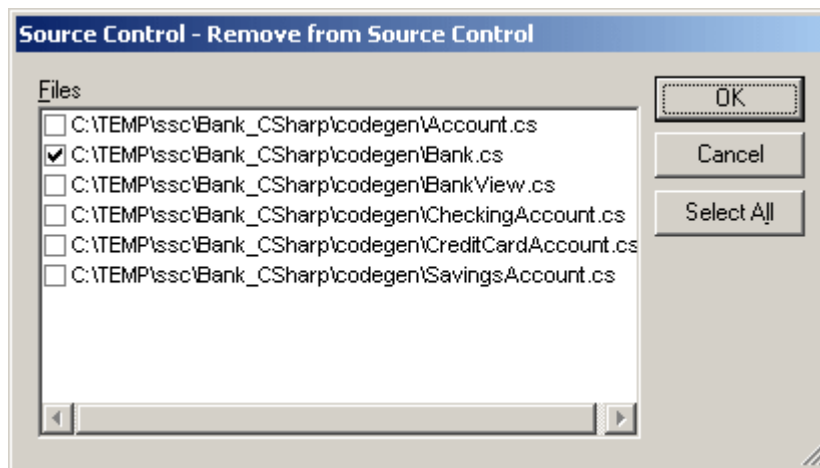
	Eingecheckt. Die Datei kann ausgecheckt werden.
	Die Datei wurde von einem anderen Benutzer ausgecheckt und kann nicht ausgecheckt werden.
	Lokal ausgecheckt. Die Datei kann bearbeitet und eingecheckt werden.

Von Versionskontrolle ausgliedern

Mit diesem Befehl **entfernen** Sie zuvor hinzugefügte Dateien aus dem Versionskontrollprodukt. Diese Dateiarten werden zwar im Projektfenster weiterhin angezeigt, können aber nicht ein- oder ausgecheckt werden. Mit Hilfe des Befehls "Zu Versionskontrolle hinzufügen" können Sie die Dateien wieder unter Versionskontrolle stellen.

Um Dateien aus der Versionskontrolle auszugliedern, gehen Sie folgendermaßen vor:

1. Wählen Sie die gewünschten Dateien im Projektfenster aus
2. Wählen Sie **Projekt | Versionskontrolle | Von Versionskontrolle ausgliedern**.
3. Wählen Sie im Dialogfeld "Von Versionskontrolle ausgliedern" die gewünschten Dateien aus, die entfernt werden sollen und klicken Sie auf OK.



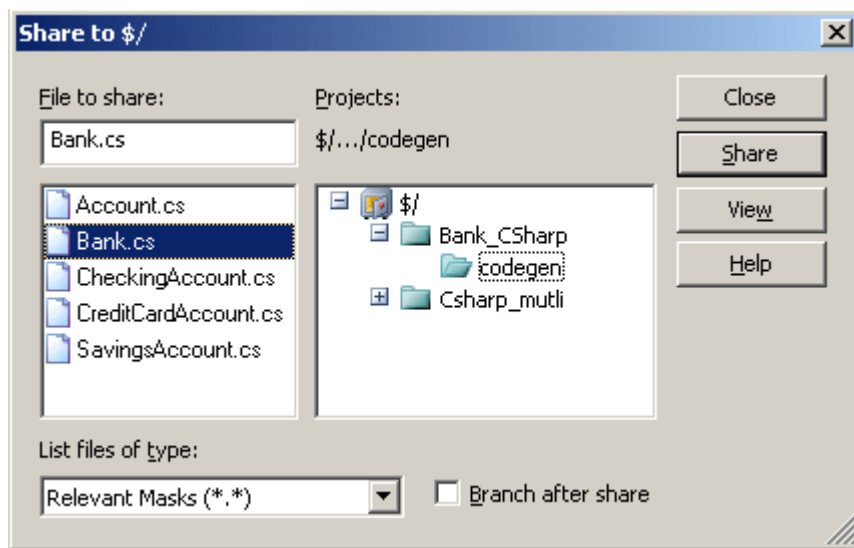
Folgende Objekte könne aus der Versionskontrolle ausgegliedert werden: (i) Einzelne Dateien - (klicken Sie im Projektfenster auf die Datei (oder mehrere Dateien mittels Strg + Klick); (ii) Ordner (klicken Sie im Projektfenster auf den Ordner, um ihn auszuwählen oder verwenden Sie Strg+Klick um mehrere Ordner zu selektieren)

Aus Versionskontrolle freigeben

Um dem Freigeben-Befehl verwenden zu können, müssen Sie für das Projekt, das Sie freigeben möchten, Ein- und Auscheck-Rechte besitzen.

Um eine Datei aus der Versionskontrolle freizugeben, gehen Sie folgendermaßen vor:

1. Wählen Sie die gewünschte Datei im Projektfenster aus und wählen Sie den Befehl **Projekt | Versionskontrolle | Aus Versionskontrolle freigeben**.
2. Wählen Sie im Listefeld "Projekte" den Projektordner aus, der die Datei enthält.



3. Wählen Sie im Listenfeld "Files to share" die gewünschte Datei aus und klicken Sie auf die Schaltfläche "Freigeben".
Die Datei wird nun aus der Liste "Files to share" entfernt.
4. Klicken Sie zum Fortfahren auf "Schließen".

Branch after share

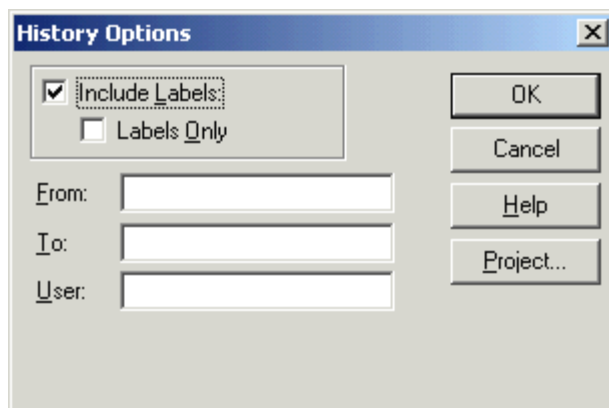
Gibt die Datei frei und erstellt einen neuen Branch zum Erstellen einer separaten Version.

Verlauf anzeigen

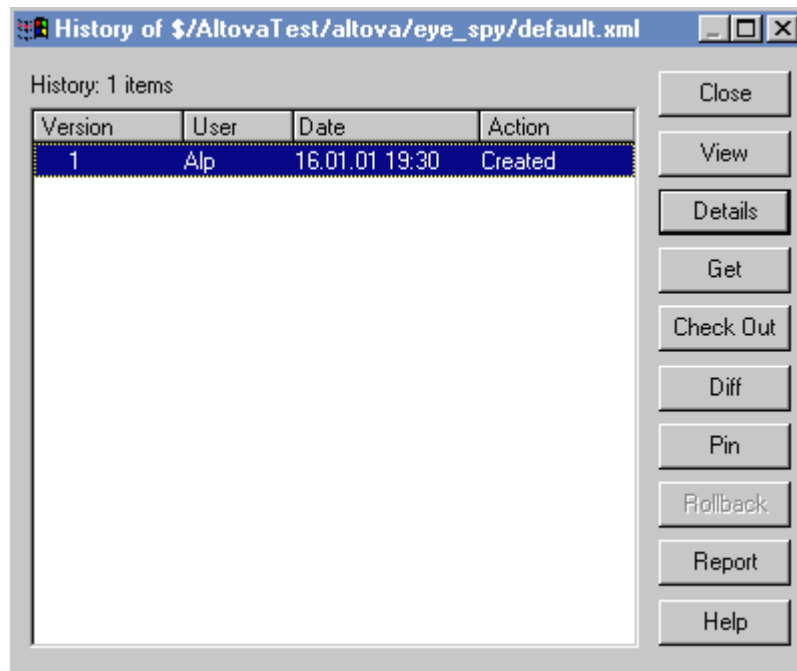
Mit diesem Befehl zeigen Sie den Verlauf einer unter Versionskontrolle stehenden Datei an. Sie können damit eine detaillierte Verlaufsliste anzeigen, genauere Verlaufsinformationen anzeigen, einen Vergleich durchführen oder frühere Versionen der Datei abrufen.
Um den Verlauf einer Datei anzuzeigen, gehen Sie folgendermaßen vor.

1. Klicken Sie Projektfenster auf die Datei, deren Verlauf Sie ansehen möchten.
2. Wählen Sie die Menüoption **Projekt | Versionskontrolle | Verlauf anzeigen**.

Es erscheint eventuell ein Dialogfeld, in dem Sie nähere Informationen eingeben müssen (in diesem Beispiel wird Visual Source Safe verwendet).



3. Wählen Sie den gewünschten Eintrag und bestätigen Sie mit OK.



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](#) 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](#), 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. 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.

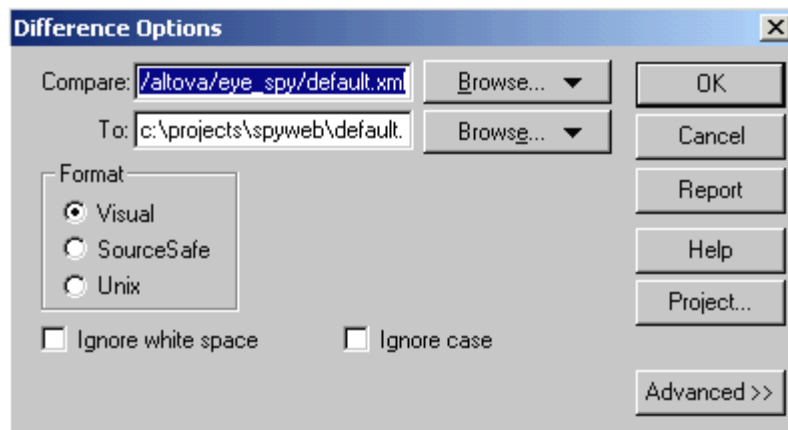
Unterschiede anzeigen

Dieser Befehl ermöglicht die Anzeige von Unterschieden zwischen der in der Versionskontroll-Ablage befindlichen und der gleichnamigen Datei, die Sie **ein-/ausgecheckt** haben. Wenn Sie eine der Dateien im Dialogfeld "Verlauf" mit einem Pin markiert haben, so wird diese Datei im Textfeld "Vergleichen" verwendet. Sie können mit Hilfe der

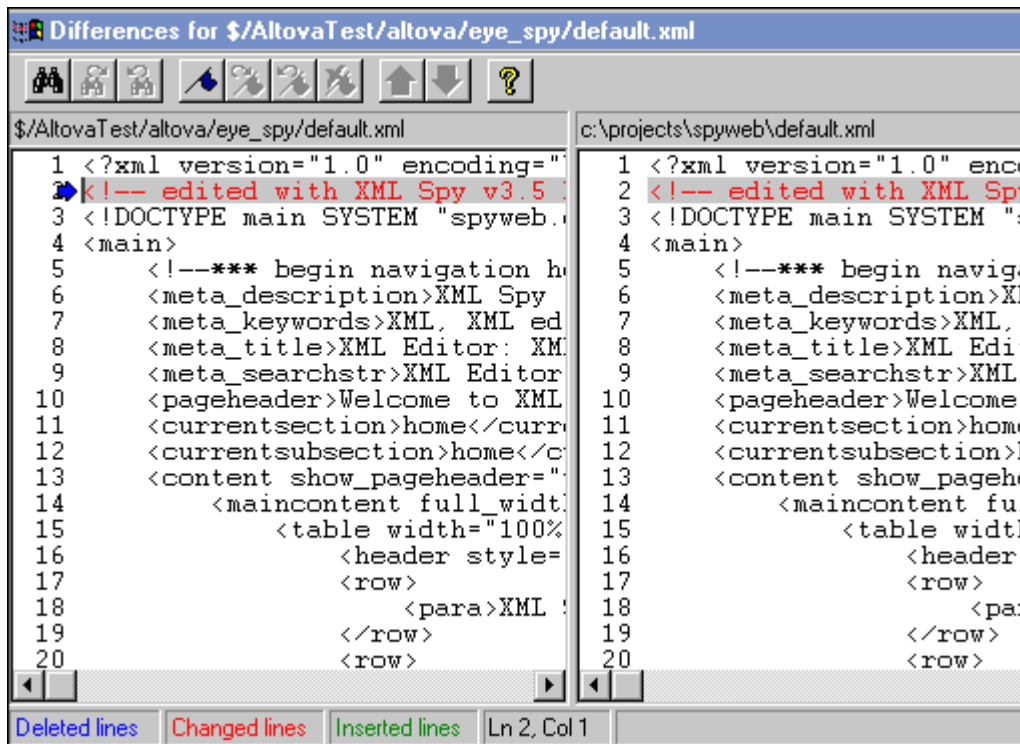
Durchsuchen-Schaltfläche 2 beliebige Dateien auswählen.

So zeigen Sie die Unterschiede zwischen zwei Dateien an:

1. Checken Sie eine Datei aus Ihrem Projekt aus. Klicken Sie im Projektfenster auf die Datei.
2. Wählen Sie die Menüoption **Projekt | Versionskontrolle | Unterschiede anzeigen**. Es erscheint eventuell ein Dialogfeld, in dem Sie weitere Informationen eingeben können.



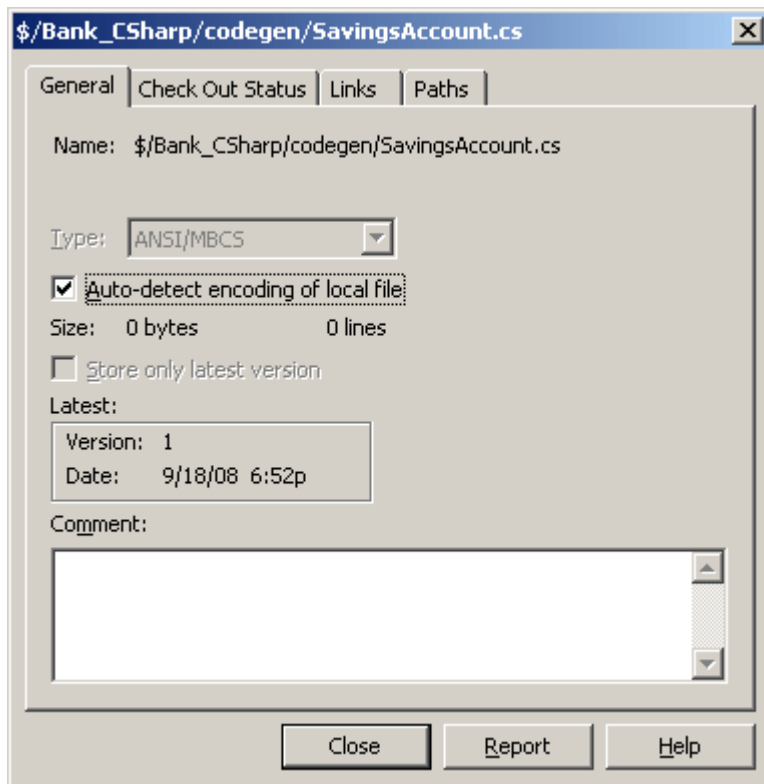
3. Wählen Sie die gewünschten Einträge und bestätigen Sie mit OK.



Die Unterschiede zwischen beiden Dateien werden in beiden Fenstern farblich markiert (in diesem Beispiel wird MS Source-Safe verwendet).

Eigenschaften anzeigen

Mit diesem Befehl können Sie die Eigenschaften der ausgewählten Datei anzeigen (*Abbildung unten*). Die Anzeige kann je nach verwendetem Versionskontrollprodukt unterschiedlich sein.



Beachten Sie, dass dieser Befehl nur für einzelne Dateien verwendet werden kann.

Status aktualisieren

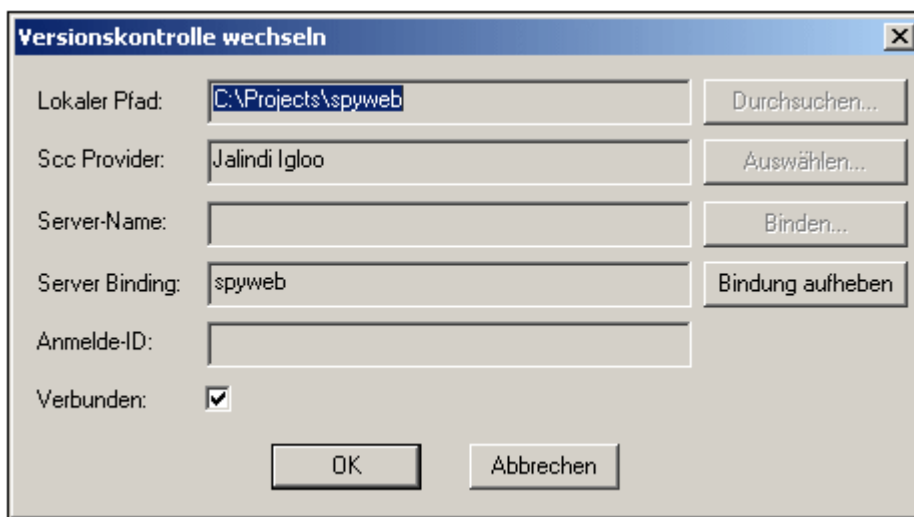
Mit diesem Befehl **aktualisieren** Sie den Status aller Projektdateien unabhängig von ihrem derzeitigen Status.

Versionskontrollmanager

Mit diesem Befehl **starten** Sie die ursprüngliche Oberfläche Ihrer Versionskontroll-Software.

Versionskontrolle wechseln

Über dieses Dialogfeld können Sie das Versionskontrollsystem wechseln. Klicken Sie dazu zuerst auf die Schaltfläche "Bindung aufheben" und anschließend auf die Schaltfläche "Auswählen", um ein neues Versionskontrollsystem auszuwählen.



Nachdem Sie den Versionskontroll-Provider ausgewählt haben, müssen Sie ein Binding zwischen dem lokalen Ordner und dem Versionskontrollserver erstellen. Der lokale Ordner ist derjenige, der im Testfeld "Lokaler Pfad" eingegeben ist. Um einen Ordner auf dem Versionskontrollserver auszuwählen, klicken Sie auf die Schaltfläche **Binden**, damit Sie sich beim Server anmelden und anschließend zum gewünschten Ordner auf dem Versionskontrollserver navigieren können. Klicken Sie anschließend auf **OK**.

18.3.7 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](#)).

18.3.8 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](#) 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](#) definierten Dateierweiterung zum entsprechenden Ordner hinzugefügt. Eine Beschreibung globaler Ressourcen finden Sie im Abschnitt "Globale Ressourcen" in dieser Dokumentation.

18.3.9 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.

18.3.10 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.

18.3.11 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 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.

18.3.12 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](#) gelb sind.

18.3.13 Externes Verzeichnis zum Projekt hinzufügen

Mit diesem Befehl fügen Sie einen neues **externes Verzeichnis** in das aktuelle Projekt ein. Verwenden Sie diese Funktion, um ein lokales Verzeichnis (oder ein Netzwerkverzeichnis) in das aktuelle Projekt einzufügen. Sie können diesen Befehl auch über das Kontextmenü aufrufen (Rechtsklick auf ein Verzeichnis im Projektfenster). Bitte beachten Sie, dass Dateien in externen Verzeichnissen nicht unter Versionskontrolle gestellt werden können.

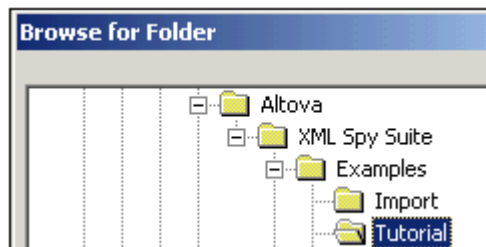
Anmerkung: Externe Ordner sind gelb, während [Projektordner](#) grün sind.

Anmerkung: In externen Ordnern enthaltene Dateien können nicht unter Versionskontrolle gestellt werden.

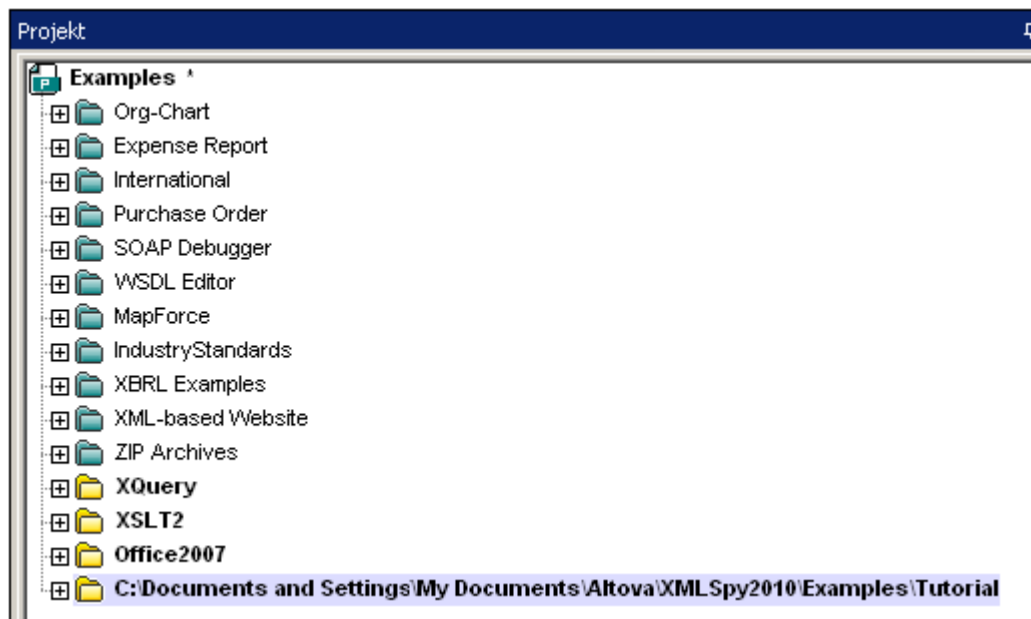
Hinzufügen von externen Ordnern zu Projekten

So fügen Sie einen externen Ordner zum Projekt hinzu:

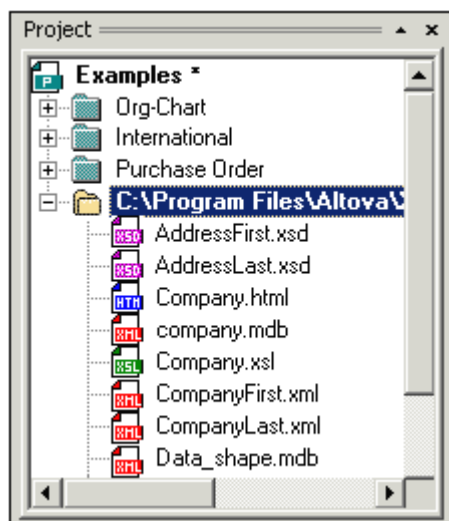
1. Wählen Sie die Menüoption **Projekt | Externes Verzeichnis zum Projekt hinzufügen...**
2. Wählen Sie das Verzeichnis, welches hinzugefügt werden soll (Dialogfeld "Ordner suchen"), und klicken Sie auf OK zum Bestätigen.



Das ausgewählte Verzeichnis wird nun im Projektfenster angezeigt.



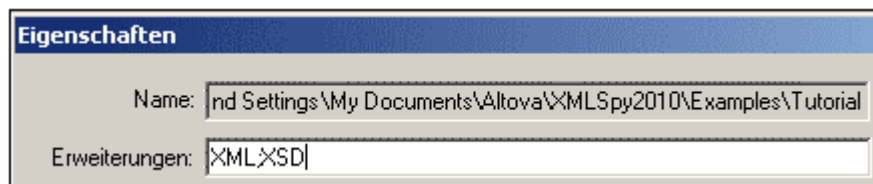
3. Klicken Sie auf das Symbol "+", um den Inhalt des Verzeichnisses anzuzeigen.



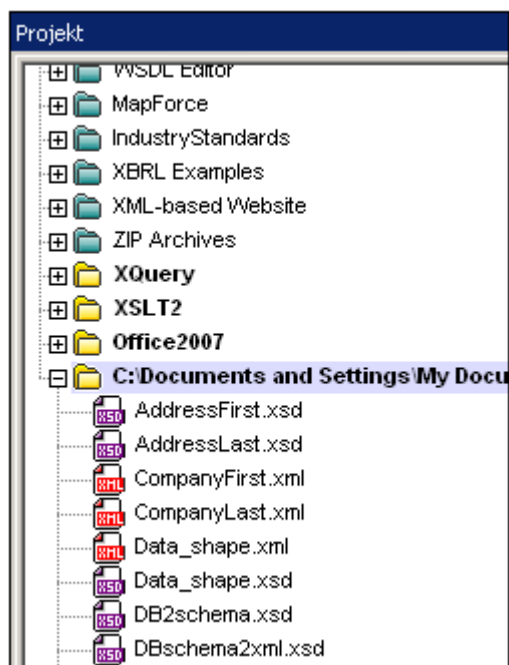
Filtern von Ordnerinhalt

So filtern Sie den Inhalt eines Ordners:

1. Rechtsklicken Sie auf den lokalen Ordner. Wählen Sie im Popup-Menü die Option **Eigenschaften**.
Das Dialogfeld Eigenschaften wird geöffnet.



2. Klicken Sie auf das Feld **Erweiterungen**, und geben Sie die Erweiterungen der Dateien an, die angezeigt werden sollen. Bei mehreren Erweiterungen trennen Sie diese durch ein **Semikolon** (in diesem Beispiel XML und XSD Schema).
3. Klicken Sie auf OK um Ihre Auswahl zu bestätigen.



Im Projektfenster werden jetzt ausschließlich XML und XSD Dateien im Verzeichnis "Tutorial" angezeigt.

Validieren von externen Ordnern

So validieren Sie ein externes Verzeichnis und prüfen es auf Wohlgeformtheit:

1. Wählen Sie aus, welche Dateiarnten angezeigt werden sollen oder markieren Sie diese im externen Ordner.
2. Klicken Sie auf den Ordner und anschließend auf das Symbol **Wohlgeformtheit**

prüfen  oder **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 Projektverzeichnis

In lokalen oder netzwerkgebundenen Verzeichnissen können jederzeit Dateien gelöscht oder hinzugefügt werden. Um die Verzeichnis-Ansicht zu aktualisieren, Rechtsklicken Sie auf das externe Verzeichnis, und wählen Sie im Popup-Menü die Option **Externes Verzeichnis aktualisieren**.

Dateien oder Verzeichnisse löschen:

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.

18.3.14 Externes Webverzeichnis zum Projekt hinzufügen

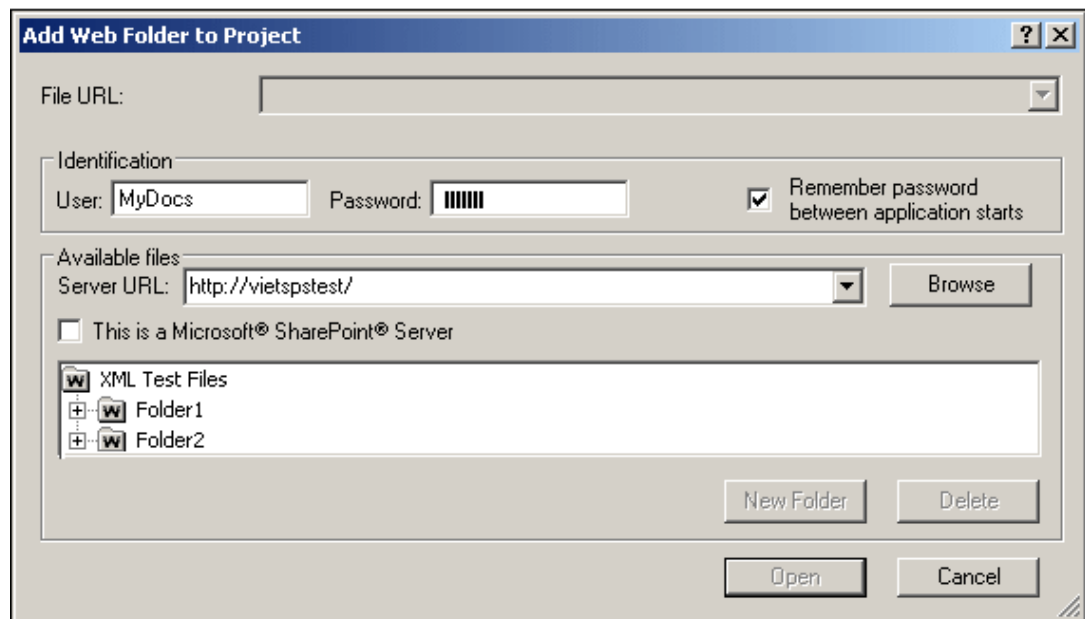
Verwenden Sie diese Funktion, um ein neues Webverzeichnis in das aktuelle Projekt einzufügen. Sie können diesen Befehl auch über das Kontextmenü (Rechtsklick auf ein Verzeichnis im Projektfenster) aufrufen. Bitte beachten Sie, dass Dateien in externen Verzeichnissen nicht unter Versionskontrolle gestellt werden können.

Hinzufügen eines externen Webverzeichnisses zum Projekt

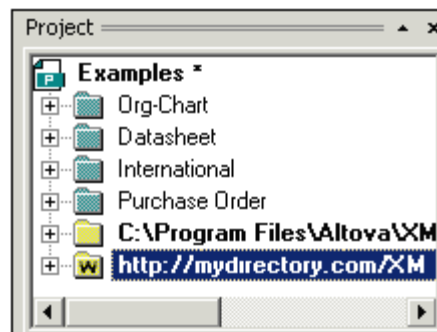
Um ein externes Webverzeichnis zum Projekt hinzuzufügen, gehen Sie folgendermaßen vor:

1. Wählen Sie die Menüoption **Projekt | Externes Webverzeichnis zum Projekt hinzufügen**. Daraufhin wird das Dialogfeld "Webordner in Projekt einfügen" angezeigt (*Abbildung unten*).

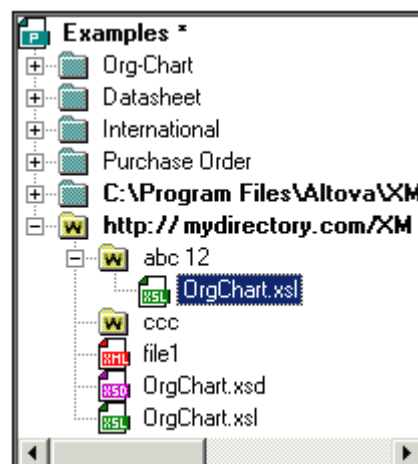
2. Klicken Sie in das Feld "Server URL" und geben Sie die URL des Servers ein. Wenn es sich beim Server um einen Microsoft® SharePoint® Server handelt, aktivieren Sie das Kontrollkästchen *Microsoft® SharePoint® Server*. Nähere Informationen zum Arbeiten mit Dateien dieses Servertyps finden Sie im Abschnitt *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.



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 Verzeichnisses

Um die Dateien in einem Ordner auf Wohlgeformtheit zu prüfen oder sie zu validieren, klicken

Sie auf das Verzeichnis und anschließend auf das Symbol **Wohlgeformtheit prüfen**  bzw.

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

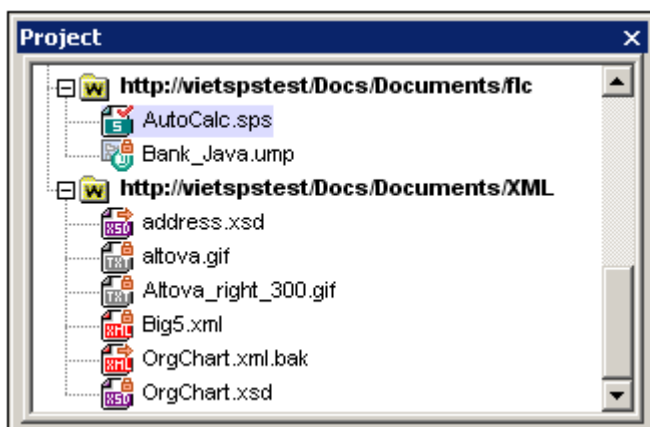
Da es sich um einen Webordner handelt, der zum Projekt hinzugefügt wurde, kann nur der Webordner (und nicht die Dateien darin) aus dem Projekt gelöscht werden. Um einen Webordner aus dem Projektfenster zu löschen, klicken Sie entweder (i) mit der rechten Maustaste auf den Ordner und wählen Sie den Befehl **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.

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](#) gespeichert werden, sodass Sie diese Zugriffsdaten nicht jedes Mal eingeben müssen, wenn Sie auf den Server zugreifen.

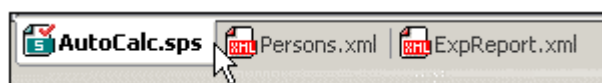


Die Dateien im Projektfenster (*Abbildung unten*) sind mit Symbolen versehen, die den Status der Dateien (Eingecheckt/Ausgecheckt) kennzeichnen. Hier sehen Sie die verschiedenen Dateisymbole:

	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.

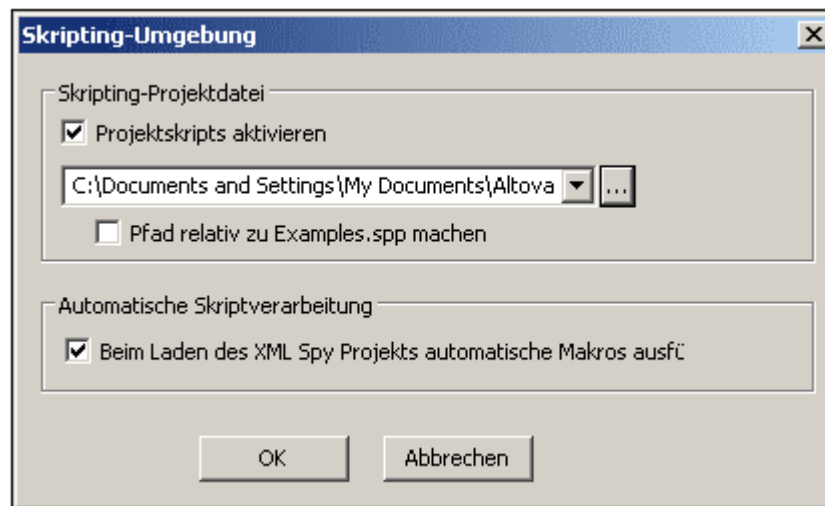


- Wenn eine Datei von einem anderen Benutzer ausgecheckt wurde, kann sie nicht ausgecheckt werden.
- Wenn eine Datei lokal von Ihnen ausgecheckt wurde, können Sie dies mit dem Kontextmenübefehl "Auschecken rückgängig" rückgängig machen. In diesem Fall wird die Datei dem Server unverändert zurückgegeben.
- 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.

18.3.15 Skript-Einstellungen

Ein Skripting-Projekt wird einem XMLSpy-Projekt auf folgende Weise zugewiesen:

- Öffnen Sie auf der XMLSpy-Benutzeroberfläche das gewünschte Applikationsprojekt.
- 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*.

18.3.16 Projekteigenschaften



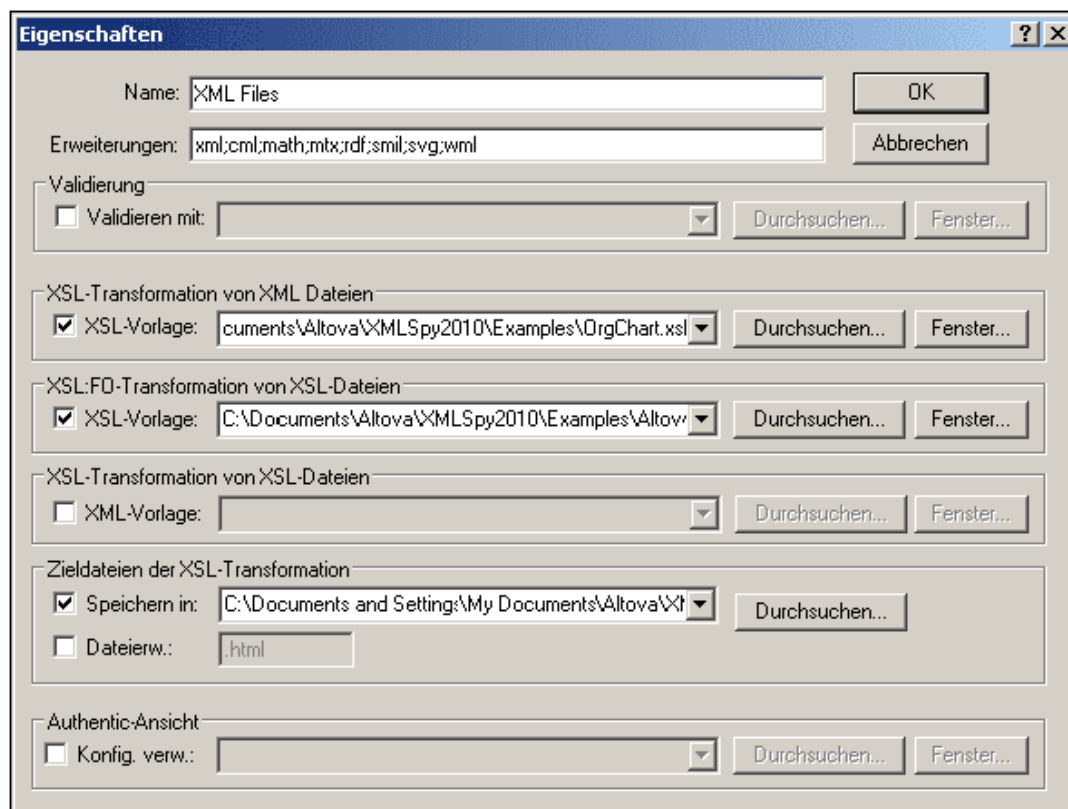
Mit diesem Befehl können Sie wichtige Einstellungen für spezifische Ordner Ihres Projekts definieren.

So definieren Sie Projekteigenschaften für ein Verzeichnis:

1. Rechtsklicken Sie auf das Verzeichnis, für welches Sie Einstellungen vornehmen möchten, und
2. Wählen Sie den Befehl **Eigenschaften...** aus dem Kontextmenü.

Bitte beachten Sie:

Falls die Projektdatei unter Versionskontrolle steht, werden Sie gefragt, ob die Projektdatei (*.spp) ausgecheckt werden soll. Nach dem Ändern der Einstellungen klicken Sie auf OK um diese zu speichern.



Die Zuweisungen in den Feldern "XSL-Vorlage" bzw. "XML-Vorlage" haben Vorrang vor anderen lokalen Zuweisungen innerhalb einer XML-Datei. So wird z.B. die Datei OrgChart.xml ("XSL-Vorlage") immer dann verwendet, wenn XML-Dateien im **XML-Dateiverzeichnis** transformiert werden. Die hier vorgenommene Zuweisung hat auch Vorrang vor anderen Zuweisungen, die für übergeordnete Ordner vorgenommen wurden.

Dateierweiterungen

Die Dateierweiterungen helfen beim Hinzufügen neuer Dateien zum Projekt (im Gegensatz zum Einordnen in einen bestimmten Ordner) bei der automatischen Einordnung der Dateien in Ordner eines Projekts.

Benutzer-ID und Passwort für externe Ordner

Unter den Eigenschaften von externen Ordnern (auch externen Webverzeichnissen) können Sie die Benutzer-ID und das Passwort speichern, die unter Umständen für den Zugriff auf den Server benötigt werden.

Validierung

Definieren Sie das DTD- oder Schemadokument, welches bei der [Validierung](#) aller Dateien des aktuellen Verzeichnisses verwendet werden soll (in diesem Beispiel die Hauptseiten).

XSL Transformation von XML-Dateien

Definieren Sie das XSL Stylesheet, welches zur [XSL Transformation](#) aller Dateien eines Verzeichnisses verwendet werden soll.

Wenn Sie selbst XSL Stylesheets erstellen, können Sie ein XML-Beispieldokument zuweisen, das als Vorschau für das XSL Stylesheet bei der XML-Transformation dient. Sie können dies anstelle des XML Instanzdokuments verwenden.

XSL:FO Transformation von XML-Dateien

Sie können das XSL Stylesheet mit XSL:FO Markup definieren, das für eine [XSL:FO Transformation](#) aller Dateien eines Verzeichnisses verwendet werden soll.

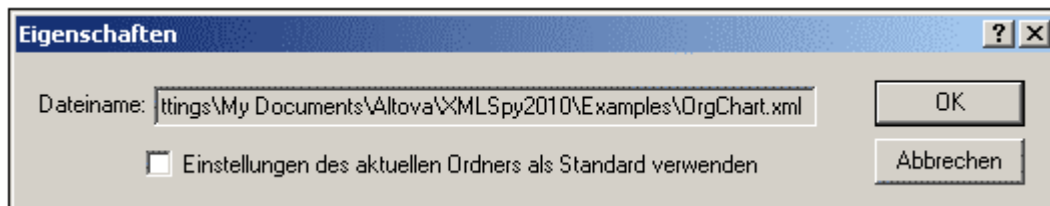
Zieldateien bei XSL Transformation

Für Batchtransformationen in XSL können Sie das Zielverzeichnis angeben, in dem die transformierten Dateien abgelegt werden sollen.

Wenn Sie eine Datei oder URL in mehreren Verzeichnissen Ihres Projekts abgelegt haben, können Sie über das Dialogfeld "Eigenschaften" das Standardverzeichnis definieren, mit dessen Einstellungen die Validierung oder Transformation im non-batch Modus erfolgen soll. Aktivieren Sie dazu das Kontrollkästchen **Einstellungen des aktuellen Ordners als Standard verwenden** (siehe Screenshot).

So rufen Sie das Dialogfeld "Eigenschaften" auf und aktivieren das Kontrollkästchen:

1. Kopieren Sie eine XML-Datei in einem Projekt in einen anderen Ordner.
2. Rechtsklicken Sie im Projektfenster auf die kopierte Datei und wählen Sie im Kontextmenü den Befehl **Eigenschaften**.



Authentic

Die Option "Konfig. verwenden" ermöglicht es Ihnen, das StyleVision Power Stylesheet (SPS Datei) im aktuellen Verzeichnis auszuwählen, welche beim Editieren von XML Dateien in der Authentic-Ansicht zugrunde gelegt werden soll. Nachdem Sie die Schema-, SPS- und XML-Dateien miteinander verknüpft und in ein Projekt eingegeben haben, kann ein Verschieben dieser Dateien in einen anderen Ordner Fehler bei den Verknüpfungen verursachen.

Um solche Fehler zu vermeiden, sollten Sie am besten einen endgültigen Speicherort für Ihre Schema-, SPS- und XML-Dateien festlegen, bevor Sie sie miteinander verknüpfen und einem Projekt zuweisen. Weitere Informationen zur Verwendung der Authentic-Ansicht finden Sie im Handbuch zur Authentic-Ansicht.

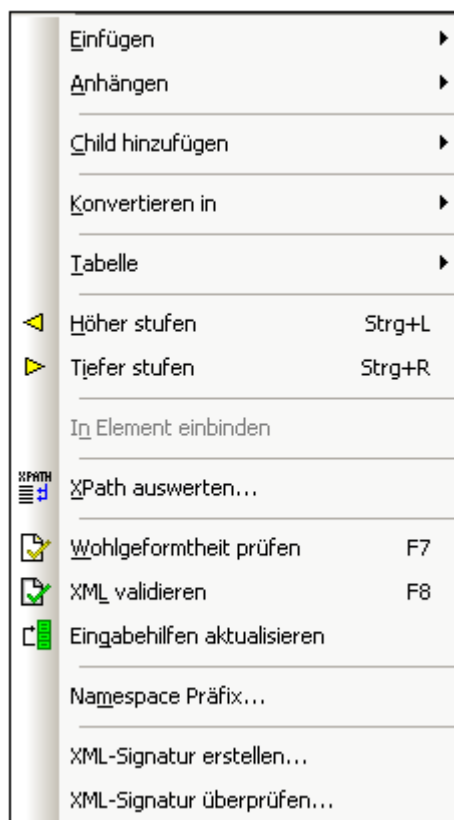
18.3.17 Zuletzt verwendete Projekte

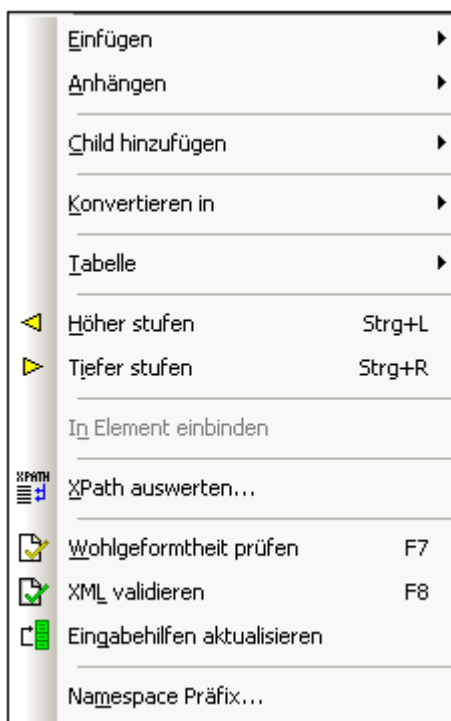
Mit dieser Funktion werden die neun zuletzt verwendeten Projekte angezeigt um einen schnelleren Zugriff auf diese zu erhalten.

Beachten Sie bitte, dass XMLSpy automatisch das [zuletzt verwendete Projekt](#) beim Starten von XMLSpy öffnen kann. (**Extras | Optionen**| Register **Datei**, Projekt | Letztes Projekt beim Programmstart öffnen).

18.4 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 XPath's im Zusammenhang mit einzelnen XML-Dokumenten evaluieren.





Zu den am häufigsten verwendeten XML-Funktionen zählen die [Überprüfung von Dokumenten auf ihre Wohlgeformtheit](#) und [Gültigkeit](#). Sie finden diese Befehle in diesem Menü.

18.4.1 Einfügen

Der Befehl **Einfügen** ist zwar in allen Menüs aktiv, kann aber nur in der Grid-Ansicht verwendet werden. Er enthält ein Untermenü (*siehe Screenshot*), über das Sie folgende Möglichkeiten haben:

- die XML-Deklaration und Node-Typen (Attribut, Element, Text, CDATA, Comment, Processing Instruction) in XML-Dokumente einfügen,
- DOCTYPE-Deklarationen und externe DTD-Deklarationen in XML-Dokumente einfügen und
- DTD-Deklarationen (ELEMENT, ATTLIST, ENTITY und NOTATION) in DTD-Dokumente und interne DTD-Deklarationen von XML-Dokumenten einfügen.

	A <u>tt</u> ribute	Strg+I
	E <u>l</u> ement	Strg+E
	T <u>e</u> xt	Strg+T
	C <u>D</u> ATA	Strg+D
	C <u>o</u> mment	Strg+M
	X <u>M</u> L	
	P <u>r</u> ocessing Instruction	
	X <u>I</u> nclude...	
	D <u>O</u> CTYPE	
	E <u>x</u> ternalID	
	E <u>L</u> EMENT	
	A <u>T</u> TLIST	
	E <u>N</u> TITY	
	N <u>O</u> TATION	
	Externe k <u>o</u> dierte Datei...	

Attribut einfügen



Strg+Umschalt+I

Diese Funktion ist nur in der Grid-Ansicht aktiv und fügt ein neues **Attribut** vor dem ausgewählten Datenelement ein. Ein eingefügtes Attribut kann in der Grid-Ansicht einige Zeilen vor dem aktuellen Datenelement angezeigt werden, da Attribute in der Grid-Ansicht unmittelbar nach deren Parent-Element und vor allen Child-Elementen dieses Parent-Elements angezeigt werden.

Element einfügen



Strg+Umschalt+E

Diese Funktion steht nur in der Grid-Ansicht zur Verfügung und fügt ein neues **Element** vor dem ausgewählten Datenelement ein. Falls die Auswahl ein Attribut ist, wird das neue Element vor dem ersten Child-Element des Parent-Elements des Attributs eingefügt.

Text einfügen



Strg+Umschalt+T

Diese Funktion steht nur in der Grid-Ansicht zur Verfügung und fügt eine neue Textzeile vor dem ausgewählten Datenelement ein. Falls die Auswahl ein Attribut ist, wird die Textzeile nach dem Attribut und vor dem ersten Child-Element des Parent-Elements des Attributs eingefügt.

CDATA einfügen



Strg+Umschalt+D

Diese Funktion steht nur in der Grid-Ansicht zur Verfügung und fügt einen neuen CDATA-Block vor dem ausgewählten Datenelement ein. Falls die Auswahl ein Attribut ist, wird der CDATA-Block nach dem Attribut und vor dem ersten Child-Element des Parent-Elements des Attributs eingefügt.

Comment einfügen



Strg+Umschalt+M

Diese Funktion steht nur in der Grid-Ansicht zur Verfügung und fügt einen neuen **Comment** vor dem ausgewählten Datenelement ein. Falls die Auswahl ein Attribut ist, wird die neue Comment-Zeile nach dem Attribut und vor dem ersten Child-Element des Parent-Elements des Attributs eingefügt.

XML einfügen



Diese Funktion steht nur in der Grid-Ansicht zur Verfügung und fügt eine Zeile für die XML-Deklaration vor dem ausgewählten Datenelement ein. Sie müssen die Child-Attribute der XML-Deklaration und die Werte dieser Attribute einfügen. Eine XML-Deklaration muss ungefähr folgendermaßen aussehen:

```
<?xml version="1.0" encoding="UTF-8"?>
```

Bitte beachten Sie: Da ein XML-Dokument nur eine XML-Deklaration am Anfang der Datei enthalten darf, sollten Sie diesen Befehl nur verwenden, wenn die erste Zeile des Dokuments ausgewählt ist und wenn das Dokument noch keine XML-Deklaration enthält.

Processing Instruction einfügen



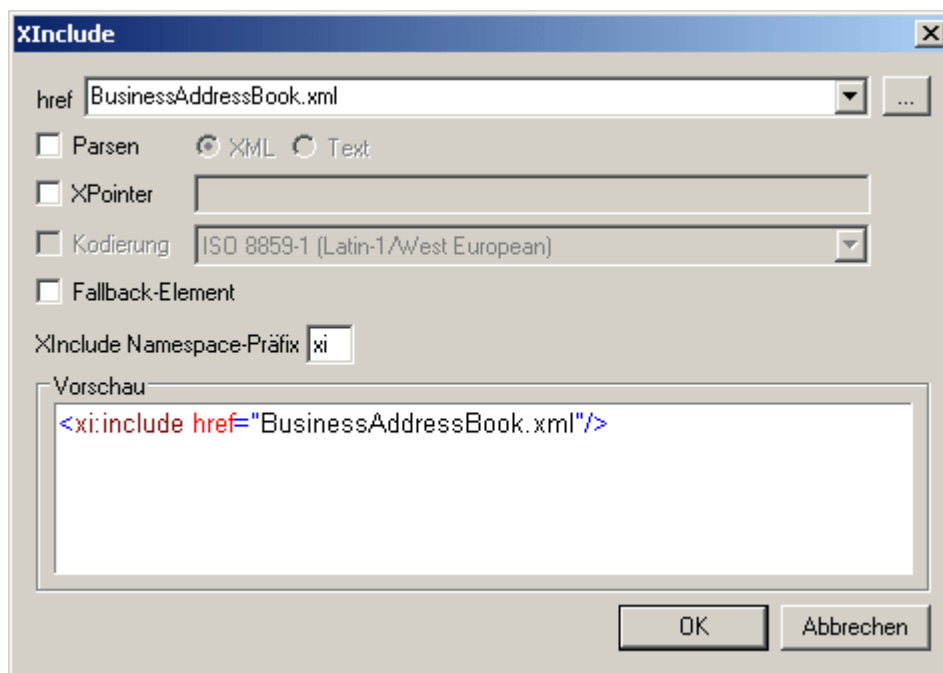
Diese Funktion steht nur in der Grid-Ansicht zur Verfügung und fügt eine neue Processing Instruction (PI) vor dem ausgewählten Datenelement ein. Falls die Auswahl ein Attribut ist, wird die PI nach dem Attribut und vor dem ersten Child-Element des Parent-Elements des Attributs eingefügt.

XInclude einfügen



Der Befehl **XML | Einfügen | XInclude** steht nur in der Grid-Ansicht zur Verfügung und fügt ein neues XInclude-Element vor dem ausgewählten Objekt ein. Wenn 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).



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 geparkt wird. (XML ist die Standardeinstellung und muss daher nicht extra angegeben werden.) Das Attribut `xpointer` identifiziert ein bestimmtes Fragment des Dokuments, das mit dem Attribut `href` lokalisiert wird; dieses Element wird inkludiert. Das Attribut `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 [Attribute mit dem Präfix :xml](#) des Abschnitts Schema-Ansicht.

XPointers

XMLSpy unterstützt XPointers in XInclude. Die relevanten W3C Recommendations dazu sind die [XPointer Framework](#) und die [XPointer element\(\) Scheme](#) 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.

DOCTYPE einfügen



Der Befehl **DOCTYPE einfügen** steht in der Grid-Ansicht einer XML-Datei zur Verfügung, wenn der Node der obersten Ebene ausgewählt ist. Sie hängen damit eine DOCTYPE-Deklaration am Beginn des XML-Dokuments an. Sie müssen den Namen des DOCTYPE angeben und dieser Name muss mit dem Namen des Dokument-Elements übereinstimmen.



Nach Eingabe des DOCTYPE-Namens können Sie die Deklarationen eingeben, die in der internen DTD-Untermenge verwendet werden sollen.

Bitte beachten Sie:

- Die DOCTYPE-Deklaration darf nur zwischen der XML-Deklaration und dem XML-Dokument-Element stehen.
- Sie können stattdessen auch den Befehl [DTD zuweisen](#) verwenden, mit dem Sie eine DOCTYPE-Anweisung erstellen, die auf ein externes DTD-Dokument referenziert.

ExternalID einfügen



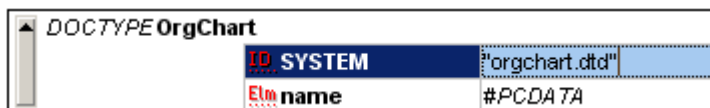
Eine DOCTYPE-Deklaration in einer XML-Datei kann eine Referenz auf eine externe Ressource, die eine DTD-Deklaration enthält, beinhalten. Diese Ressource wird entweder über einen Public- oder einen System Identifier referenziert.

Beispiel:

```
<!DOCTYPE doc_element_name PUBLIC "publicID" "systemID">
<!DOCTYPE doc_element_name SYSTEM "systemID">
```

Ein System Identifier ist eine URI, die eine externe Ressource spezifiziert. Der Public Identifier ist ein pfadunabhängiger Identifier, der dazu dient, die Referenz auf eine externe Ressource aufzuheben. So sind z.B. in Ihrem XMLSpy Installationsverzeichnis URIs für häufig verwendete DTDs und XML-Schemas in den verschiedenen Schema-Ordern unter `C:\Program Files\Altova\Common2012\Schemas\` in den Katalogdateien mit dem Namen `catalog.xml` gespeichert. Um die Referenz auf eine in diesen Katalogdateien aufgelistete DTD aufzuheben, können Sie einen Public Identifier in einem XML-Dokument verwenden.

Der Befehl **XML | Einfügen | ExternalID** steht zur Verfügung, wenn in der Grid-Ansicht in einer XML-Datei ein Child-Datenelement der DOCTYPE-Deklaration ausgewählt ist. Mit diesem Befehl fügen Sie in der Grid-Ansicht eine Zeile für einen externen Identifier (`PUBLIC` oder `SYSTEM`) ein. Sie müssen die Art des Identifiers und seinen Wert angeben.



In der Textansicht, die der oben gezeigten Abbildung aus der Grid-Ansicht entspricht, sieht dies aus wie folgt:

```
<!DOCTYPE OrgChart SYSTEM "orgchart.dtd" [  
  <!ELEMENT name ( #PCDATA) >  
>
```

Bitte beachten Sie: Sie können eine Zeile für eine External ID als Child hinzufügen, wenn das Datenelement DOCTYPE ausgewählt ist oder Sie können diese einfügen oder anhängen, wenn eines der Child-Elemente des Datenelements DOCTYPE, z.B. die ELEMENT-Deklaration `name` im obigen Beispiel, ausgewählt ist.

Element einfügen



Diese Funktion steht nur in der Grid-Ansicht und nur für DTD-Dokumente zur Verfügung oder, wenn in der DOCTYPE-Deklaration eines XML-Dokuments ein Datenelement ausgewählt ist. Der Befehl fügt eine ELEMENT-Deklaration vor der ausgewählten Deklaration ein.

ATTLIST einfügen



Diese Funktion steht nur in der Grid-Ansicht und nur für DTD-Dokumente zur Verfügung oder, wenn in der DOCTYPE-Deklaration eines XML-Dokuments ein Datenelement ausgewählt ist. Der Befehl fügt eine ATTLIST-Deklaration vor der ausgewählten Deklaration ein.

ENTITY einfügen



Diese Funktion steht nur in der Grid-Ansicht und nur für DTD-Dokumente zur Verfügung oder, wenn in der DOCTYPE-Deklaration eines XML-Dokuments ein Datenelement ausgewählt ist. Der Befehl fügt eine ENTITY-Deklaration vor der ausgewählten Deklaration ein.

NOTATION einfügen



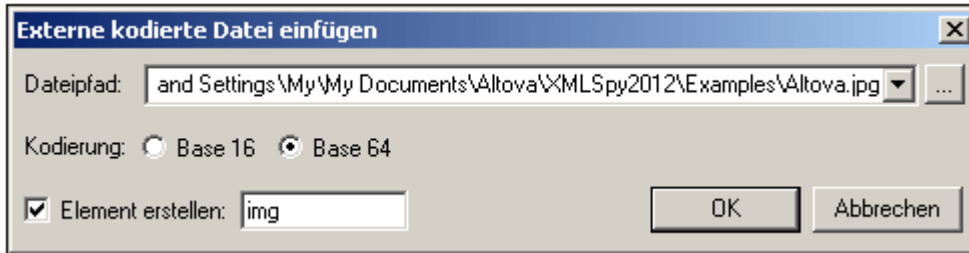
Diese Funktion steht nur in der Grid-Ansicht und nur für DTD-Dokumente zur Verfügung oder, wenn in der DOCTYPE-Deklaration eines XML-Dokuments ein Datenelement ausgewählt ist. Der Befehl fügt eine NOTATION-Deklaration vor der ausgewählten Deklaration ein.

Externe kodierte Datei einfügen

Der Befehl **XML | Einfügen | Externe kodierte Datei** steht nur in der Grid-Ansicht zur Verfügung. Mit diesem Befehl wird eine kodierte binäre Datei wie z.B. eine Bilddatei in Form

kodierter Zeichen eingefügt. Die externe kodierte Datei wird vor dem in der Grid-Ansicht ausgewählten Abschnitt eingefügt.

Bei Auswahl dieses Befehls wird das Dialogfeld "Externe kodierte Datei einfügen" (*Abbildung unten*) aufgerufen. In dieses Dialogfeld müssen Sie den Pfad zur Datei eingeben, die gewünschte Kodierung auswählen und angeben, ob die kodierte Datei in ein Element eingefügt werden soll oder nicht.



Die kodierte Datei wird in der Grid-Ansicht eingefügt (*markiertes Element in der Abbildung unten*).

Person	
First	Fred
Last	Landis
Title	Project Manager
Phone	123-456-7890
Email	f.landis@nanonull.com

img	
path	C:\Examples\Altova.jpg
encoding	xs:base64Binary
base64Text	/9j/4AAQSkZJRgABAgEASABIAAD/4QqTRXhpZi...

expense-item (4)			
	type	expto	Date
1	Lodging	Sales	2003-01-01
2	Lodging	Development	2003-01-02
3	Lodging	Marketing	2003-01-02
4	Entertainment	Development	2003-01-02

18.4.2 Anhängen

Der Befehl **Anhängen** ist zwar in allen Menüs aktiv, kann aber nur in der Grid-Ansicht verwendet werden. Er enthält ein Untermenü (*siehe Screenshot*), über das Sie folgende Möglichkeiten haben:

- die XML-Deklaration und Node-Typen (Attribut, Element, Text, CDATA, Comment, Processing Instruction) in XML-Dokumenten anhängen,
- DOCTYPE-Deklarationen und externe DTD-Deklarationen in XML-Dokumenten anhängen und
- DTD-Deklarationen (ELEMENT, ATTLIST, ENTITY und NOTATION) in DTD-Dokumenten und internen DTD-Deklarationen von XML-Dokumenten anhängen.

	A_ttribute	Strg+I
	E_lement	Strg+E
	T_ext	Strg+T
	C_DATA	Strg+D
	C_omment	Strg+M
	XML	
	P_rocessing Instruction	
	XInclude...	
	D_OCTYPE	
	E_xternalID	
	ELEMENT	
	ATTLIST	
	ENTITY	
	NOTATION	
	Externe kodierte Datei...	

Attribut anhängen



Strg+I

Diese Funktion steht nur in der Grid-Ansicht zur Verfügung und hängt ein neues Attribut an.

Element anhängen



Strg+E

Diese Funktion steht nur in der Grid-Ansicht zur Verfügung und hängt ein neues **Element** nach dem letzten Geschwisterelement des ausgewählten Elements an. Falls es sich bei der aktuellen Auswahl um einen Attribut-Node handelt, wird der Element-Node nach dem letzten Child des Parent-Elements des Attributs angehängt.

Text anhängen



Strg+T

Diese Funktion steht nur in der Grid-Ansicht zur Verfügung und hängt einen neuen Textblock nach dem letzten Geschwisterelement des ausgewählten Elements an. Falls es sich bei der Auswahl um einen Attribut-Node handelt, wird der Textblock nach letzten Child des Parent-Elements des Attributs angehängt.

CDATA anhängen



Strg+D

Diese Funktion steht nur in der Grid-Ansicht zur Verfügung und hängt einen CDATA Node nach dem letzten Geschwisterelement eines ausgewählten Node (mit Ausnahme eines Attribut-Node) an. Falls es sich bei der Auswahl um einen Attribut-Node handelt, wird der CDATA-Abschnitt nach letzten Child des Parent-Elements des ausgewählten Attributs angehängt.

Comment anhängen



Strg+M

Diese Funktion steht nur in der Grid-Ansicht zur Verfügung und hängt einen Comment Node nach dem letzten Geschwisterelement eines ausgewählten Node (mit Ausnahme eines Attribut-Node) an. Falls es sich bei der Auswahl um einen Attribut-Node handelt, wird der Comment Node nach letzten Child des Parent-Elements des ausgewählten Attributs angehängt.

XML anhängen



Mit dieser Funktion fügen Sie eine neue XML-Deklaration `<?xml version="1.0" encoding="UTF-8"?>` als das erste Datenelement in einem Dokument ein. Bitte beachten Sie, dass ein XML-Dokument nur eine XML-Deklaration enthalten darf, welche am Beginn der Datei stehen muss.

Processing Instruction anhängen

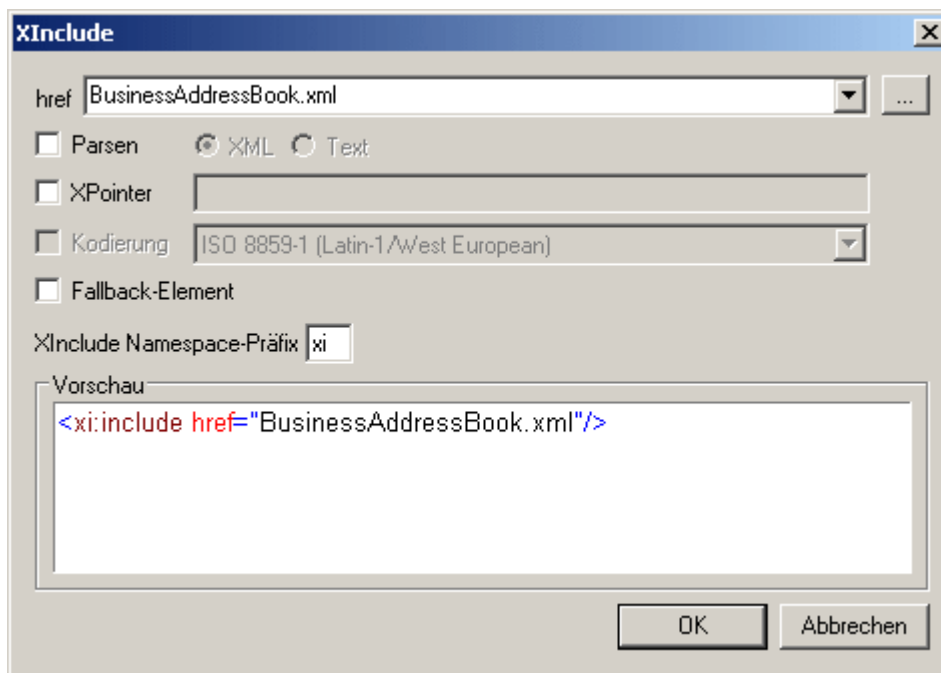


Diese Funktion steht nur in der Grid-Ansicht zur Verfügung und hängt einen Processing Instruction Node nach dem letzten Geschwisterelement eines ausgewählten Node (mit Ausnahme eines Attribut-Node) an. Falls es sich bei der Auswahl um einen Attribut-Node handelt, wird der Processing Instruction Node nach dem letzten Child des Parent-Elements des ausgewählten Attributs angehängt.

XInclude anhängen



Der Befehl **XML | Anhängen | XInclude** steht nur in der Grid-Ansicht zur Verfügung und dient zum Anhängen eines XInclude-Elements hinter dem letzten gleichrangigen Element eines ausgewählten Node (nicht Attribut-Node). Wenn es sich bei der aktuellen Auswahl um ein Attribut handelt, wird das XInclude-Element hinter dem letzten Child-Element des Elements angehängt, das dem ausgewählten Attribut übergeordnet ist. Bei Auswahl dieses Befehls wird das Dialogfeld XInclude (*Abbildung unten*) angezeigt.



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 geparkt 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 Attribut `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-URLs 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 [Attribute mit dem Präfix :xml](#) des Abschnitts Schema-Ansicht.

XPointers

XMLSpy unterstützt XPointers in XInclude. Die relevanten W3C Recommendations dazu sind die [XPointer Framework](#) und die [XPointer element\(\) Scheme](#) 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.

DOCTYPE anhängen



Der Befehl **DOCTYPE anhängen** steht in der Grid-Ansicht einer XML-Datei zur Verfügung, wenn der Node der obersten Ebene ausgewählt ist. Sie hängen damit eine DOCTYPE-Deklaration am Beginn des XML-Dokuments an. Sie müssen den Namen des DOCTYPE angeben und dieser Name muss mit dem Namen des Dokument-Elements übereinstimmen.



Nach Eingabe des DOCTYPE-Namens können Sie die Deklarationen eingeben, die in der internen DTD-Untermenge verwendet werden sollen.

Bitte beachten Sie:

- Die DOCTYPE-Deklaration darf nur zwischen der XML-Deklaration und dem XML-Dokument-Element stehen.
- Sie können stattdessen auch den Befehl [DTD zuweisen](#) verwenden, mit dem Sie eine DOCTYPE-Anweisung erstellen, die auf ein externes DTD-Dokument referenziert.

ExternalID anhängen



Eine DOCTYPE-Deklaration in einer XML-Datei kann eine Referenz auf eine externe Ressource, die eine DTD-Deklaration enthält, beinhalten. Diese Ressource wird entweder über einen Public- oder eine System Identifier referenziert.

Beispiel:

```
<!DOCTYPE doc_element_name PUBLIC "publicID" "systemID">
<!DOCTYPE doc_element_name SYSTEM "systemID">
```

Ein System Identifier ist eine URI, die eine externe Ressource spezifiziert. Der Public Identifier ist ein pfadunabhängiger Identifier, der dazu dient, die Referenz auf eine externe Ressource aufzuheben. So sind z.B. in Ihrem XMLSpy Installationsverzeichnis URIs für häufig verwendete DTDs und XML-Schemas in den verschiedenen Schema-Ordern unter `C:\Program Files\Altova\Common2012\Schemas\` in den Katalogdateien mit dem Namen `catalog.xml` gespeichert. Um die Referenz auf eine in diesen Katalogdateien aufgelistete DTD aufzuheben, können Sie einen Public Identifier in einem XML-Dokument verwenden.

Der Befehl **ExternalID anhängen** steht zur Verfügung, wenn in der Grid-Ansicht in einer XML-Datei ein Child-Datenelement der DOCTYPE-Deklaration ausgewählt ist. Mit diesem Befehl fügen Sie in der Grid-Ansicht eine Zeile für einen externen Identifier (**PUBLIC** oder **SYSTEM**) ein. Sie müssen die Art des Identifiers und seinen Wert angeben.



In der Textansicht, die der oben gezeigten Abbildung aus der Grid-Ansicht entspricht, sieht dies aus wie folgt:

```
<! DOCTYPE OrgChart SYSTEM "orgchart.dtd" [  
  <! ELEMENT name ( #PCDATA) >  
>
```

Bitte beachten Sie: Sie können eine Zeile für eine External-ID als Child hinzufügen, wenn das Datenelement DOCTYPE ausgewählt ist oder Sie können diese einfügen oder anhängen, wenn eines der Child-Elemente des Datenelements DOCTYPE, z.B. die ELEMENT-Deklaration `name` im obigen Beispiel, ausgewählt ist.

Element anhängen



Diese Funktion steht nur in der Grid-Ansicht und nur für DTD-Dokumente zur Verfügung oder, wenn in der DOCTYPE-Deklaration eines XML-Dokuments ein Datenelement ausgewählt ist. Der Befehl hängt eine ELEMENT-Deklaration an die Liste der Deklarationen an.

ATTLIST anhängen



Diese Funktion steht nur in der Grid-Ansicht und nur für DTD-Dokumente zur Verfügung oder, wenn in der DOCTYPE-Deklaration eines XML-Dokuments ein Datenelement ausgewählt ist. Der Befehl hängt eine ATTLIST-Deklaration an die Liste der Deklarationen an.

Entity anhängen



Diese Funktion steht nur in der Grid-Ansicht und nur für DTD-Dokumente zur Verfügung oder, wenn in der DOCTYPE-Deklaration eines XML-Dokuments ein Datenelement ausgewählt ist. Der Befehl hängt eine ENTITY-Deklaration an die Liste der Deklarationen an.

NOTATION anhängen



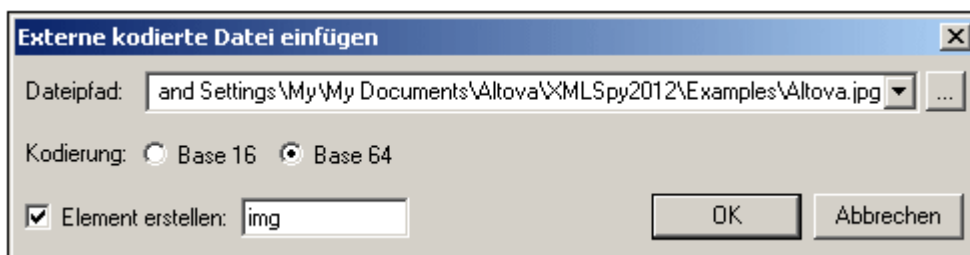
Diese Funktion steht nur in der Grid-Ansicht und nur für DTD-Dokumente zur Verfügung oder, wenn in der DOCTYPE-Deklaration eines XML-Dokuments ein Datenelement ausgewählt ist. Der Befehl hängt eine NOTATION-Deklaration an die Liste der Deklarationen an.

Externe kodierte Datei anhängen

Der Befehl **XML | Anhängen | Externe kodierte Datei** steht nur in der Grid-Ansicht zur Verfügung. Mit diesem Befehl wird eine kodierte binäre Datei wie z.B. eine Bilddatei in Form

kodierter Zeichen angehängt. Die externe kodierte Datei wird nach dem in der Grid-Ansicht ausgewählten Abschnitt angehängt.

Bei Auswahl dieses Befehls wird das Dialogfeld "Externe kodierte Datei einfügen" (*Abbildung unten*) aufgerufen. In dieses Dialogfeld müssen Sie den Pfad zur Datei eingeben, die gewünschte Kodierung auswählen und angeben, ob die kodierte Datei in ein Element eingefügt werden soll oder nicht.

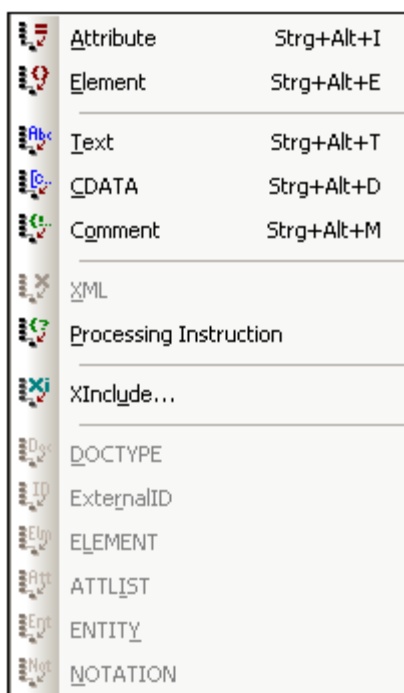


Die kodierte Datei wird in der Grid-Ansicht angehängt.

18.4.3 Child hinzufügen

Der Befehl **Child hinzufügen** ist zwar in allen Menüs aktiv, kann aber nur in der Grid-Ansicht verwendet werden. Der Befehl enthält ein Untermenü (*siehe Screenshot*), über das Sie die folgenden Child-Datenelemente zum aktuell ausgewählten Element hinzufügen können:

- die XML-Deklaration und Node-Typen (Attribut, Element, Text, CDATA, Comment, Processing Instruction) in XML-Dokumenten,
- DOCTYPE-Deklarationen und externe DTD-Deklarationen in XML-Dokumenten und
- DTD-Deklarationen (ELEMENT, ATTLIST, ENTITY und NOTATION) in DTD-Dokumenten und internen DTD-Deklarationen von XML-Dokumenten.



Child-Attribut hinzufügen



Strg+ALT+I

Dieser Befehl steht nur in der Grid-Ansicht und nur bei Auswahl eines Element-Node zur Verfügung. Er fügt ein **neues Attribut** als Child des ausgewählten Element-Node ein.

Child-Element hinzufügen



Strg+ALT+E

Dieser Befehl steht nur in der Grid-Ansicht zur Verfügung. Er fügt ein neues Element als Child des ausgewählten Node ein.

Child-Text hinzufügen



Strg+ALT+T

Dieser Befehl steht nur in der Grid-Ansicht zur Verfügung. Er fügt einen neuen Textinhalt als Child des ausgewählten Datenelements ein.

Child CDATA hinzufügen



Strg+ALT+D

Dieser Befehl steht nur in der Grid-Ansicht zur Verfügung. Er fügt einen neuen CDATA-Abschnitt als Child des ausgewählten Datenelements ein.

Child Comment hinzufügen



Strg+ALT+M

Dieser Befehl steht nur in der Grid-Ansicht zur Verfügung. Er fügt einen neuen Comment Node als Child des ausgewählten Datenelements ein.

Child XML hinzufügen



Dieser Befehl ist nur in der Grid-Ansicht und nur, wenn die Datei **leer** ist, verfügbar. Er fügt eine neue XML-Deklaration `<?xml version="1.0" encoding="UTF-8"?>` als **das erste** Datenelement in ein Dokument ein. Bitte beachten Sie, dass ein XML-Dokument nur eine XML-Deklaration enthalten darf, welche am Beginn der Datei stehen muss.

Child Processing Instruction hinzufügen

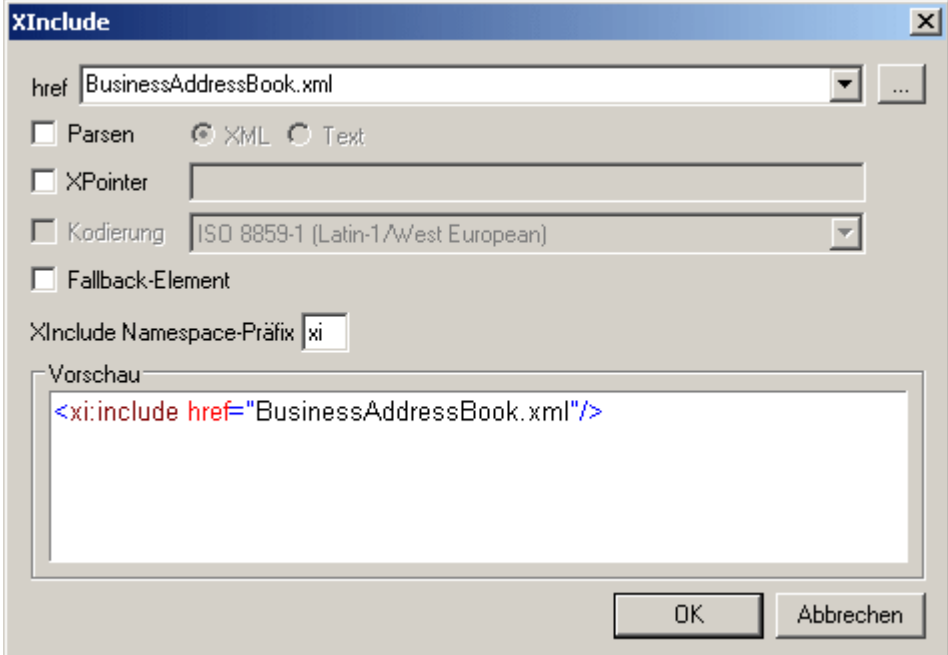


Diese Funktion steht nur in der Grid-Ansicht zur Verfügung und fügt eine neue Processing Instruction (PI) als Child des ausgewählten Datenelements ein.

Child XInclude anhängen



Der Befehl **XML | Child hinzufügen | XInclude** steht nur in der Grid-Ansicht zur Verfügung und fügt ein XInclude-Element als Child des ausgewählten Datenelements ein. Bei Auswahl dieses Befehls wird das Dialogfeld "XInclude" (Abbildung unten) aufgerufen.



The dialog box titled "XInclude" contains the following fields and controls:

- href:** A text field containing "BusinessAddressBook.xml" with a browse button (...).
- Parse:** A checkbox.
- XML/Text:** Radio buttons, with "XML" selected.
- XPointer:** A text field.
- Kodierung:** A dropdown menu showing "ISO 8859-1 (Latin-1/West European)".
- Fallback-Element:** A checkbox.
- XInclude Namespace-Präfix:** A text field containing "xi".
- Vorschau:** A text area showing the XML code: `<xi:include href="BusinessAddressBook.xml"/>`
- Buttons:** "OK" and "Abbrechen" (Cancel).

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 geparkt 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 Attribut `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-URLs 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 [Attribute mit dem Präfix :xml](#) des Abschnitts Schema-Ansicht.

XPointers

XMLSpy unterstützt XPointers in XInclude. Die relevanten W3C Recommendations dazu sind die [XPointer Framework](#) und die [XPointer element\(\) Scheme](#) 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.

Child DOCTYPE hinzufügen

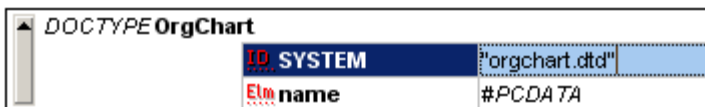


Dieser Befehl steht nur in der Grid-Ansicht und nur bei einem **leeren** Dokument zur Verfügung. Er fügt eine DOCTYPE-Deklaration in ein XML-Dokument ein. Mit Hilfe der DOCTYPE-Deklaration können Sie eine interne DTD-Untermenge deklarieren.

Child ExternalID hinzufügen



Der Befehl **ExternalID hinzufügen** steht nur dann zur Verfügung, wenn in der Grid-Ansicht in einer XML-Datei die DOCTYPE-Deklaration ausgewählt ist. Mit diesem Befehl fügen Sie in der Grid-Ansicht eine Zeile für einen externen Identifier (`PUBLIC` oder `SYSTEM`) ein. Sie müssen die Art des Identifiers und seinen Wert angeben.



In der Textansicht, die der oben gezeigten Abbildung aus der Grid-Ansicht entspricht, sieht dies aus wie folgt:

```
<!DOCTYPE OrgChart SYSTEM "orgchart.dtd" [
  <!ELEMENT name ( #PCDATA) >
]>
```

Bitte beachten Sie: Sie können eine Zeile für eine External-ID als Child hinzufügen, wenn das Datenelement DOCTYPE ausgewählt ist oder Sie können diese einfügen oder anhängen, wenn eines der Child-Elemente des Datenelements DOCTYPE, z.B. die ELEMENT-Deklaration `name` im obigen Beispiel, ausgewählt ist.

Child Element hinzufügen



Diese Funktion steht nur in der Grid-Ansicht und nur für DTD-Dokumente zur Verfügung oder, wenn die DOCTYPE-Deklaration eines XML-Dokuments ausgewählt ist. Der Befehl hängt eine ELEMENT-Deklaration an die Liste der Deklarationen an.

Child ATTLIST hinzufügen



Diese Funktion steht nur in der Grid-Ansicht und nur für DTD-Dokumente zur Verfügung oder, wenn die DOCTYPE-Deklaration eines XML-Dokuments ausgewählt ist. Der Befehl hängt eine ATTLIST-Deklaration an die Liste der Deklarationen an.

Child ENTITY hinzufügen



Diese Funktion steht nur in der Grid-Ansicht und nur für DTD-Dokumente zur Verfügung oder, wenn die DOCTYPE-Deklaration eines XML-Dokuments ausgewählt ist. Der Befehl hängt eine ENTITY-Deklaration an die Liste der Deklarationen an.

Child NOTATION hinzufügen

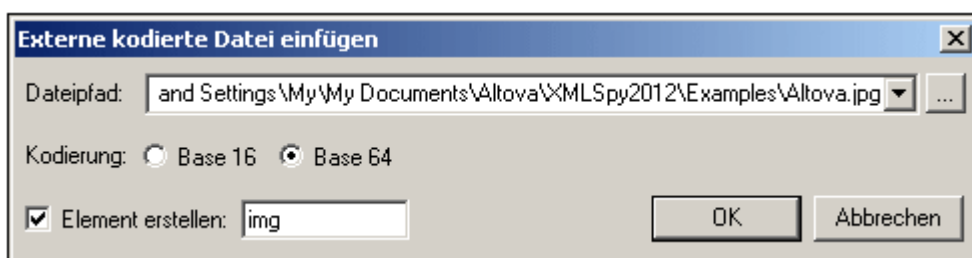


Diese Funktion steht nur in der Grid-Ansicht und nur für DTD-Dokumente zur Verfügung oder, wenn die DOCTYPE-Deklaration eines XML-Dokuments ausgewählt ist. Der Befehl hängt eine NOTATION-Deklaration an die Liste der Deklarationen an.

Externe kodierte Child-Datei hinzufügen

Der Befehl **XML | Child hinzufügen | Externe kodierte Datei** steht nur in der Grid-Ansicht zur Verfügung. Mit diesem Befehl wird eine kodierte binäre Datei wie z.B. eine Bilddatei in Form kodierter Zeichen hinzugefügt. Die externe kodierte Datei wird als Child des in der Grid-Ansicht ausgewählten Abschnitts hinzugefügt.

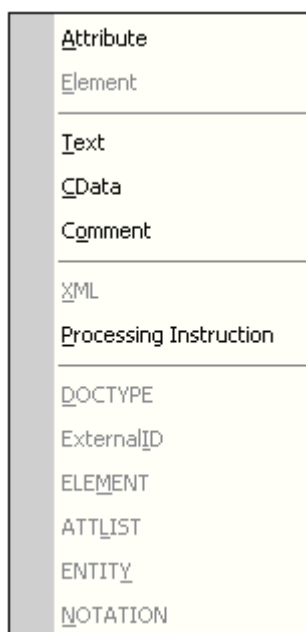
Bei Auswahl dieses Befehls wird das Dialogfeld "Externe kodierte Datei einfügen" (*Abbildung unten*) aufgerufen. In dieses Dialogfeld müssen Sie den Pfad zur Datei eingeben, die gewünschte Kodierung auswählen und angeben, ob die kodierte Datei in ein Element eingefügt werden soll oder nicht.



Die kodierte Datei in der Grid-Ansicht wird als Child hinzugefügt.

18.4.4 Konvertieren in

Mit diesem Befehl können Sie ein ausgewähltes Datenelement in der Grid-Ansicht in einen anderen Datenelement-Typ konvertieren. Diese Operation steht nur in der Grid-Ansicht für einzelne Datenelemente zur Verfügung, die keinen Child Node enthalten. Wenn Sie den Cursor über den Befehl **Konvertieren in** platzieren, wird ein Untermenü (*siehe Screenshot*) mit den Datenelementen angezeigt, in die das ausgewählte Datenelement konvertiert werden kann.



Bitte beachten Sie: Gehen durch die Konvertierung möglicherweise Daten verloren (z.B. geht beim Konvertieren eines Attributs in einen Comment der Attributname verloren), werden Sie mittels einer Meldung darüber informiert.

In Attribut konvertieren

Das ausgewählte Datenelement wird in ein **Attribut** konvertiert.

In Element konvertieren

Das ausgewählte Datenelement wird in ein **Element** konvertiert.

In Text konvertieren

Das ausgewählte Datenelement wird in **Text** konvertiert.

In CDATA konvertieren

Das ausgewählte Datenelement wird in einen **CDATA Block** konvertiert.

In Comment konvertieren

Das ausgewählte Datenelement wird in einen **Comment** konvertiert.

In XML konvertieren

Mit dieser Funktion konvertieren Sie ein ausgewähltes Datenelement in eine XML Deklaration

```
<?xml version="1.0" encoding="UTF-8"?>.
```

Bitte beachten Sie: Jedes XML-Dokument darf nur eine XML-Deklaration enthalten, die am Beginn der Datei stehen muss.

In Processing Instruction konvertieren

Das ausgewählte Datenelement wird in eine neue **Processing Instruction** (PI) konvertiert.

In DOCTYPE konvertieren

Mit diesem Befehl konvertieren Sie das ausgewählte Datenelement in eine DOCTYPE-Deklaration (in einer XML-Datei).

Bitte beachten Sie: Die DOCTYPE-Deklaration darf nur am Beginn eines XML-Instanzdokuments zwischen der XML-Deklaration und dem Dokument-Element des XML-Dokuments stehen.

In ExternalID konvertieren

Das ausgewählte Datenelement wird in eine externe DTD-Referenz in einer DOCTYPE-Deklaration (**PUBLIC** oder **SYSTEM** Identifier) konvertiert.

In Element konvertieren

Das ausgewählte Datenelement wird in eine Element-Deklaration in einer DOCTYPE-Deklaration oder einer externen DTD konvertiert.

In ATTLIST konvertieren

Das ausgewählte Datenelement wird in eine Attributlisten-Deklaration in einer DOCTYPE Deklaration oder einer externen DTD konvertiert.

In ENTITY konvertieren

Das ausgewählte Datenelement wird in eine Entity-Deklaration in einer DOCTYPE Deklaration oder einer externen DTD konvertiert.

In NOTATION konvertieren

Das ausgewählte Datenelement wird in eine Notationsdeklaration in einer DOCTYPE Deklaration oder einer externen DTD konvertiert.

18.4.5 Tabelle

Der Befehl **Tabelle** ist zwar in allen Ansichten aktiv, kann aber nur in der Grid-Ansicht verwendet werden. Er enthält ein Untermenü mit allen für die [Datenbank-/Tabellenansicht](#) der Grid-Ansicht relevanten Befehlen.

	Als Tabelle anzeigen	F12
	Zeile einfügen	Umschalt+F12
	Zeile anhängen	Strg+F12
	Aufsteigend sortieren	
	Absteigend sortieren	

Als Tabelle anzeigen

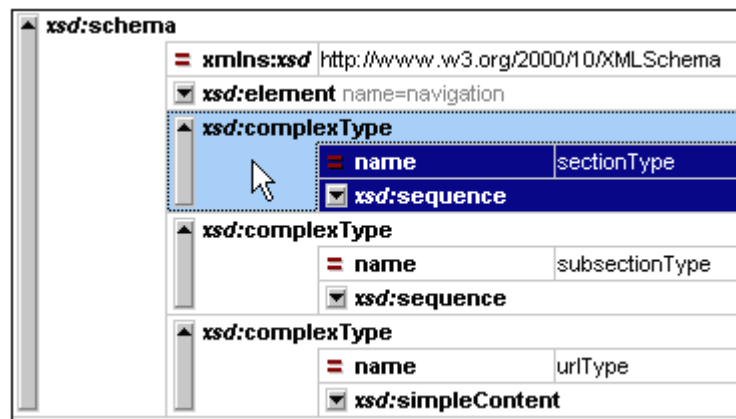


F12

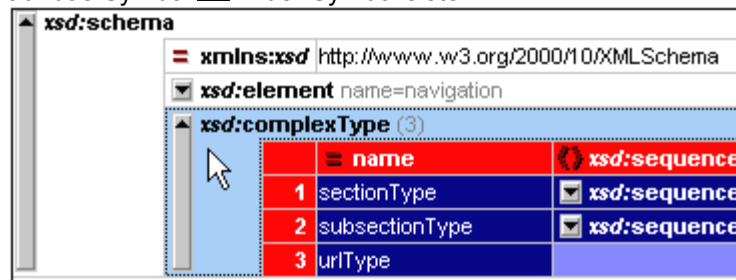
Mit diesem Befehl können Sie zwischen der [Grid-Ansicht](#) und der [Datenbank-/Tabellen-Ansicht](#) (oder Tabellenansicht) eines Dokument-Elements wechseln. In der Tabellenansicht werden sich wiederholende Elemente als Tabelle angezeigt, wobei die Zeilen die Instanzen enthalten und die Spalten die Child Nodes (einschl. Comments, CDATA-Abschnitte und PIs).

So wechseln Sie zur Tabellenansicht:

1. Markieren Sie ein sich wiederholendes Element, das in Tabellenform angezeigt werden soll.



2. Wählen Sie **XML | Tabelle | Als Tabelle anzeigen**, drücken Sie **F12** oder klicken Sie auf das Symbol  in der Symbolleiste.



Das Element wird als Tabelle dargestellt und das Symbol  ist aktiv.

Um von der Tabellenansicht eines Dokument-Elements zur normalen Grid-Ansicht dieses Elements zu wechseln, wählen Sie die Tabelle oder eine ihrer Zeilen oder Spalten aus und klicken Sie in der Symbolleiste auf das Symbol . Daraufhin wird das Tabellenelement in der Grid-Ansicht angezeigt.

Anmerkung: Die Farben der Tabellenansicht können im Dialogfeld "Optionen" auf dem Register "Farben" (**Extras | Optionen | Farben**) definiert werden.

Zeile einfügen



Umschalt+F12

Der Befehl **Zeile einfügen** ist in der [Datenbank-/Tabellen-Ansicht](#) aktiv, wenn eine Zeile oder Zelle ausgewählt ist. Der Befehl fügt eine neue Zeile vor der markierten Zeile ein. Die neue Zeile entspricht einer Instanz des Tabellenelements. Für das neue Element werden obligatorische Child-Elemente erstellt.

Zeile anhängen



Strg+F12

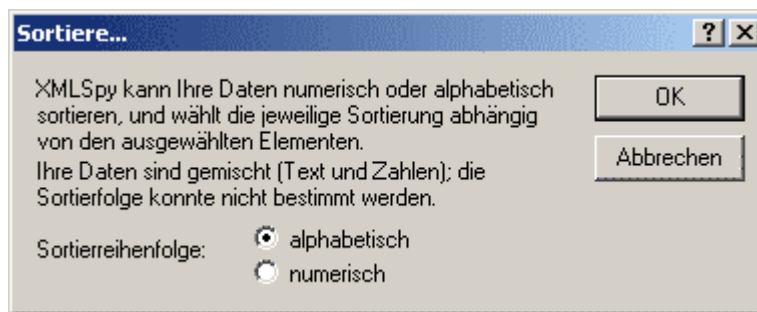
Der Befehl **Zeile anhängen** ist in der [Datenbank-/Tabellen-Ansicht](#) aktiv, wenn eine Zeile oder Zelle ausgewählt ist. Der Befehl hängt eine neue Zeile nach der letzten Zeile der Tabelle an. Die neue Zeile entspricht einer Instanz des Tabellenelements. Für das neue Element werden

obligatorische Child-Elemente erstellt.

Aufsteigend sortieren



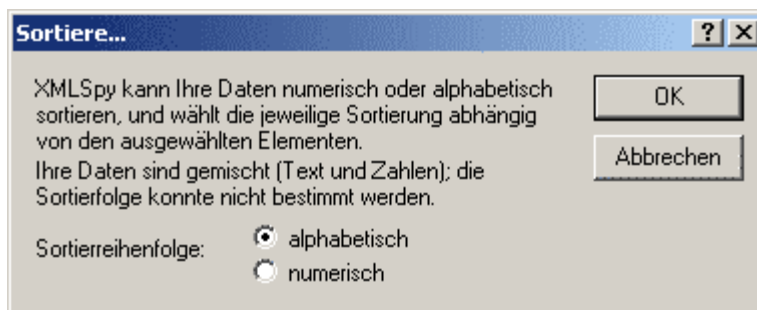
Der Befehl **Aufsteigend sortieren** ist in der [Datenbank-/Tabellen-Ansicht](#) aktiv, wenn eine Spalte oder Zelle ausgewählt ist. Der Befehl sortiert die Spalte entweder alphabetisch oder numerisch in aufsteigender Reihenfolge. XMLSpy versucht automatisch zu ermitteln, um welche Art von Daten es sich handelt und sortiert diese alphabetisch oder numerisch. Wenn vom System nicht erkannt wird, um welche Art es sich handelt, werden Sie aufgefordert die Sortiermethode anzugeben (*siehe Abbildung unten*).



Absteigend sortieren



Der Befehl **Absteigend sortieren** ist in der [Datenbank-/Tabellen-Ansicht](#) aktiv, wenn eine Spalte oder Zelle ausgewählt ist. Der Befehl sortiert die Spalte entweder alphabetisch oder numerisch in absteigender Reihenfolge. XMLSpy versucht automatisch zu ermitteln, um welche Art von Daten es sich handelt und sortiert diese alphabetisch oder numerisch. Wenn vom System nicht erkannt wird, um welche Art es sich handelt, werden Sie aufgefordert die Sortiermethode anzugeben (*siehe Abbildung unten*).



18.4.6 Höher stufen



Strg+L

Der Befehl **Höher stufen** steht nur in der Grid-Ansicht zur Verfügung. Sie verschieben damit

den markierten Node um eine Ebene nach links, d.h. ein Child-Element wird hierdurch zu einem Geschwister-Element seines Parent. Dieser Befehl wird auch "**Promote**" genannt.

18.4.7 Tiefer stufen



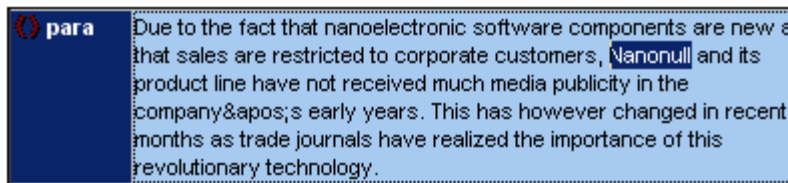
Strg+R

Der Befehl **Tiefer stufen** steht nur in der Grid-Ansicht zur Verfügung. Sie verschieben damit den markierten Node um eine Ebene nach rechts, sodass er hierdurch zu einem Child-Element seines ehemaligen Geschwister-Element wird. Dieser Befehl wird auch "**Demote**" genannt.

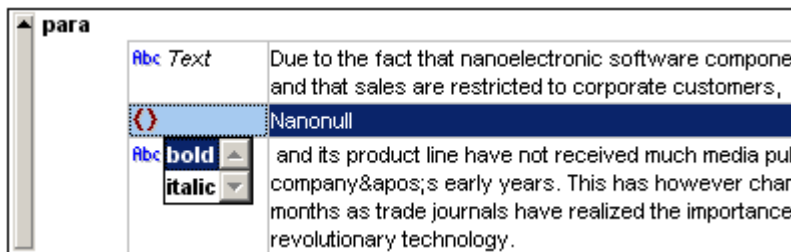
18.4.8 In Element einbinden

Der Befehl **In Element einbinden** steht nur in der Grid-Ansicht zur Verfügung. Er bindet einen markierten Textbereich in ein neues Element ein. Das neue Element wird innerhalb des Textbereichs rund um den markierten Text angelegt. Wenn Sie ein Dokument auf Basis eines Schemas oder einer DTD bearbeiten, steht Ihnen automatisch eine Liste mit gültigen Auswahlmöglichkeiten für den Namen des Elements, in das der Text eingeschlossen werden soll, zur Verfügung.

In der unten gezeigten Abbildung ist z.B. der Text `Nanonull` im Element `para` markiert.



Wenn Sie den Befehl **XML | In Element einbinden** auswählen, wird der Text "Nanonull" in ein neues Element eingeschlossen, das innerhalb des Texts erstellt wird. Es wird eine Liste angezeigt, aus der Sie für den Elementnamen `bold` oder `italic` auswählen können. Diese Elemente werden im Schema als Child-Elemente von `para` definiert.



Der gewählte Eintrag wird als Name des neuen Elements verwendet. Alternativ dazu können Sie auch einen anderen Namen für das Element eingeben.

18.4.9 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](#). Auf dem Register "XPath" können Sie einen XPath-Ausdruck im aktiven Dokument auswerten und

das Ergebnis im Fenster "Ausgabe" anzeigen.

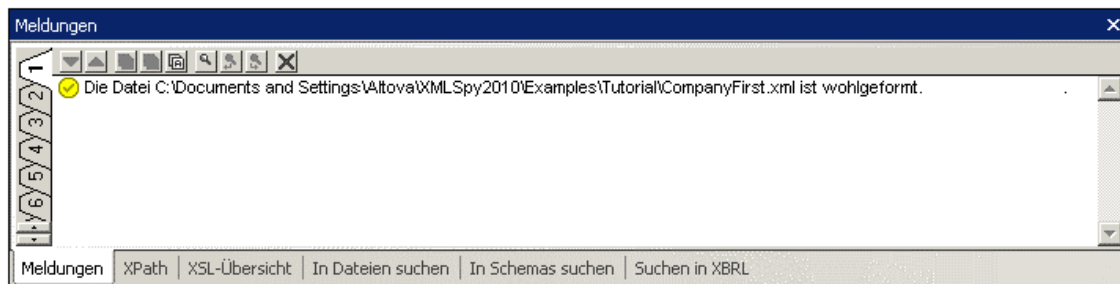
18.4.10 Wohlgeformtheit prüfen



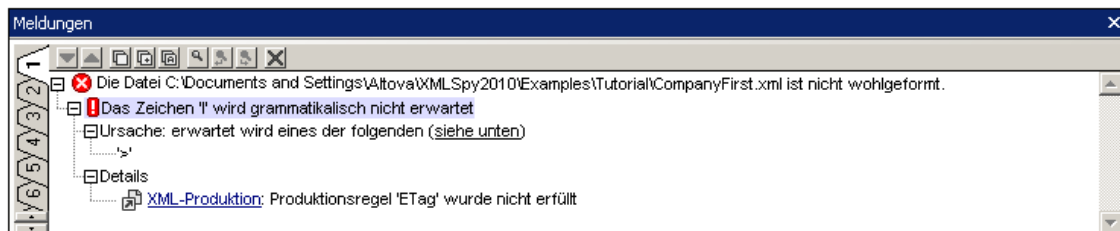
F7

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:



Wenn bei der Wohlgeformtheitsprüfung ein Fehler entdeckt wurde, wird das Fehler verursachende Objekt markiert und eine entsprechende Fehlermeldung am unteren Rand des Hauptfensters angezeigt.



Beachten Sie, dass das Fenster "Meldungen" neun Ausgaberegister enthält. Das Ergebnis der Validierung wird immer auf dem aktuellen Register angezeigt. Sie können daher die Wohlgeformtheit für eine Schemadatei auf Register 1 überprüfen und das Ergebnis aufbewahren, indem Sie zu Register 2 wechseln, bevor Sie das nächste Schema-Dokument validieren (andernfalls wird Register 1 durch das Validierungsergebnis überschrieben.)

Eigentlich ist es nicht erlaubt, ein schlecht geformtes XML-Dokument zu speichern; XMLSpy enthält die Funktion "Save anyway" (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.

Bitte beachten Sie: Sie können den Befehl **Wohlgeformtheit prüfen** auch an jeder Datei, jedem Ordner oder jeder Gruppe von Dateien im aktiven [Projektfenster](#) anwenden. Klicken Sie auf den entsprechenden Eintrag, und anschließend auf das Symbol "Wohlgeformtheit prüfen".

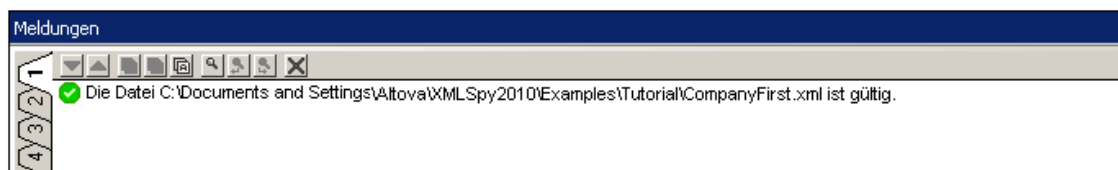
18.4.11 XML validieren



F8

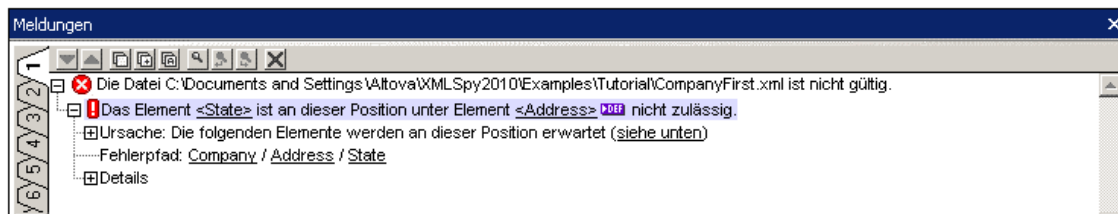
Mit diesem Befehl können Sie ein XML-Dokument gegen DTDs, XML-Schemas und andere Schemas validieren. Die Validierung wird automatisch durchgeführt, wenn Sie von der Textansicht in eine andere Ansicht wechseln. Sie können einstellen, dass ein Dokument beim Öffnen oder Speichern automatisch validiert wird (**Extras | Optionen | Datei**). Der Befehl **Validieren** führt vor Überprüfung der Gültigkeit gleichzeitig eine Wohlgeformtheitsprüfung durch, sodass diese vorher nicht separat überprüft werden muss.

Bei erfolgreicher Validierung wird am unteren Rand des Hauptfensters folgende Meldung angezeigt:



Wenn bei der Validierung ein Fehler entdeckt wurde, wird das Fehler verursachende Objekt markiert, und eine entsprechende Fehlermeldung wird angezeigt:

Bitte beachten Sie: Wenn Sie auf die Links in der Fehlermeldung klicken, können Sie zu der Stelle in der XML-Datei springen, an der der Fehler gefunden wurde.



Bitte beachten Sie: Das Validierungsfenster enthält neun Ausgaberegister. Das Ergebnis der Validierung wird immer auf dem aktuellen Register angezeigt. Sie können daher die Wohlgeformtheit für eine Schemadatei auf Register 1 überprüfen und das Ergebnis aufbewahren, indem Sie zu Register 2 wechseln, bevor Sie das nächste Schema-Dokument validieren (andernfalls wird Register 1 durch das Validierungsergebnis überschrieben.)

Der Befehl wird normalerweise am aktiven Dokument durchgeführt. Sie können die Validierung über Dateien, Ordner oder Dateigruppen laufen lassen, die im aktiven Projektfenster angezeigt werden. Klicken Sie im Projektfenster auf die gewünschte Datei/den gewünschten Ordner und wählen Sie die Option **XML | Validieren** oder **F8**. Ungültige Dateien werden als aktive Datei im Hauptfenster geöffnet und die entsprechende Fehlermeldung wird angezeigt.

Validieren von XML-Dokumenten

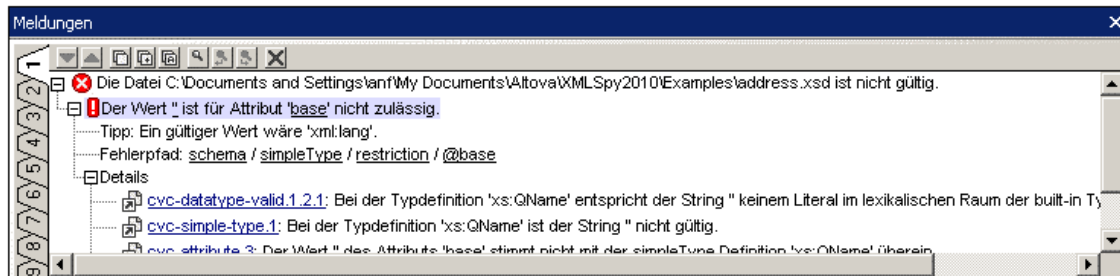
Um eine XML-Datei zu validieren, machen Sie das XML-Dokument im Hauptfenster zum aktiven und klicken Sie auf **XML | Validieren** oder drücken Sie **F8**. Das XML-Dokument wird daraufhin anhand des in der XML-Datei referenzierten Schemas validiert. Existiert keine Referenz, wird im Fenster "Meldungen" eine Fehlermeldung angezeigt. Solange das XML-Dokument geöffnet ist, bleibt das Schema im Speicher (siehe [Cache-Speicher leeren](#) 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**.

Schema-Validierung im Fenster "Validierung"

Wenn das Schema (DTD oder XML-Schema) gültig ist, wird im Validierungsfenster eine entsprechende Meldung angezeigt; das Validierungsfenster wird unterhalb vom Applikationsfenster eingeblendet. Wenn das Schema nicht gültig ist, werden eine oder mehrere Fehlermeldungen im Validierungsfenster angezeigt (*Abbildung unten*).



Die Fehlermeldung ist in drei Teile gegliedert:

1. Eine Beschreibung des Fehlers, die Links zu den entsprechenden Deklarationen oder Definitionen enthält.
2. Wo in der Schemastruktur sich der Fehler befindet. Wenn Sie in der Pfadangabe auf einen Node klicken, wird dieser Node im Dokument markiert angezeigt.
3. Unter "Details" werden nähere Informationen zum Fehler angezeigt. Dieser Teil enthält einen Link zum entsprechenden Absatz in der jeweiligen Spezifikation (der XML-Spezifikation für DTD-Validierung; und der XML-Schema-Spezifikation für XML-Schema-Validierung).

Bitte beachten Sie: Wenn Sie bei Ausführung der Validierung in der Text-Ansicht im Validierungsfenster auf einen Link klicken, wird die entsprechende Deklaration oder Definition in der Text-Ansicht markiert. Wenn die Validierung in der Schema-Ansicht durchgeführt wird, wird die entsprechende Deklaration oder Definition entweder im Design-Fenster der Schema-Ansicht oder in der jeweiligen Eingabehilfe markiert.

Kataloge

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 nur in den Katalogdateien geändert werden müssen). Der Mechanismus funktioniert folgendermaßen:

- XMLSpy lädt eine Datei namens `RootCatalog.xml`, die eine Liste von Katalogdateien enthält, die durchsucht werden. Sie können jeweils im Element `nextCatalog` in `RootCatalog.xml` beliebig viele Katalogdateien definieren, die durchsucht werden sollen.
- Die in `RootCatalog.xml` inkudierten Katalogdateien werden durchsucht und die URIs werden entsprechend der Zuordnungen in den Katalogdateien aufgelöst. Achten Sie darauf, keine Zuordnungen doppelt zu vergeben, da dies Fehler verursachen könnte.
- Mit XMLSpy werden zwei Katalogdateien installiert. Eine Beschreibung der Funktionsweise dieser Kataloge finden Sie im Abschnitt [Kataloge in XMLSpy](#).
- Der `PUBLIC` oder der `SYSTEM` Identifier in der `DOCTYPE`-Anweisung Ihrer XML-Datei wird für das Durchsuchen des Katalogs verwendet. Bei häufig verwendeten Schemas ist der `PUBLIC` Identifier normalerweise bereits vordefiniert, sodass nur mehr die URI in

den Katalog-Lookup-Dateien geändert werden muss, wenn XML-Dokumente auf mehreren Rechnern verwendet werden.

Wenn Sie Ihre eigene `CustomCatalog.xml` (oder eine andere benutzerdefinierte Katalogdatei) schreiben, verwenden Sie nur die folgende Teilmenge des OASIS-Katalogs, damit XMLSpy den Katalog korrekt verarbeiten kann. Jedes der Elemente in der unterstützten Teilmenge kann das Attribut `xml:base` erhalten, welches dazu dient, die Basis-URI dieses Elements zu definieren.

```
<catalog...>
...
<public publicId="PublicID of Resource" uri="URL of local file"/>
<system systemId="SystemID of Resource" uri="URL of local file"/>
<rewriteURI uriStartString="StartString of URI to rewrite"
rewritePrefix="String to replace StartString"/>
<rewriteSystem systemIdStartString="StartString of SystemID"
rewritePrefix="Replacement string to locate resource locally"/>
<uri name="filename" uri="URL of file identified by filename"/>
...
</catalog>
```

Bitte beachten Sie:

- Die Datei `catalog.xml` im Ordner `%AltovaCommonFolder%\Schemas\schemas` enthält Referenzen auf DTDs, die ältere XML-Schema-Spezifikationen implementieren. Sie sollten Ihre XML-Schema-Dokumente nicht anhand eines dieser Schemas validieren. Die referenzierten DTD-Dateien wurden nur inkludiert, um XMLSpy Eingabehilfeninformationen zu Bearbeitungszwecken bereitzustellen, falls Sie Dateien gemäß diesen älteren Empfehlungen erstellen wollen. *siehe auch nächster Punkt*
- Wenn Sie eine benutzerdefinierte Dateierweiterung für ein bestimmtes Schema erstellen (z.B. die Erweiterung `.myhtml` für (HTML)-Dateien, die gemäß der HTML-DTD gültig sein sollen), können Sie die intelligente Editierung für Dateien mit dieser Erweiterung aktivieren, indem Sie eine Textzeile zu `CustomCatalog.xml` hinzufügen. Für das erwähnte Beispiel sollten Sie das Element `<spy:fileExtHelper ext="myhtml" uri="schemas/xhtml1/xhtml1-transitional.dtd"/>` als Child des Elements `<catalog>` hinzufügen, damit die intelligente Editierung (Autokomplettierung, Eingabehilfen usw.) von `.myhtml`-Dateien in XMLSpy gemäß der XHTML 1.0 Transitional DTD aktiviert werden kann.
- Nähere Informationen zu Katalogen finden Sie in der [XML Catalogs-Spezifikation](#).

Automatisches Validieren mit Altova XML 2012

Altova XML ist eine kostenlose Applikation, die den Altova XML-Validator sowie einen XSLT 1.0-, XSLT 2.0- und XQuery 1.0-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 1.0 und 2.0 Stylesheets sowie zum Ausführen von XQuery-Dokumenten verwendet werden.

Validierungen können daher mittels Altova XML automatisiert werden. So können Sie z.B. eine Batch-Datei erstellen, die Altova XML aufruft, um damit eine Reihe von XML-Dokumenten zu validieren und die Ausgabe an eine Textdatei zu senden. Nähere Informationen dazu finden Sie in der [Altova XML Dokumentation](#).

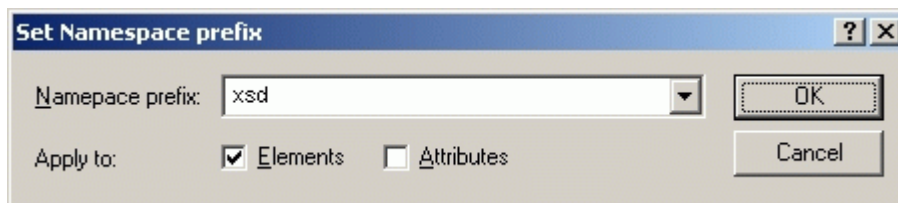
18.4.12 Eingabehilfen aktualisieren



Mit diesem Befehl 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, das Schema oder die DTD erneut zu laden, damit die intelligenten Editierinformationen im XML-Dokument aktualisiert werden.

18.4.13 Namespace Präfix

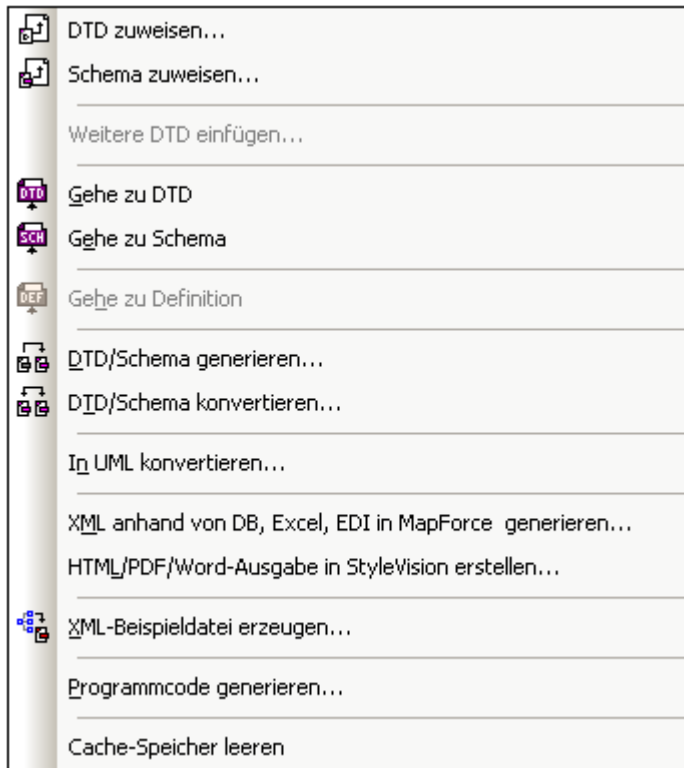
Dieser Befehl steht in der Grid-Ansicht zur Verfügung und öffnet ein Dialogfeld, in dem Sie das Namespace Präfix des ausgewählten Elements oder Attributs - und im Fall von Elementen - seiner Nachfahren definieren können.



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.

18.5 Menü "DTD/Schema"

Das **DTD/Schema**-Menü enthält Befehle für das effiziente Arbeiten mit DTDs und XML-Schemas.



Dieser Abschnitt enthält eine umfassende Beschreibung aller Befehle in diesem Menü.

18.5.1 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**) 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.

18.5.2 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**) 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"
```

Die mit XMLSpy erzeugten Deklarationen hängen vom Schematyp und von der potentiellen Verwendung eines targetNamespace im Schemadokument ab.

18.5.3 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.

18.5.4 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.

18.5.5 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.

18.5.6 Gehe zu Definition



Mit diesem Befehl 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 im Menü **DTD/Schema | Gehe zu Definition**, oder klicken Sie auf das oben abgebildete 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.

18.5.7 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.

DTD/Schema generieren [?] [X]

DTD/Schemaformat

☐ DTD

☒ W3C Schema

OK

Abbrechen

☒ Für alle gleichnamigen Elemente einen gemeinsamen Typ erzeugen

☒ Entities validieren und auflösen

Für Elemente verwendeten Typ definieren

☒ Lokal (falls anwendbar)

☐ Global

Für Attribute verwendete SimpleTypes definieren

☐ Global, gleiche Typen zusammenlegen

☐ Als unterschiedliche globale Typen

☒ Lokal

Attribute mit demselben Namen und Typ definieren

☒ Lokal

☐ Global

SimpleType-Erkennung

☐ Bestmögliche

☐ Nur Zahlen

☒ Keine Erkennung

Enumerationen erstellen für

☒ Alle Arten von Werten

☐ Nur einfache Strings

☒ Immer

☐ Für maximal unterschiedliche Werte

Werte mit mehr als Zeichen bei Enumerationen ignorieren

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. geparte 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.

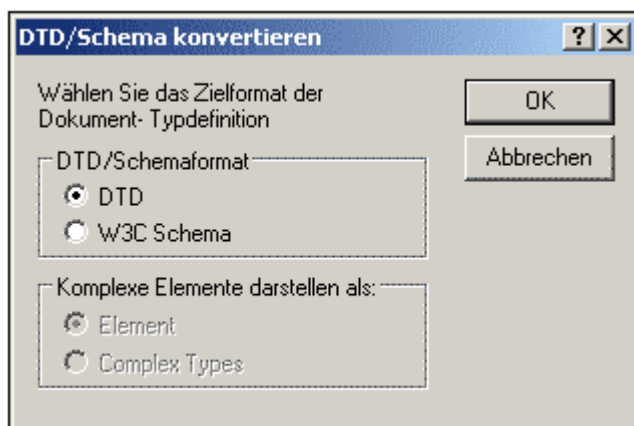
Bei Generierung einer DTD stehen die Optionen für die Entity-Auflösung und Enumerationen zur Verfügung.

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.

18.5.8 DTD/Schema konvertieren



Mit diesem Befehl konvertieren Sie eine existierende DTD in ein XML-Schema und umgekehrt. Bei Auswahl der Option "W3C Schema" müssen Sie außerdem angeben, wie Complex-Elemente dargestellt werden sollen - als Elemente oder complexTypes.



Konvertieren einer DTD in ein XML-Schema

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 geparkt 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. 44 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 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:

1. Eine Notation wird aus einem Namen und einem Identifier bestehend deklariert, z.B.

```
<! NOTATION gif SYSTEM "image/gif">
```
2. Sie deklarieren die Entity, z.B.

```
<! ENTITY cover_img SYSTEM "graphics/cover_img.gif" NDATA gif>
```
3. 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:

1. 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.
2. 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 mit Hilfe des Befehls "DTD/Schema konvertieren..." 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
`<! 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.

Bitte beachten Sie: Gemäß der XML-Schemaspezifikation, ist es nicht notwendig - bzw. je nach Sichtweise, nicht möglich - eine externe Entity zu deklarieren.

Konvertieren eines XML-Schemas in eine DTD

Bei der Konvertierung eines XML-Schemas in eine DTD werden die im XML-Schema verwendeten **Namespace-Präfixe** - nicht die Namespace URLs oder die Namespace-Deklarationen - in die Namen der entsprechenden Elemente und Attribute in der DTD übernommen. Beachten Sie die folgenden Punkte:

1. Da XML-Parser Namespaces beim Validieren eines Dokuments gegen eine DTD ignorieren, werden die Namespace-Deklarationen selbst nicht konvertiert.
2. 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.
3. 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).
4. 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.

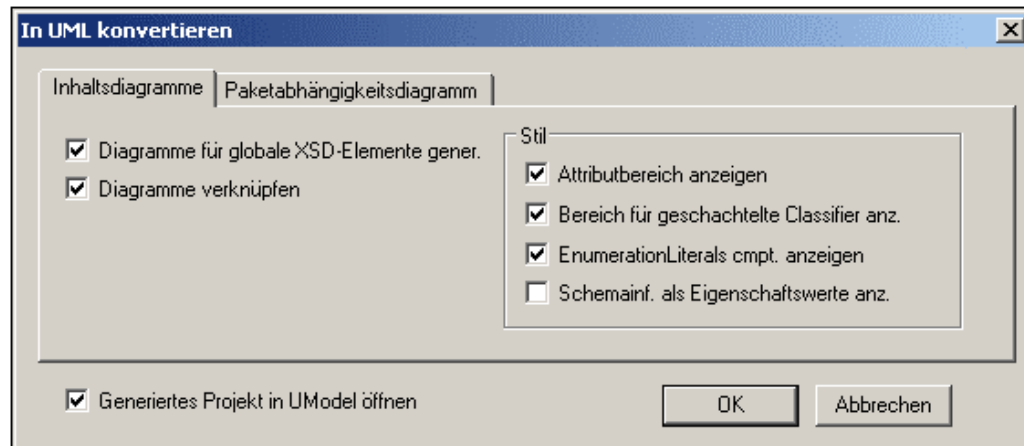
Bitte beachten Sie: Namespaces haben keinen semantischen Wert in einer DTD. Namespace-Präfixe, die aus dem XML-Schema übernommen werden, werden einfach nur Teil des Namens des in der DTD definierten Elements oder Attributs.

18.5.9 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-Modellierungssaplikation 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 "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.

18.5.10 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.

18.5.11 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.

18.5.12 XML-Beispieldatei erzeugen

Dieser Befehl erstellt eine XML-Datei, die auf dem aktiven Schema (DTD oder XML-Schema) im Hauptfenster basiert.

Zu generierende Elemente

Sie haben eine der folgenden Auswahlmöglichkeiten: (i) Anzahl der Elemente minimieren. Hierbei wird nur die minimale Anzahl der zum Erstellen der gültigen Datei erforderlichen Elemente generiert; (ii) nur obligatorische Elemente; (iii) alle Elemente generieren (obligatorische und nicht obligatorische). Wenn Elemente als wiederholt vorkommende definiert sind, kann (für jedes diese wiederholt vorkommenden Elemente) die Anzahl der zu generierenden Elemente definiert werden.

Nicht obligatorische Attribute erzeugen

Beim Aktivieren dieser Option werden alle obligatorischen und optionalen Attribute, die im Schema definiert sind, generiert.

Erzeuge "X" Element(e), wenn als wiederholend definiert (Schema/DTD)

Beim Aktivieren dieser Option wird die im Textfeld eingegebene Anzahl von sich wiederholenden Elementen generiert.

Elemente und Attribute mit Daten befüllen

Beim Aktivieren dieser Option werden die Datentype-Deskriptoren/Werte der jeweiligen Elemente/Attribute eingefügt. Z.B.: Boolean = 1, xsd:string = string, Max/Min

inklusive = der im Schema definierte Wert.

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"](#) 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.

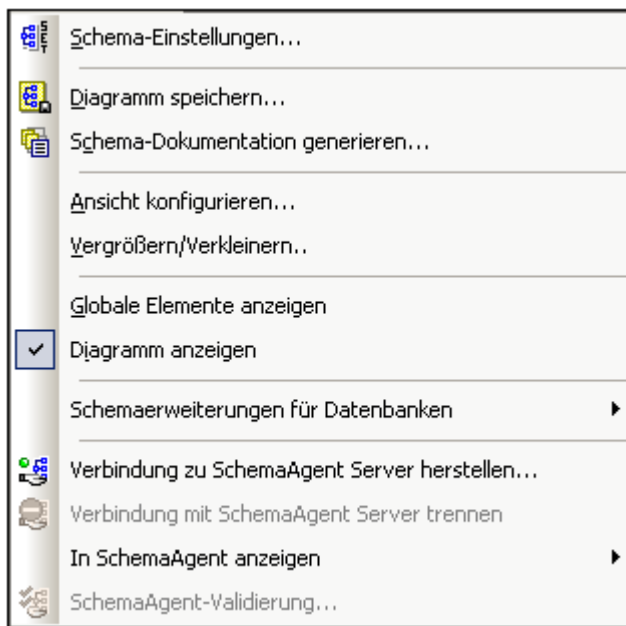
18.5.13 Cache-Speicher leeren

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](#) 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.

18.6 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.

18.6.1 Schema-Einstellungen



Mit diesem Befehl definieren Sie globale Einstellungen für das aktive Schema. Dabei handelt es sich um Attribute und deren Werte für das XML-Schema-Dokument-Element `xs:schema`.

Schema-Einstellungen

Default Element Form: ☒ Qualified ☐ Unqualified
Default Attribute Form: ☐ Qualified ☒ Unqualified

Block default: extension
Final default: restriction
Version: 3.5a
xml:lang: ID:

☐ Kein Target Namespace
☒ Target Namespace http://www.xmlspy.com/schemas/orgchart

Prefix	Namespace
	http://www.xmlspy.com/schemas/orgchart
xs	http://www.w3.org/2001/XMLSchema
ipo	http://www.altova.com/IPO
ts	http://www.xmlspy.com/schemas/textstate

OK Abbrechen

In diesem Dialogfeld können Sie die folgenden Einstellungen vornehmen:

- Sie können die Attribute `elementFormDefault` und `attributeFormDefault` des `xs:schema`-Elements jeweils auf `qualified` oder `unqualified` setzen.
- Weiters können Sie die Attribute `blockDefault`, `finalDefault` and `version` durch Eingabe der benötigten Werte in das jeweilige Textfeld definieren.
- Um einen Target Namespace für das XML-Instanzdokument zu definieren, aktivieren Sie das Target Namespace-Optionsfeld und geben Sie den Target Namespace in das Textfeld ein. Bei Deklaration eines Target Namespace müssen Sie diesen Namespace im Schema-Dokument durch Eingabe einer Namespace-Zeile im unteren Bereich definieren. Sie können für diesen Namespace ein Präfix definieren oder ihn als Default Namespace verwenden (indem Sie das Präfix-Feld wie in der obigen Abbildung gezeigt leer lassen).

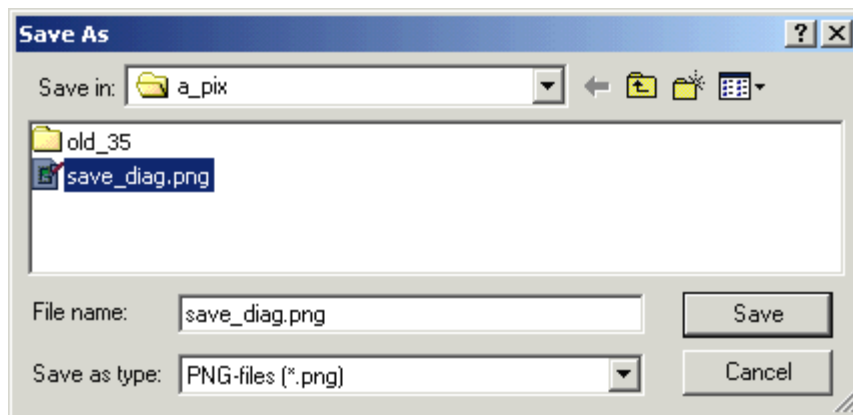
In der Textansicht sehen die obigen Schema-Einstellungen etwa folgendermaßen aus:


```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema
targetNamespace="http://www.xmlspy.com/schemas/Altova/orgchart" xmlns="http://www.xmlspy.com/schemas/Altova/orgchart"
xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"
elementFormDefault="unqualified"
attributeFormDefault="unqualified" blockDefault="extension"
finalDefault="restriction" version="3.5a">
  <xsd:notation name="Altova-Orgchart"
public="http://www.xmlspy.com/schemas/Altova/orgchart"/>
  <xsd:complexType name="DivisionType">
    <xsd:sequence>
      <xsd:element name="Name" type="xsd:string">
        <xsd:annotation>
          <xsd:documentation>Division Name
```

Bitte beachten Sie: Diese Einstellungen gelten nur für das aktive Dokument.

18.6.2 Diagramm speichern...

Mit diesem Befehl wird das Diagramm des im Hauptfenster angezeigten Content Model im PNG-Format unter dem gewünschten Pfad gespeichert.



18.6.3 Schema-Dokumentation generieren



Mit diesem Befehl wird eine detaillierte Dokumentation zum 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 "Schema-Dokumentation" auswählen (Aufruf des Dialogfelds mit dem Befehl "Schema-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 werden im Abschnitt ["Dokumentationsoptionen"](#) beschrieben.

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](#)" 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	<code><element name="comment" type="string"/></code>

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.

Dokumentationsoptionen

Mit dem Befehl **Schema-Design | Schema-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.

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 Kontrollkästchen 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 `.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 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. 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.
- 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.

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.

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\XMLSpy2012\Dokumentation\Schema` gespeichert.

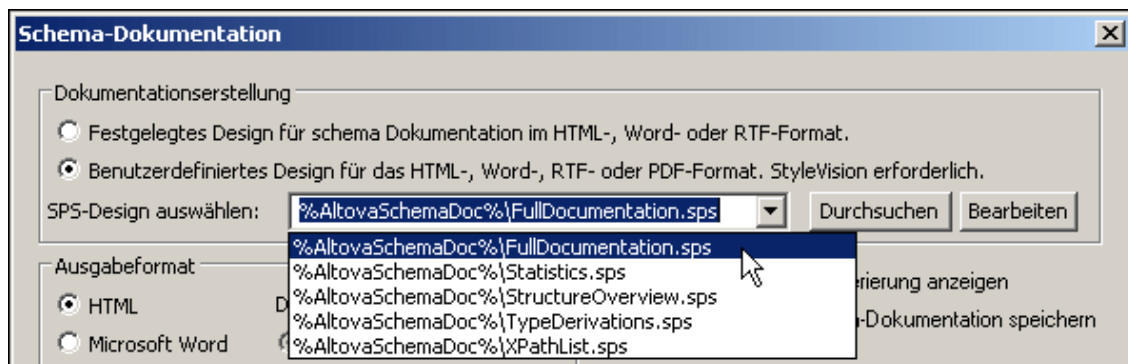
Beim Erstellen des SPS Designs in StyleVision werden Nodes aus dem Schema `SchemaDocumentation.xsd` in die Design-Vorlage eingefügt und Formatvorlagen und Eigenschaften zugewiesen. Zusätzliche Komponenten wie Links, Tabellen und Bilder können ebenfalls zum SPS hinzugefügt werden. Auf diese Weise kann 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 | Schema-Dokumentation generieren...**) ausgewählt. Wählen Sie im Bereich "Dokumentationserstellung" dieses Dialogfelds (siehe Abbildung unten) die Optionsschaltfläche *Benutzerdefiniertes 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 Verzeichnis "Eigene Dateien"](#): `C:\Dokumente und Einstellungen\<Benutzername>\Eigene Dateien\Altova\XMLSpy2011\Dokumentation\Schema\`. Es sind dies:

- `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. Zu diesem Zweck wird eine Beispieldatei namens `OrgChart.xml` mit Ihrer Applikation mitgeliefert; sie befindet sich [im Verzeichnis "Eigene Dateien"](#):

```
C:\Dokumente und Einstellungen\<Benutzername>\Eigene Dateien\Altova\XMLSpy2012
\Dokumentation\Schema\Beispieldaten
```

Anmerkung: Um ein SPS zum Generieren von Schema-Dokumentation verwenden zu können, muss StyleVision auf Ihrem Computer installiert sein.

18.6.4 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 und die Druckausgabe übernommen.

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.

Bitte beachten Sie:

- 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.

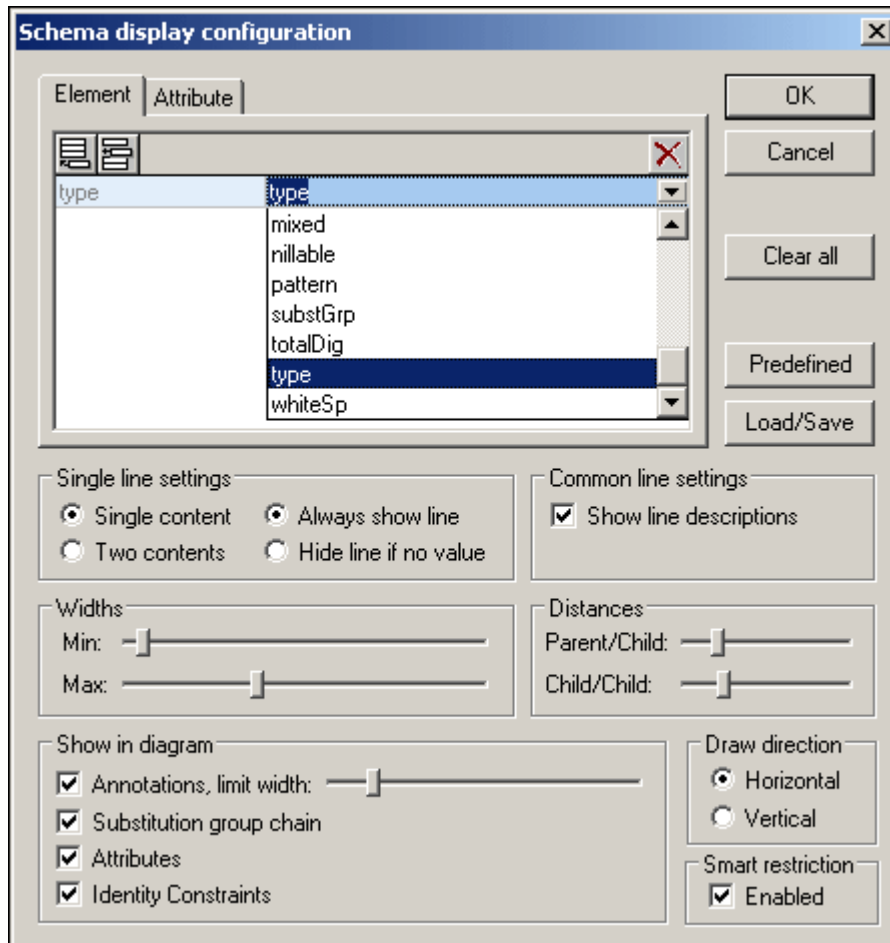
Löschen einer Eigenschaftsdeskriptorzeile aus der Content Model-Ansicht

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:



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 auszublenden. 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 Listefeld, ansonsten geben Sie den gewünschten Wert ein und bestätigen Sie mit der Eingabetaste.

Schaltflächen im Dialogfeld "Schema-Ansicht konfigurieren"

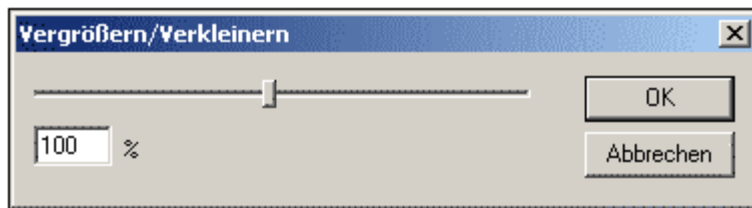
- Die Schaltfläche **Laden/Speichern** lädt und speichert die von Ihnen getätigten Einstellungen.
- Die Schaltfläche **Vordefiniert** setzt die Konfiguration der Ansicht auf die Werkseinstellungen zurück.
- Die Schaltfläche **Zurücksetzen** löscht alle Listefeld-Einträge.

Aktivieren von Smart Restrictions

Um [Smart Restrictions](#) zu aktivieren, aktivieren Sie das Kontrollkästchen für Smart Restrictions.

18.6.5 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.

18.6.6 Globale Elemente anzeigen

Mit diesem Befehl wechseln Sie von der

18.6.7 Diagramm anzeigen

Mit diesem Befehl zeigen Sie die [Content Model-Ansicht](#) 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
- Doppelklicken Sie auf den Namen der Komponente im Komponenten-Navigator (oberste rechte Eingabehilfe).

18.6.8 Schemaerweiterungen für Datenbanken

Dieser Menübefehl enthält ein Untermenü mit den Befehlen für Oracle und MS SQL Server-Schemaerweiterungen.

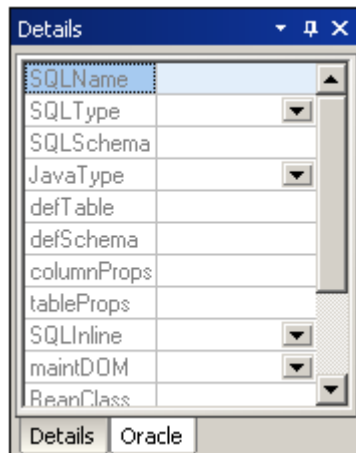
- [Oracle Schema-Extensions aktivieren](#)
- [Oracle Schema-Einstellungen](#)
- [Microsoft SQL Server Schema Extensions aktivieren](#)
- [Named Schema Relationships](#)
- [Unnamed Element Relationships](#)

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 Queries 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` and `xsd:element` hinzufügen können.

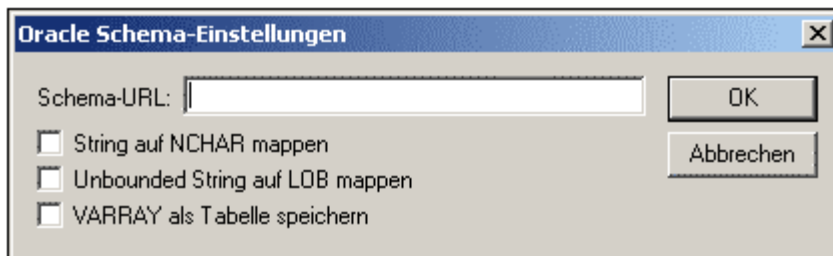


Oracle Extensions können für complexType, Elemente und Attribute definiert werden. Verwenden Sie die Eingabehilfe genau wie in XMLSpy.

Bitte beachten Sie: 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 Oracle Extensions (durch Klicken auf den aktivierten Befehl) 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!

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](#)").

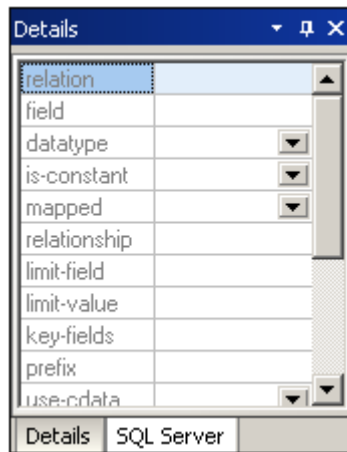
Microsoft SQL Server Schema Extensions aktivieren

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 dieser Menüoption 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 Eingabehilfe "Details" zur Verfügung, wenn die Komponente ausgewählt ist. Verwenden Sie die Eingabehilfe genau wie in XMLSpy.

Bitte beachten Sie: Sie können diesen Menübefehl aktivieren oder deaktivieren, d.h. Sie können Extensions ein- oder ausschalten. Bei aktivierten SQL Server Extensions wird links neben dem Befehl ein Häkchen angezeigt. Bei Deaktivierung von SQL Server Extensions (durch Klicken auf den aktivierten Befehl) 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!

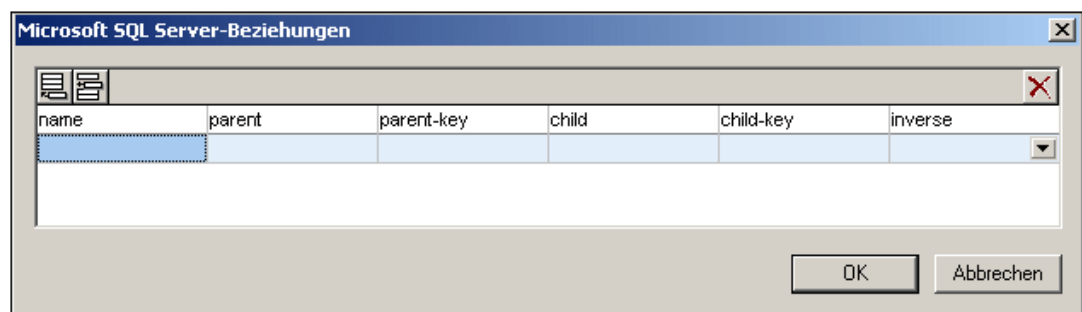
Named Schema Relationships

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.

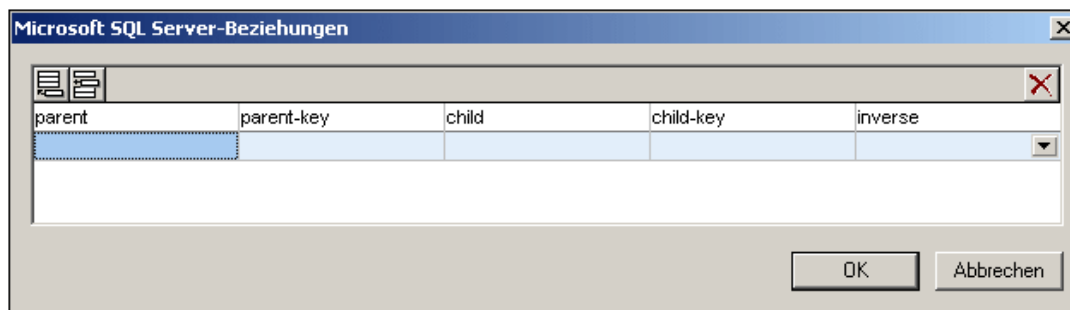
Bitte beachten Sie: Klicken Sie auf das Symbol "Löschen"  um eine Zeile aus dem Dialogfeld zu löschen.

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 Namen der Beziehung einzugeben.
3. Mit OK bestätigen Sie Ihre Eingaben.



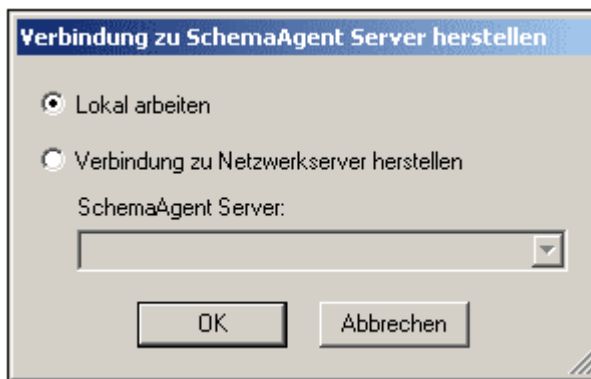
Hierdurch wird ein SQL Relationship Element für das ausgewählte Schemaelement erzeugt.

Bitte beachten Sie: Klicken Sie auf das Symbol "Löschen"  um eine Zeile aus dem Dialogfeld zu löschen.

18.6.9 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 2012 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 2012 mitgeliefert wird), oder mit einem Netzwerk-Server (dem Altova SchemaAgent Server 2012, 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](#). 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.

18.6.10 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.

18.6.11 In SchemaAgent anzeigen

Mit diesem Befehl werden das aktive Schema und optional damit verknüpfte Schemas im Altova-Produkt SchemaAgent 2012 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](#).

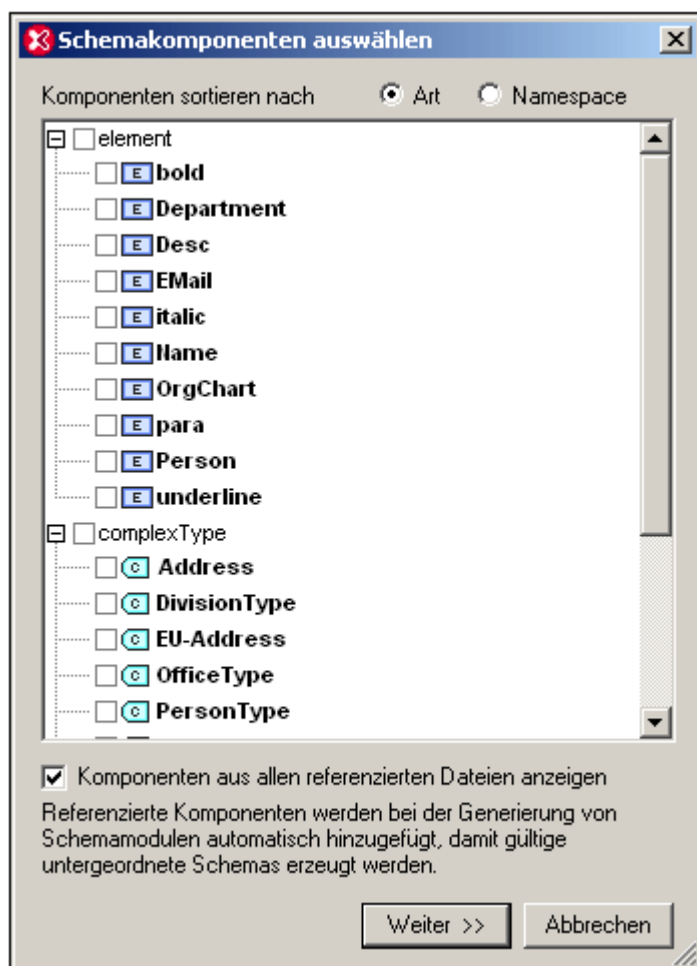
18.6.12 SchemaAgent-Validierung



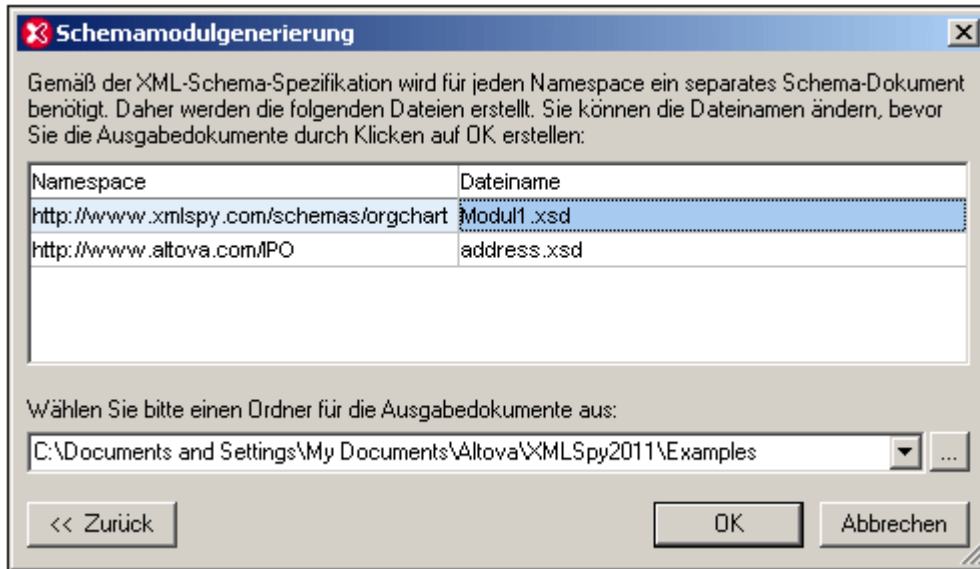
Mit diesem Befehl können Sie das aktuell aktive Schema und die damit verwandten Schemas validieren. Eine Beschreibung zu dieser Funktion finden Sie in dieser Dokumentation im Abschnitt "Schema-Ansicht" unter [SchemaAgent-Validierung](#).

18.6.13 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.

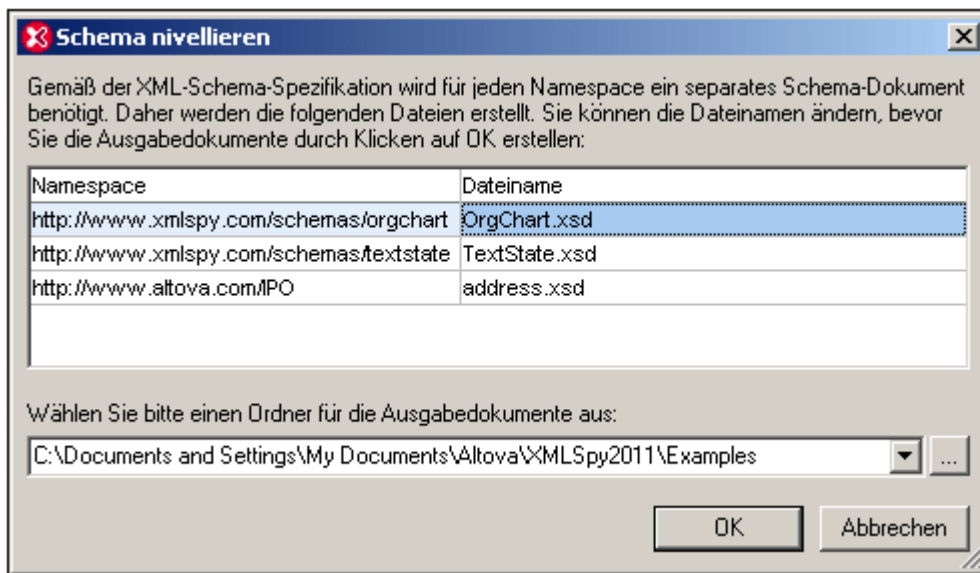


Wenn Sie auf **OK** klicken, wird die Schemamoduldatei mit dem Namespace, der dem der aktiven Datei entspricht, in der Schema-Ansicht geöffnet. Alle anderen Dateien im Paket werden erstellt, aber nicht in der Schema-Ansicht geöffnet.

18.6.14 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.

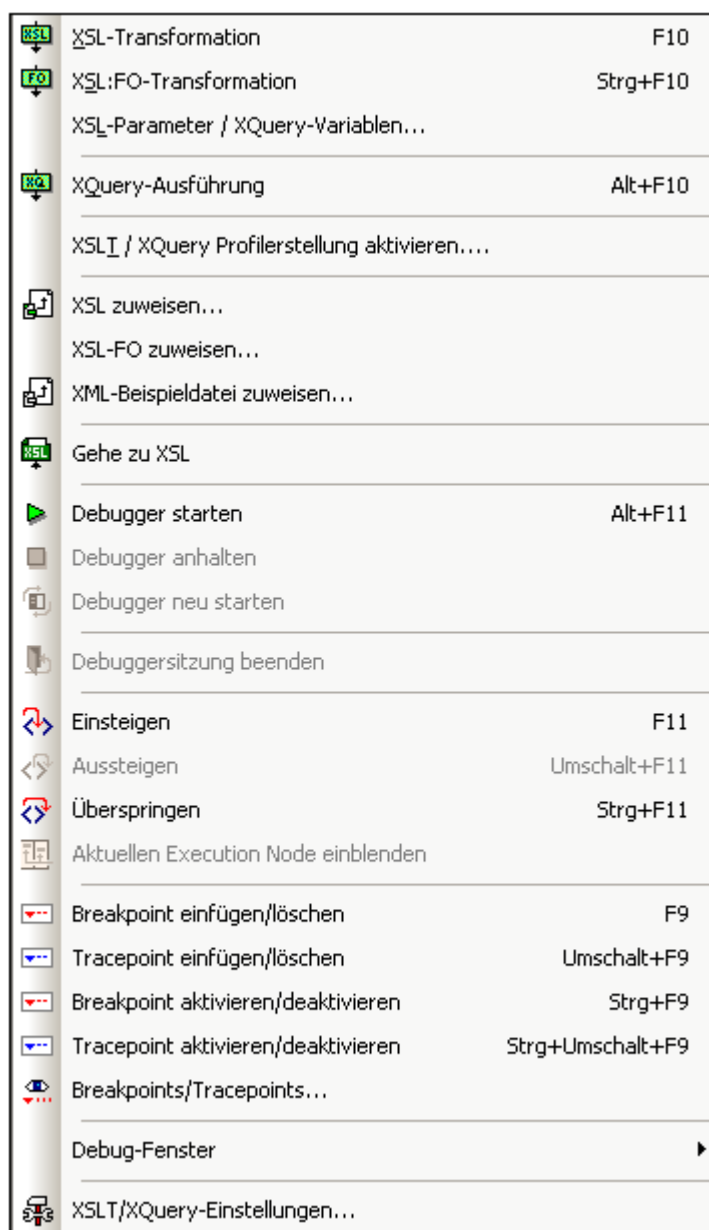
18.7 Menü "XSL/XQuery"

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 und XSLT 2.0) 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" auf dem Register XSL ([Extras | Optionen](#)) 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.



18.7.1 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), 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](#) 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](#) zugewiesen werden). Wenn das XML-Dokument Teil eines Projekts ist, können Sie im Dialogfeld "[Projekteigenschaften](#)" 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](#)) oder eines der offenen Fenster in XMLSpy auswählen (Klicken Sie dazu auf die Schaltfläche **Fenster**).

Automatisieren von XSLT-Transformationen mit Altova XML 2012

Altova XML ist eine kostenlose Applikation, die den Altova XML-Validator sowie einen XLST 1.0-, XSLT 2.0- und XQuery 1.0-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 1.0 und 2.0 Stylesheets sowie zum Ausführen von XQuery-Dokumenten verwendet werden.

XSLT-Transformationen können daher mittels Altova XML automatisiert werden. So können Sie z.B. eine Batch-Datei erstellen, die Altova XML aufruft, um damit eine Reihe von XML-Dokumenten zu transformieren. Nähere Informationen dazu finden Sie in der [Altova XML Dokumentation](#).

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).

18.7.2 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 (auch als XSL-FO bezeichnet) 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.

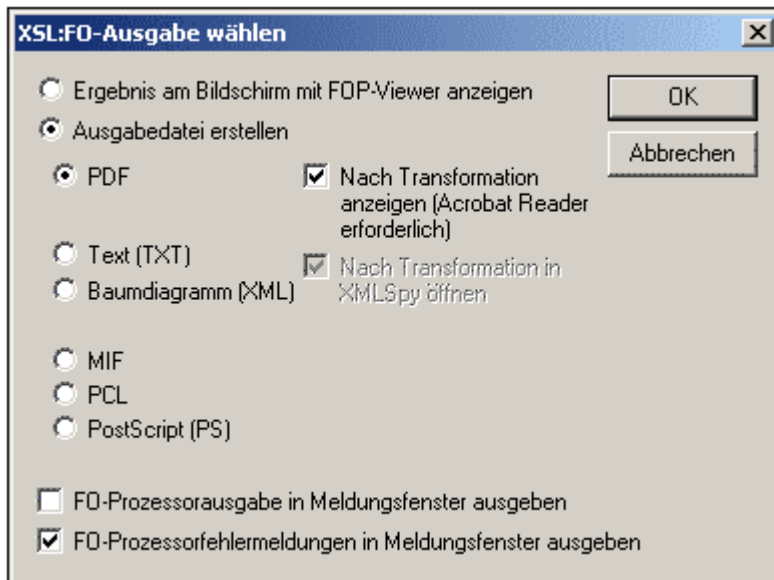
- Wird die **XSL:FO-Transformation** an einem XML-Quelldokument durchgeführt, werden beide oben beschriebenen Schritte nacheinander durchgeführt. Wenn das für die Transformation in FO benötigte XSLT (oder XSL-FO) Stylesheet im XML-Dokument nicht referenziert ist, werden Sie aufgefordert, eines für die Transformation zuzuweisen

(siehe Abbildung unten). Sie können eine Datei auch über eine globale Ressource, eine URL (klicken Sie dazu auf die Schaltfläche [Durchsuchen](#)) 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 XSL](#) 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 XSL](#) definiert wurde, direkt verarbeitet.

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](#) 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 am unteren Rand des Dialogfelds.

Bitte beachten Sie:

- Der Apache FOP-Prozessor kann kostenlos über den Link im [Altova Download Center](#) heruntergeladen werden. Laden Sie FOP herunter und installieren Sie den Prozessor und geben Sie anschließend den Pfad zur FOP Batch-Datei im Dialogfeld "Optionen" (**Extras | Optionen**) auf dem [Register XSL](#) an.
- 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.

18.7.3 XSL-Parameter / XQuery-Variablen

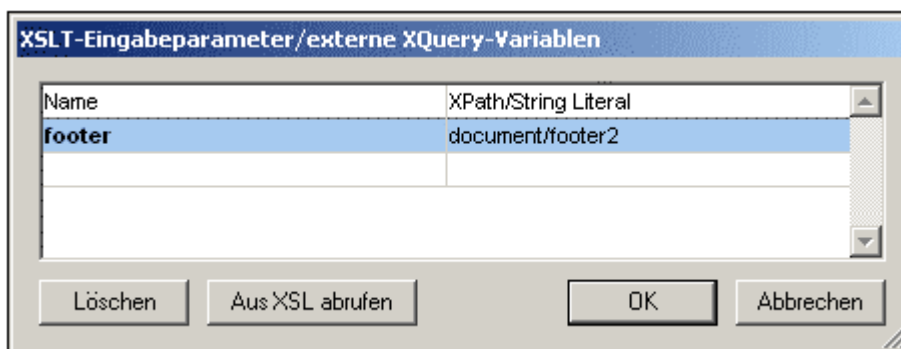
Mit dem Befehl **XSL-Parameter/XQuery-Variablen** öffnen Sie das Dialogfeld "XSLT-Eingabeparameter/externe XQuery-Variablen" (siehe Screenshot). Sie können den Namen und die dazugehörigen Werte von einem oder mehreren Parametern eingeben, die Sie an das XSLT Stylesheet übergeben möchten, oder den Namen 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.

Bitte beachten Sie: 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.

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.



Bitte beachten Sie: Sobald Sie im Dialogfeld "XSLT-Eingabeparameter/externe XQuery-Variablen" 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 "XSLT-Eingabeparameter/externe XQuery-Variablen" eingegeben wurden, werden auf Applikationsebene definiert und werden von diesem Punkt an bei jeder Transformation, die über die IDE durchgeführt wird, an das jeweilige XSLT-Dokument übergeben. Dies bedeutet:

- Parameter sind nicht mit einem bestimmten Dokument verknüpft und
- Alle Parameter, die im Dialogfeld "XSLT-Eingabeparameter/externe XQuery-Variablen" eingegeben wurden, werden gelöscht, sobald die Applikation () geschlossen wird.

Beispiel für die Verwendung von XSLT-Parametern

Im folgenden Beispiel wählen wir die für das PDF-Dokument benötigte Fußzeile aus drei Wahlmöglichkeiten im XML-Dokument (footer1, footer2, footer3) aus.

```
<?xml version="1.0" encoding="UTF-8"?>
<document xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="C:\workarea\footers\footers.xsd">
  <footer1>Footer 1</footer1>
  <footer2>Footer 2</footer2>
  <footer3>Footer 3</footer3>
  <title>Document Title</title>
  <para>Paragraph text.</para>
  <para>Paragraph text.</para>
</document>
```

Die XSLT-Datei enthält in der Vorlage für das Root-Element einen lokalen Parameter namens footer. Dieser Parameter hat einen Standardwert footer1. Der Parameter wird anschließend in der Vorlage in der Definition des Footer-Blocks mit einem Wert \$footer instanziiert.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:fo="http://www.w3.org/1999/XSL/Format">
  ...
  <xsl:param name="footer" select="document/footer1" />
  ...
  <xsl:template match="/">
    <fo:root>
      <xsl:copy-of select="$fo:layout-master-set" />
      <fo:page-sequence master-reference="default-page"
        initial-page-number="1" format="1">
        <fo:static-content flow-name="xsl-region-after"
          display-align="after">
          ...
          <fo:inline color="#800000" font-size="10pt" font-weight="bold">
            <xsl:value-of select="$footer"/>
          </fo:inline>
          ...
        </fo:static-content>
      </fo:page-sequence>
    </fo:root>
  </xsl:template>
</xsl:stylesheet>
```

Sie können im XSLT-Eingabeparameter-Dialogfeld einen neuen Wert für den Parameter footer eingeben, z.B. den XPath: document/footer2 (siehe Abbildung oben) oder einen Textstring. Während der Transformation wird dieser Wert an den Parameter footer in der Vorlage für das Root-Element übergeben und beim Instanziierten des Footer-Blocks verwendet.

Anmerkung:

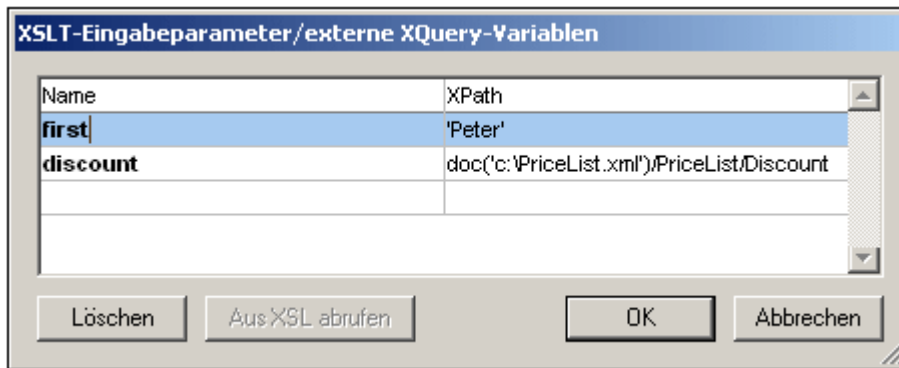
- Wenn Sie den Befehl **XSL:FO-Transformation (XSL/XQuery | XSL:FO-Transformation)** verwenden, werden Parameter, die im Dialogfeld "XSLT-Eingabeparameter/externe XQuery-Variablen" eingegeben werden, **nicht** an das Stylesheet übergeben. Damit diese Parameter in der PDF-Ausgabe verwendet

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 "XSLT-Eingabeparameter/externe XQuery-Variablen" 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 "XSLT-Eingabeparameter/externe XQuery-Variablen" eingegeben wurden, werden auf Applikationsebene definiert und werden von diesem Punkt 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 "XSLT-Eingabeparameter/externe XQuery-Variablen" eingegeben wurden, werden gelöscht, sobald die Applikation () 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.
- 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 )
```

Anmerkung: Es handelt sich nicht um einen Fehler, wenn im Dialogfeld "XSLT-Eingabeparameter/externe XQuery-Variablen" eine externe XQuery-Variable (oder ein XSLT-Parameter) definiert ist, diese im XQuery-Dokument aber nicht verwendet wird. Auch wenn ein XSLT-Parameter (oder eine externe XQuery-Variable) im Dialogfeld "XSLT-Eingabeparameter/externe XQuery-Variablen" definiert ist, aber in einer XSLT-Transformation nicht verwendet wird, handelt es sich nicht um einen Fehler.

18.7.4 XQuery-Ausführung



Mit diesem Befehl wird ein XQuery-Dokument ausgeführt. Der Befehl steht zur Verfügung, wenn eine XQuery- 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](#)) oder eines der offenen Fenster in XMLSpy auswählen (Klicken Sie dazu auf die Schaltfläche **Fenster**).

Automatisieren der XQuery-Ausführung mit Altova XML 2012

Altova XML ist eine kostenlose Applikation, die den Altova XML-Validator sowie einen XLST 1.0-, XSLT 2.0- und XQuery 1.0-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 1.0 und 2.0 Stylesheets sowie zum Ausführen von XQuery-Dokumenten verwendet werden.

XQuery-Aufgaben können daher mittels Altova XML automatisiert werden. So können Sie z.B. eine Batch-Datei erstellen, die Altova XML aufruft, um eine Reihe von XQuery-Dokumenten auszuführen. Nähere Informationen dazu finden Sie in der [Altova XML Dokumentation](#).

18.7.5 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](#)) 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:

```
<?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.

18.7.6 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:

```
<?xmlspyxslfo C:\Program Files\Altova\xmlspy\Examples\OrgChartFO.xsl?>
```

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](#)) 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.

18.7.7 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](#)) oder eines der offenen Fenster in XMLSpy auswählen (Klicken Sie dazu auf die Schaltfläche **Fenster**).

18.7.8 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.

18.7.9 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 Sitzung läuft, wird das XSLT/XQuery-Dokument bis zum Ende verarbeitet oder bis das Programm auf den nächsten Breakpoint stößt.

18.7.10 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.

18.7.11 Debugger neu starten



Mit diesem Befehl wird der XSLT Debugger mit den aktuell ausgewählten Dateien erneut gestartet und das Ausgabefenster wird gelöscht.

18.7.12 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](#) gewählt haben.

18.7.13 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.

18.7.14 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.

18.7.15 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.

18.7.16 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.

18.7.17 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. Dieser Befehl steht auch zur Verfügung, wenn Sie mit der rechten Maustaste auf die gewünschte Stelle für den Breakpoint klicken.

18.7.18 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.

18.7.19 Breakpoint aktivieren/deaktivieren



Strg+F9

Mit diesem Befehl (es gibt keine Symbolleisten-Schaltfläche) 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.

18.7.20 Tracepoint aktivieren/deaktivieren



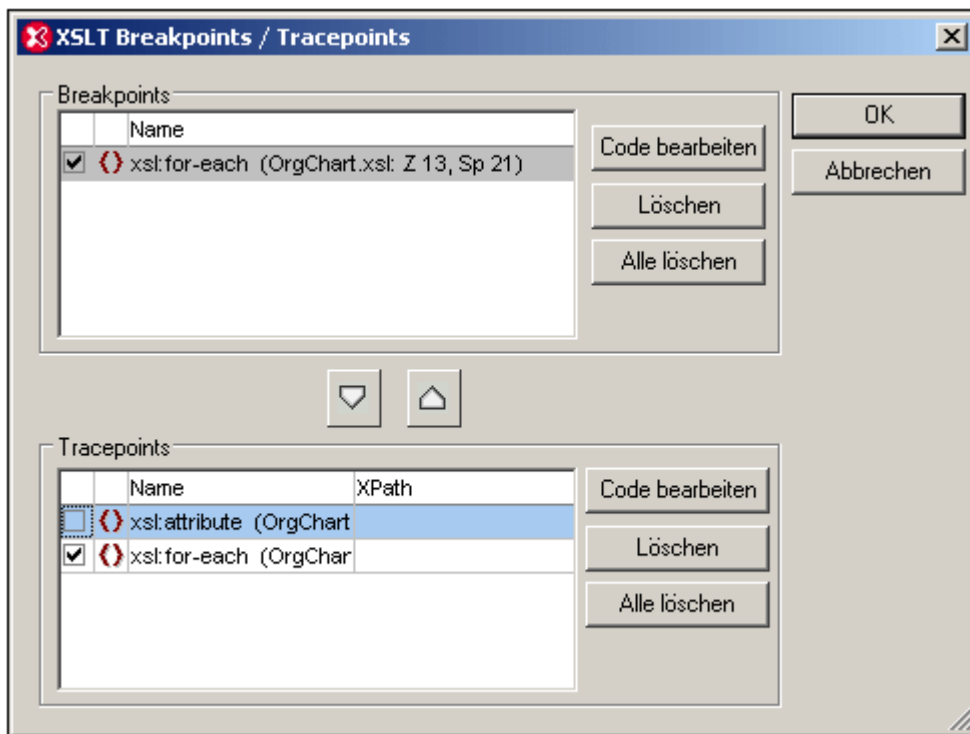
Strg+Umschalt+F9

Mit diesem Befehl (es gibt keine Symbolleisten-Schaltfläche) 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.

18.7.21 Breakpoints/Tracepoints...



Mit diesem Befehl öffnen Sie das Dialogfeld "XSLT/XQuery Breakpoints", in welchem eine Liste zuvor definierter Breakpoints und Tracepoints (einschl. aller deaktivierten Breakpoints und Tracepoints) in allen Dateien in der aktuellen Debugger-Sitzung angezeigt wird.



Das Kontrollkästchen zeigt an, ob ein Breakpoint aktiv (Häkchen) oder deaktiviert ist. Um einen markierten Breakpoint/Tracepoint zu entfernen, klicken Sie auf die Schaltfläche **Entfernen**. Zum Entfernen aller Breakpoints klicken Sie auf **Alle löschen**. Wenn Sie auf die Schaltfläche **Code bearbeiten** klicken, wird der entsprechende Breakpoint in der Datei direkt aufgerufen.

Mit Hilfe des Abwärtspfeils  können Sie den markierten Breakpoint in das Fenster

Tracepoints verschieben. Mit dem Aufwärtspfeil  können Sie den markierten Tracepoint in das Fenster **Breakpoints** verschieben.

In der Spalte **XPath** im Fenster "Tracepoints" können Sie einen XPath-Ausdruck für die einzelnen Tracepoints definieren.

18.7.22 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.

18.7.23 XSLT/XQuery-Einstellungen



Mit diesem Befehl rufen Sie das Dialogfeld "XSLT/XQuery-Einstellungen" auf, in welchem Sie

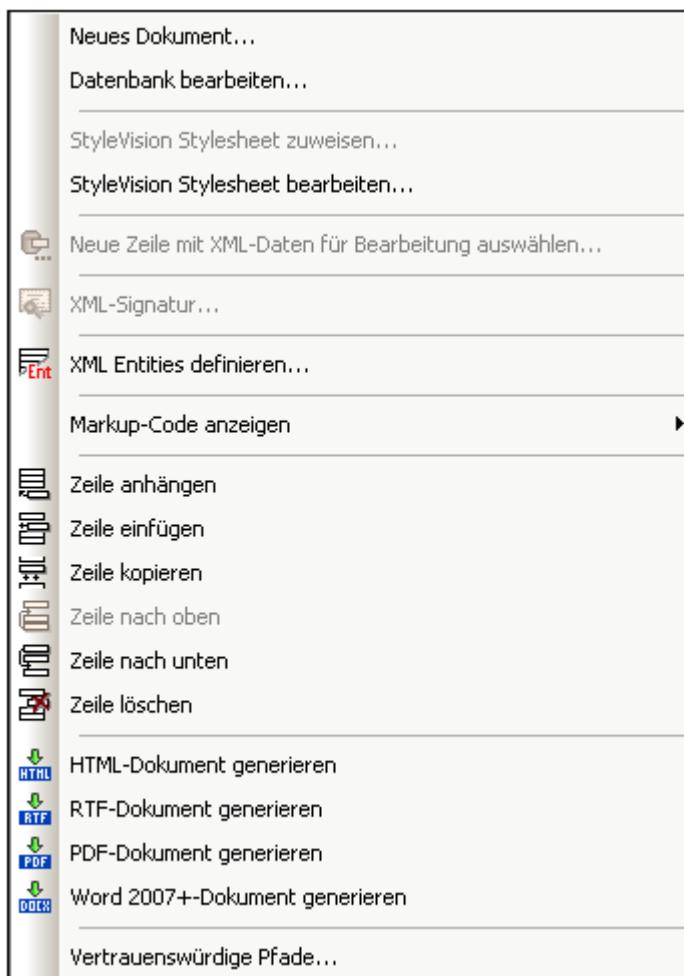
Benutzereinstellungen für den Debugger festlegen können. Nähere Informationen dazu finden Sie im Abschnitt [XSLT/XQuery Debugger](#).

18.8 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 Anzeigeeinformationen 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.

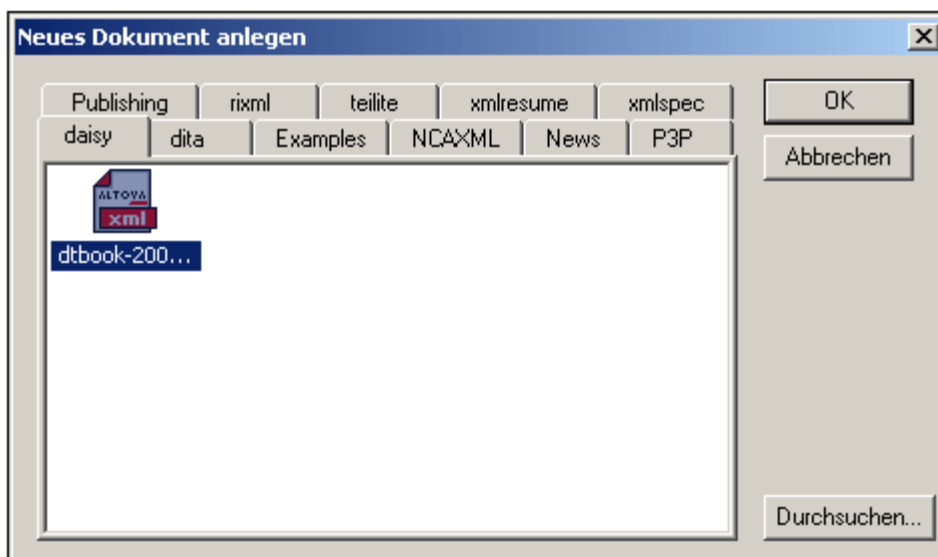
Das Menü "Authentic" enthält Befehle zum Editieren von XML-Dokumenten in der Authentic-Ansicht. Ein Tutorial zur Authentic-Ansicht finden Sie im Abschnitt [Tutorials](#).



18.8.1 Neues Dokument

Mit dem Befehl **Neues Dokument...** können Sie eine neue XML-Dokumentenvorlage in der Authentic-Ansicht öffnen. Die XML-Dokumentenvorlage basiert auf einem StyleVision Power Stylesheet (.sps-Datei) und wird durch Auswahl des StyleVision Power Stylesheet geöffnet.

Wenn Sie den Befehl **Neues Dokument...** auswählen, wird das Dialogfeld "Neues Dokument anlegen" angezeigt.



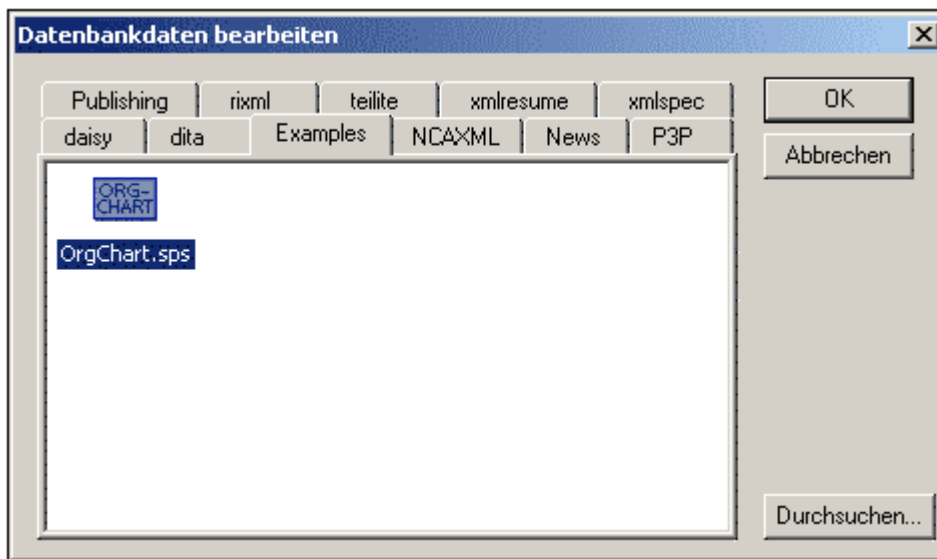
Suchen Sie die gewünschte SPS-Datei und wählen Sie diese aus. Daraufhin wird eine XML-Dokumentenvorlage in der Authentic-Ansicht geöffnet.

Bitte beachten Sie: StyleVision Power Stylesheets werden mit Hilfe von Altova <%SV-PROD%> erstellt. Dem StyleVision Power Stylesheet ist eine XML-Vorlagendatei zugewiesen. Die Daten in dieser XML-Datei liefern die Ausgangsdaten für die neue Dokumentenvorlage, die in der Authentic-Ansicht geöffnet wird.

18.8.2 Datenbankdaten bearbeiten

Mit dem Befehl **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 solches DB-basiertes StyleVision Power Stylesheet. Damit 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".



Suchen Sie die gewünschte SPS-Datei und wählen Sie diese aus. Diese Datei stellt die Verbindung zur DB her und öffnet eine editierbare Ansicht der Daten in der Authentic-Ansicht.

Warnung: 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.

Bitte beachten Sie: StyleVision Power Stylesheets werden mit Altova <%SV-PROD%> erstellt.

18.8.3 StyleVision Stylesheet zuweisen

Mit diesem Befehl 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**) oder eines der offenen Fenster in XMLSpy auswählen (Klicken Sie dazu auf die Schaltfläche **Fenster**).

```
<?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.

Bitte beachten Sie: 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.

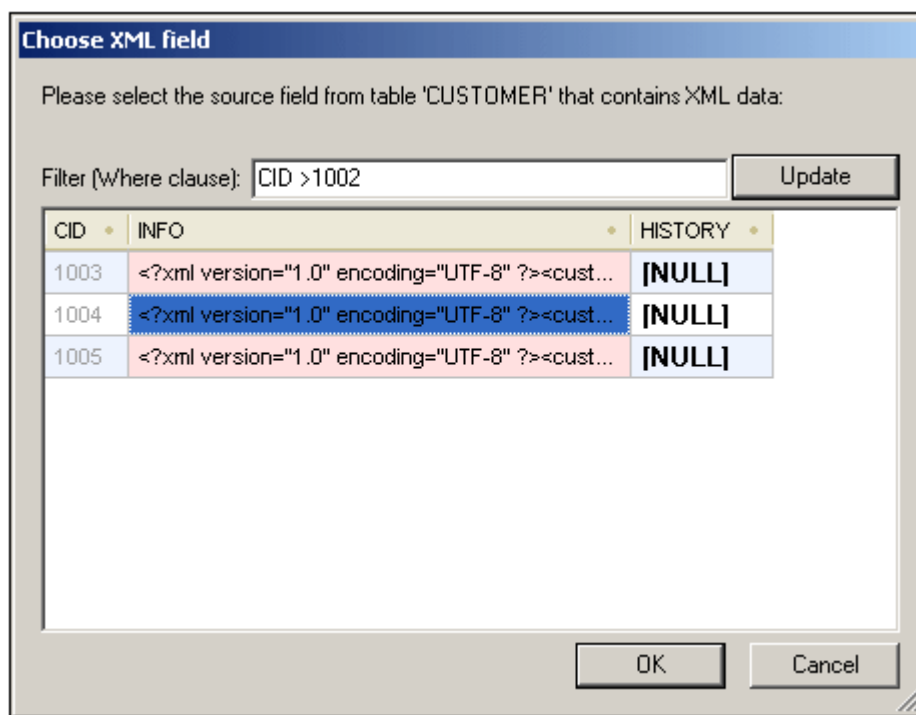
Bearbeiten eines StyleVision Stylesheet

Mit dem Befehl **Authentic | StyleVision Stylesheet bearbeiten** wird StyleVision gestartet. Auf diese Art können Sie das StyleVision Power Stylesheet sofort in StyleVision bearbeiten.

18.8.4 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.



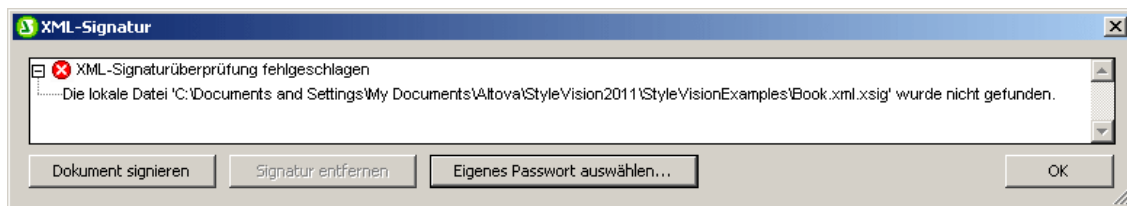
Sie können für diese Tabelle einen Filter eingeben. Dabei sollte es sich um eine SQL `WHERE`-Klausel handeln (nur die Bedingung ohne das `WHERE`-Schlüsselwort wie z.B.: `CID>1002`). Klicken Sie auf **Aktualisieren**, um das Dialogfeld zu aktualisieren. In der Abbildung oben sehen Sie das Ergebnis einer 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.

18.8.5 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

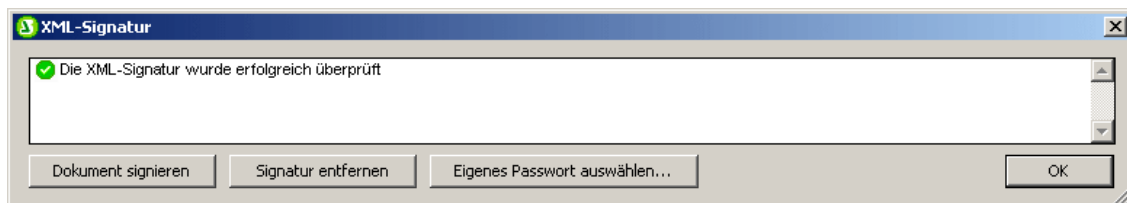
Bei Auswahl des Befehls **XML-Signatur** wird mit der Überprüfung der Signatur begonnen. Wenn keine Signatur im Dokument vorhanden ist, erscheint im Dialogfeld "XML-Signatur" eine entsprechende Meldung (*siehe Abbildung unten*) und das Dialogfeld weist eine Schaltfläche auf, mit der der Benutzer der Authentic-Ansicht das Dokument signieren kann.



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-/Passwordeinstellung wieder auf die mit dem SPS gespeicherte Originaleinstellung zurückgesetzt.

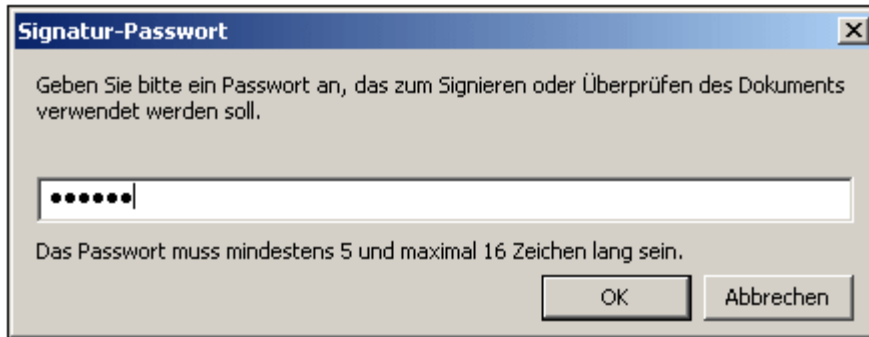
Informationen für die Überprüfung und Authentifizierung

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.



18.8.6 XML Entities definieren

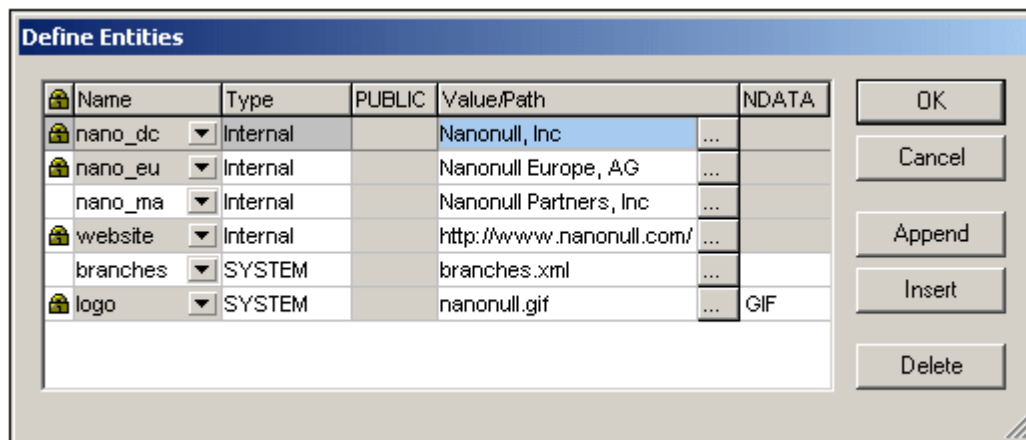
Unabhängig davon, ob Ihr Dokument auf einer DTD oder einem XML-Schema basiert, können Sie für die Authentic-Ansicht Entities definieren, die dann in der Eingabehilfe "Entities" und im Kontext-Untermenü **Entity einfügen** angezeigt werden. Wenn Sie in der Eingabehilfe "Entities" auf eine Entity doppelklicken, wird die Entity an der aktuellen Cursorposition eingefügt.

Eine Entity ist nützlich, wenn Sie einen Textstring, ein XML-Fragment oder eine andere externe Ressource an mehreren Stellen in Ihrem Dokument einfügen möchten. Sie definieren die Entity, die im Grund eine Kurzbezeichnung ist, die die gewünschten Daten repräsentiert, im Dialogfeld "Entities definieren". 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 Arten von Entities verwenden: **geparste** Entities, wobei es sich um XML-Daten handelt (entweder Textstrings oder ein Fragment eines XML-Dokuments) oder **nicht geparste** Entities, wobei es sich um nicht-XML-Daten wie z.B. eine Binärdatei handelt (gewöhnlich Grafiken, Sound oder Multimedia-Objekte). Jede Entity hat einen Namen und einen Wert. Bei 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 nicht geparsten Entities ist der Wert der Entity eine URI, die auf die nicht-XML-Datendatei verweist.

Um eine Entity zu definieren, gehen Sie vor wie folgt:

1. Wählen Sie **Authentic | XML Entities definieren....** Dadurch wird das Dialogfeld "Entities definieren" geöffnet.



2. Geben Sie in das Feld "Name" den Namen der Entity ein. Dies ist der Name, der in der Eingabehilfe "Entities" angezeigt werden wird.
3. Wählen Sie im Feld "Type" den Entitätstyp aus der Dropdown-Liste aus. Es stehen drei Typen zur Auswahl. Bei einer **internen** Entity handelt es sich um Text, der im XML-Dokument selbst gespeichert ist. Bei Auswahl von **PUBLIC** oder **SYSTEM** geben Sie an, dass sich die Ressource außerhalb der XML-Datei befindet und mit Hilfe eines Public Identifier bzw. eines System Identifier aufgerufen wird. Ein System Identifier ist eine URI, die den Speicherort einer Ressource angibt. Der Public Identifier ist ein pfadunabhängiger Identifier, der es einigen Prozessoren erlaubt, die Ressource zu identifizieren. Wenn Sie sowohl einen Public Identifier als auch einen System Identifier definieren, wird der Public Identifier mit dem System Identifier aufgelöst und der System Identifier wird verwendet.
4. Wenn Sie als Typ PUBLIC ausgewählt haben, geben Sie den Public Identifier Ihrer Ressource im Feld PUBLIC ein. Wenn Sie Internal oder SYSTEM als Typ ausgewählt haben, wird das Feld PUBLIC deaktiviert.
5. Im Feld **Value/Path** haben Sie folgende Eingabemöglichkeiten:
 - Ist der Entitätstyp "Internal", geben Sie den gewünschten Textstring als Wert der Entity ein. Verwenden Sie zur Begrenzung der Entity keine Anführungszeichen, da diese als Teil des Textstrings behandelt werden.
 - Ist der Entitätstyp SYSTEM, geben Sie die URI der Ressource an oder wählen Sie mit Hilfe der Schaltfläche zum Durchsuchen eine Ressource in Ihrem lokalen Netzwerk aus. Wenn die Ressource gepackte Daten enthält, muss es eine XML-Datei sein (d.h. sie muss eine XML-Erweiterung haben). Andernfalls kann als Ressource eine Binärdatei wie z.B. eine GIF-Datei verwendet werden.
 - Ist der Entitätstyp PUBLIC, müssen Sie in diesem Feld zusätzlich noch einen System Identifier eingeben.
6. Der Eintrag NDATA sagt dem Prozessor, dass diese Entity nicht gepackt, sondern an den entsprechenden Prozessor gesendet werden soll. Das Feld NDATA sollte daher nur bei nicht gepackten 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 ein Schlosssymbol in der ersten Spalte gekennzeichnet. Durch das Sperren einer Entity wird

sichergestellt, dass das XML-Dokument hinsichtlich der Entities gültig ist. (Das Dokument wäre ungültig, wenn eine Entity referenziert ist, die nicht definiert ist.)

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.

18.8.7 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.



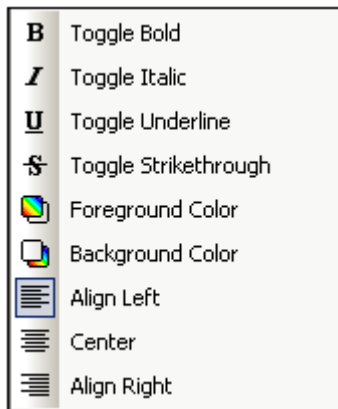
Mit dem Befehl **Große Markup-Symbole einblenden** werden große Markup Symbole in der Authentic-Ansicht angezeigt.



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.

18.8.8 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 die Schriftart, Schriftbreite, den Schriftstil, die Schriftverzierung, die Farbe, Hintergrundfarbe und Ausrichtung des ausgewählten Texts definieren.

18.8.9 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.

18.8.10 Zeile nach oben/unten



Mit dem Befehl **Zeile nach oben** wird in der Authentic-Ansicht die markierte Zeile um eine Zeile nach oben verschoben.

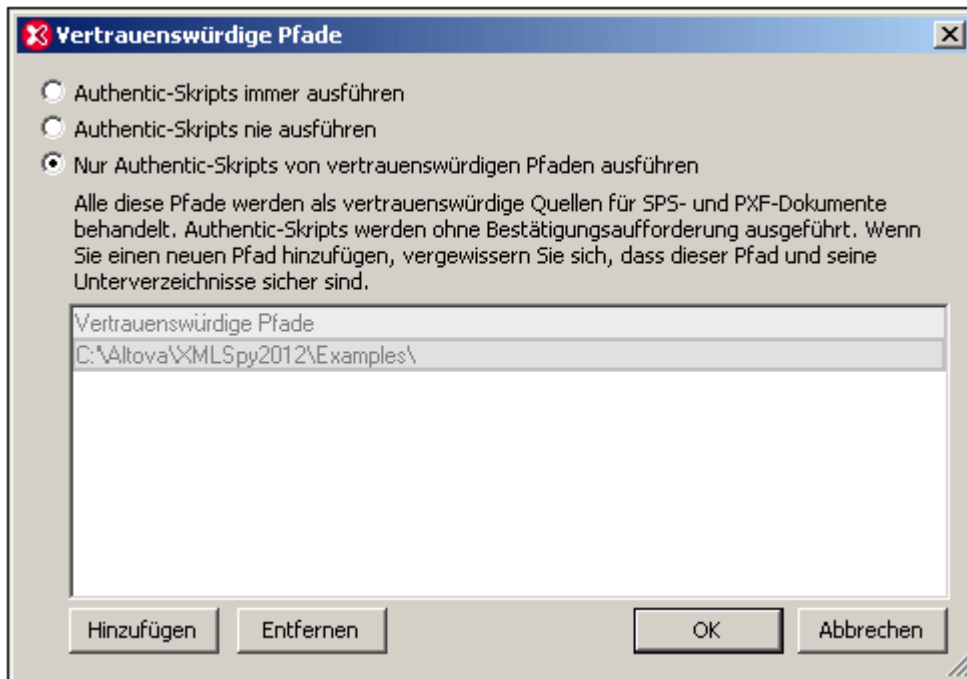


Mit dem Befehl **Zeile nach unten** wird in der Authentic-Ansicht die markierte Zeile um eine

Zeile nach unten verschoben.

18.8.11 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.



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**.

18.9 Menü "DB"

Das Menü **DB** enthält die folgenden Menüeinträge:

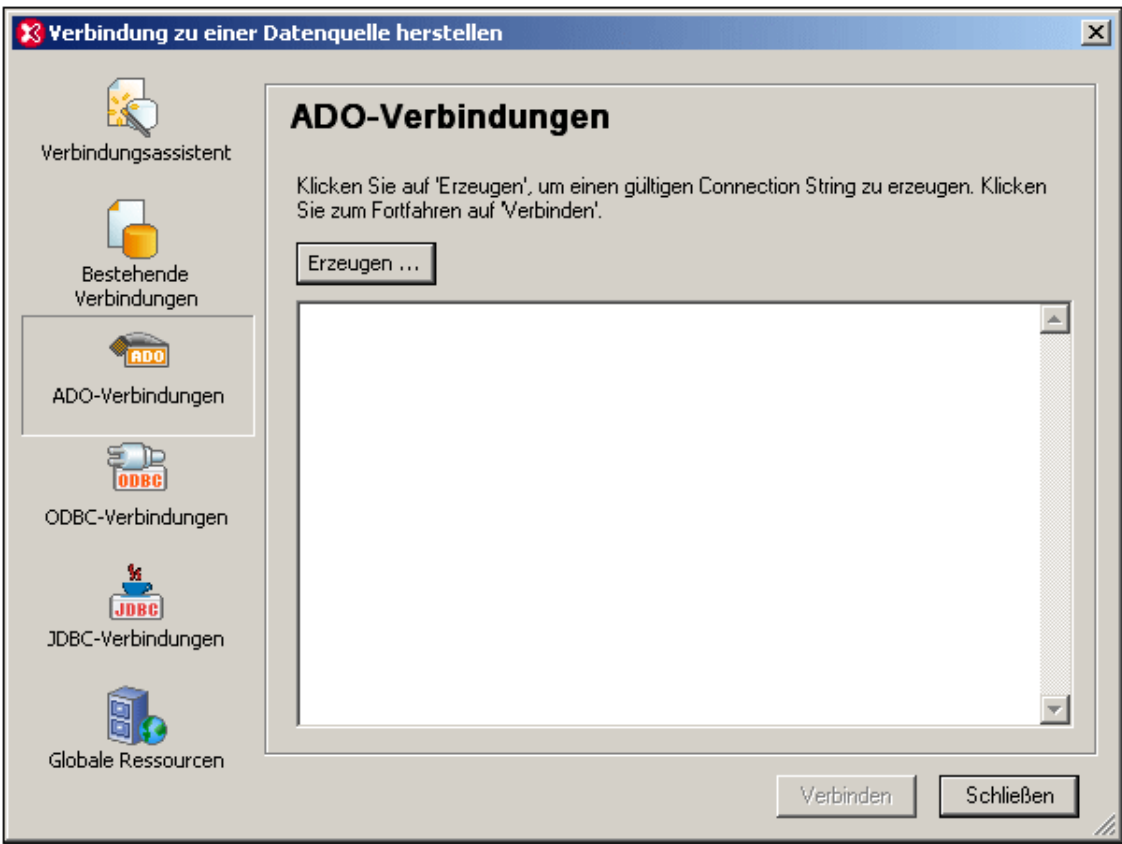
- [Datenbank abfragen](#): Dient zum Abfragen verschiedener Datenbanken.
- [IBM DB2](#): Enthält Befehle zur Unterstützung DB2-spezifischer Funktionalitäten.
- [SQL Server](#): Enthält Befehle für SQL Server-Funktionalitäten
- [Oracle-Datenbanken](#): Enthält Befehle für die Arbeit mit Oracle-Datenbanken.

Da für alle in diesem Abschnitt beschriebenen Operationen eine Datenbankverbindung erforderlich ist, enthält dieser Abschnitt auch den Unterabschnitt [Herstellen einer Verbindung zu einer Datenquelle](#).

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 jeweilige Datenbank, mit der Sie eine Verbindung herstellen, benötigt werden.

18.9.1 Herstellen einer Verbindung zu einer Datenquelle

Für eine Reihe von Befehlen in den Menüs **DB** und **Konvertieren** muss eine Verbindung zu einer Datenbank hergestellt werden. XMLSpy unterstützt die Verbindung mit verschiedenen Datenbanken (*siehe Liste unten*). Die Verbindung wird über das Dialogfeld "Verbindung zur Datenquelle herstellen" oder das Dialogfeld "Verbindungsassistent" (*Abbildung unten*) konfiguriert und hergestellt. Beide Dialogfelder sind im Wesentlichen gleich. Sie führen Sie durch die einzelnen Schritte. In diesem Abschnitt finden Sie Informationen über die Funktionsweise dieser Dialogfelder. In der Beschreibung der **Datenbank-** und **Konvertierungsbefehle**, bei denen eine Datenbankverbindung erforderlich ist, finden Sie in diesem Abschnitt nähere Informationen über das Herstellen der Verbindung.



Die Optionen zum Herstellen einer Verbindung zur Datenbank sind:

- [Über einen Verbindungsassistenten](#), der Sie durch den Vorgang führt.
- Über eine [bestehende Verbindung](#)
- [Über eine ADO-Verbindung](#)
- [Über eine ODBC-Verbindung](#)
- Über eine [JDBC-Verbindung](#)
 - Über eine [globale Ressource](#).

Die einzelnen Optionen werden in den Unterabschnitten dieses Abschnitts näher beschrieben. In diesem Abschnitt wird nur die Herstellung einer Verbindung beschrieben. Die Dialogfelder, die anschließend angezeigt werden, sind abhängig vom jeweils ausgeführten Befehl und werden daher in den Abschnitten zu den einzelnen Befehlen beschrieben. Wenn Sie z.B. eine Beschreibung zum Befehl [Konvertieren | Datenbankdaten importieren](#) suchen, finden Sie die Anleitung zum Herstellen der Verbindung in diesem Abschnitt und die Informationen zur Auswahl der DB-Tabellen im Abschnitt [Konvertieren | Datenbankdaten importieren](#).

Unterstützte Datenbanken

Es werden die folgenden Datenbanken unterstützt. Nachdem Sie eine Verbindung zur Datenbank hergestellt haben, stehen für die einzelnen unterstützten Datenbanken die folgenden Root-Objekte zur Verfügung:

Datenbank (nativ unterstützt)	Root-Objekt
MS SQL Server 2000, 2005 und 2008	database

MS SQL Server 2005 und 2008	schema
Oracle 9i, 10g und 11g	schema
MS Access 2003 und 2007	database
MySQL 4.x und 5.x	database
IBM DB2 8.x und 9	schema
Sybase 12	database
IBM DB2 for iSeries v 5.4	schema
PostgreSQL 8.0, 8.1, 8.2, 8.3	database

Hinweis: XMLSpy bietet vollständige Unterstützung für die oben aufgelisteten Datenbanken. Altova ist bestrebt, auch Unterstützung für andere ODBC/ADO-Datenbanken zu gewährleisten, doch wurde die erfolgreiche Herstellung der Verbindung und Verarbeitung der Daten nur bei den aufgelisteten Datenbanken getestet.

Verbindungsassistent

Bei Auswahl der Schaltfläche "Verbindungsassistent" im Dialogfeld "Connect to a Data Source" wird der Verbindungsassistent (*siehe Abbildung unten*) aufgerufen. Der Verbindungsassistent hilft Ihnen, eine Verbindung zu den gängigsten Datenbanktypen herzustellen. In diesem Abschnitt wird die Herstellung der Verbindung zu einer Microsoft Access Datenbank und einer IBM DB 2-Datenbank beschrieben.

MS Access

Um eine Verbindung zu einer MS Access-Datenbank herzustellen, führen Sie die folgenden Schritte durch. Wählen Sie Microsoft Access (ADO) (*Abbildung unten*) und klicken Sie auf **Weiter**.



Suchen Sie im Dialogfeld "Verbindung zu MS Access herstellen" die MS Access-Datenbank und klicken Sie auf **Weiter**. Die Verbindung zur Datenquelle wird hergestellt. Informationen über Dialogfelder, die in der Folge angezeigt werden, finden Sie unter der Beschreibung des ausgeführten Befehls.

IBM DB2

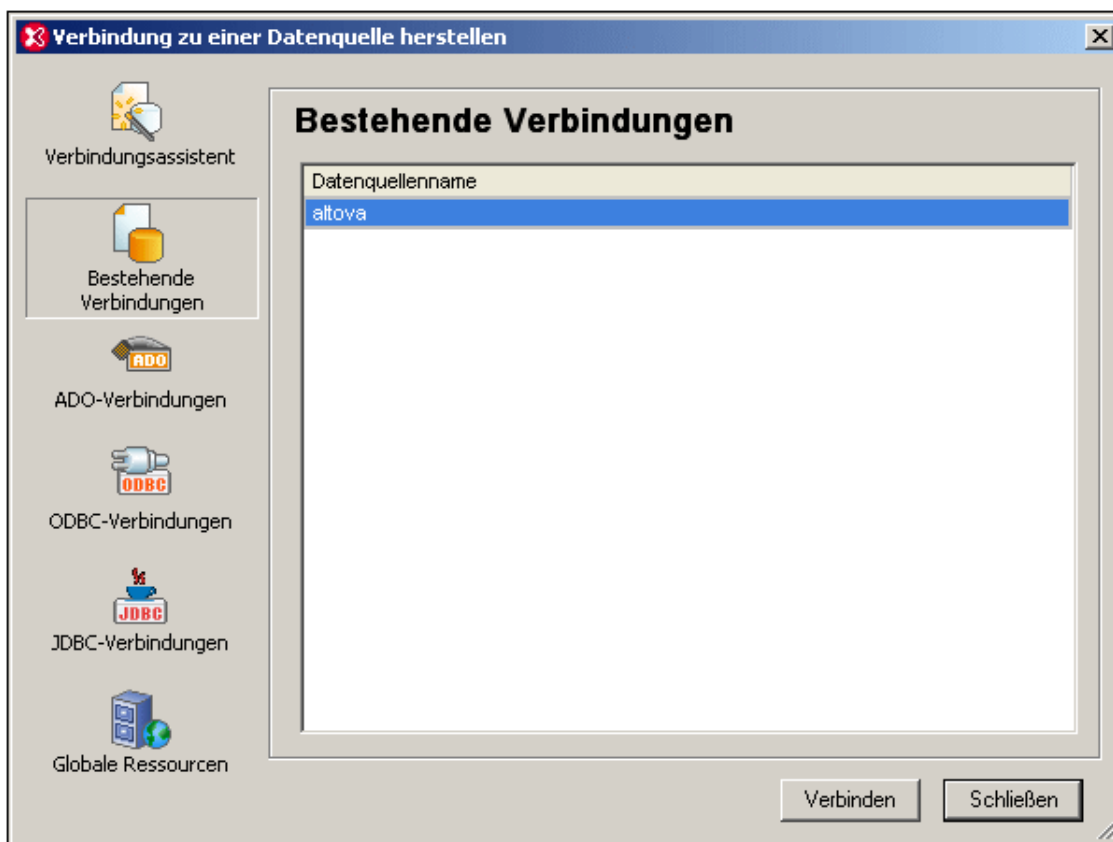
Wählen Sie im ersten Bildschirm des Verbindungsassistenten (*siehe erste Abbildung in diesem Abschnitt*) den Eintrag IBM DB2 und klicken Sie auf **Weiter**. Daraufhin wird das Konfigurationsdialogfeld (*Abbildung unten*) angezeigt. Der Assistent führt Sie anschließend durch den Konfigurationsvorgang für die IBM DB2-Datenbank und stellt die Verbindung her. Dabei wird zuerst der entsprechende Treiber für die Verbindung mit der DB ausgewählt, anschließend die DB und schließlich werden die für die Verbindung erforderlichen Benutzerinformationen eingegeben.



Wählen Sie im Konfigurationsdialogfeld oben eine bestehende Datenquelle aus oder erstellen Sie eine neue Datenquelle mit einem entsprechenden Treiber (Zusätzliche Treiber können zur Liste der verfügbaren Treiber durch Klicken auf die Schaltfläche **Treiber bearbeiten** und Auswahl des gewünschten Treibers hinzugefügt werden). Im folgenden Bildschirm werden Sie aufgefordert, die Benutzerdaten (ID und Passwort) einzugeben. Wenn Sie auf **OK** klicken, wird die Verbindung zur Datenbank hergestellt. Informationen über Dialogfelder, die in der Folge angezeigt werden, finden Sie unter der Beschreibung des ausgeführten Befehls.

Bestehende Verbindungen

Bei Auswahl der Schaltfläche "Bestehende Verbindungen" im Dialogfeld "Connect to a Data Source" wird das Fenster "Bestehende Verbindungen" (*Abbildung unten*) geöffnet, in dem alle derzeit bestehenden Verbindungen angezeigt werden.

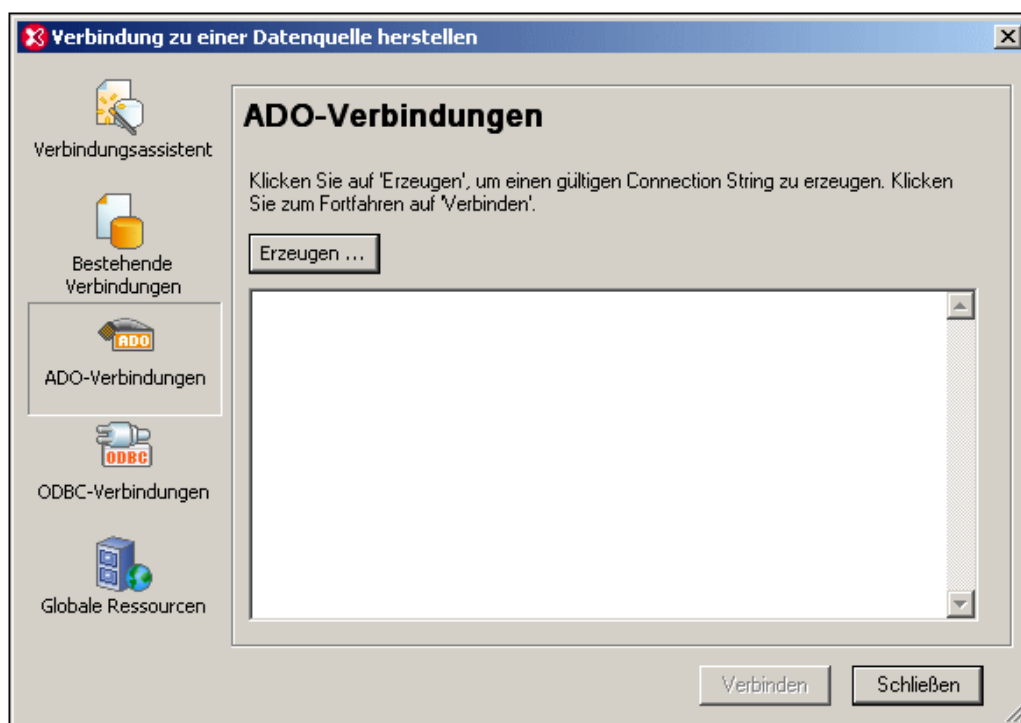


Wählen Sie die gewünschte Datenbank aus und klicken Sie auf **Verbindung herstellen**. Die Verbindung zur Datenbank wird hergestellt. Informationen über Dialogfelder, die in der Folge angezeigt werden, finden Sie unter der Beschreibung des ausgeführten Befehls.

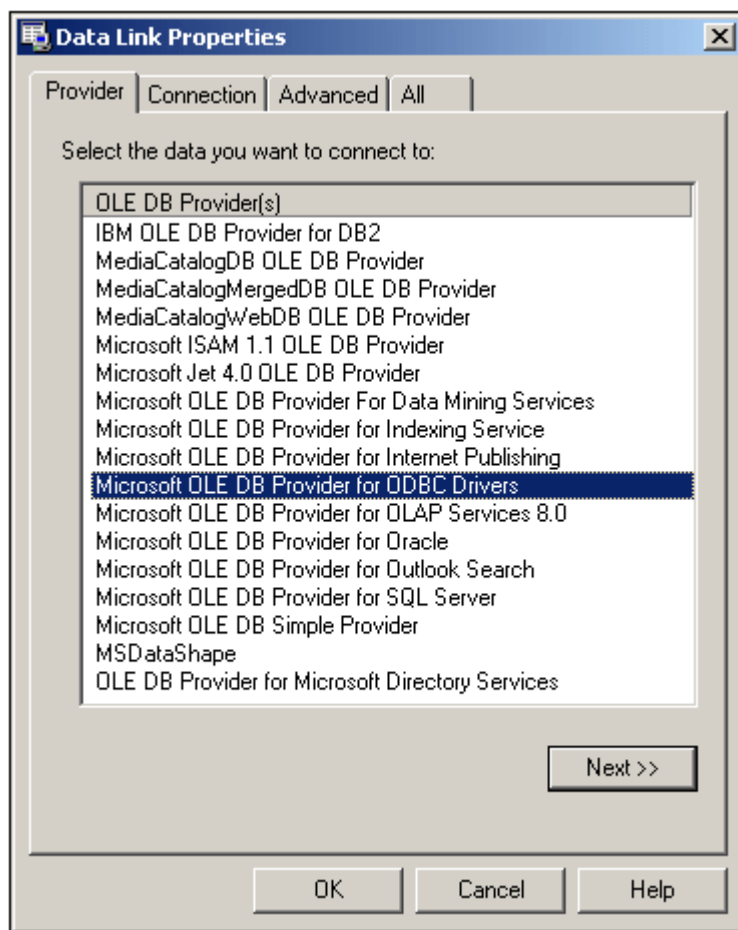
ADO-Verbindungen

Über die Option "ADO-Verbindungen" können Sie über ADO eine Verbindung zu einer Datenbank herstellen. In diesem Abschnitt wird die Herstellung einer Verbindung über ADO anhand einer IBM DB2-Datenbank als Ziel erklärt. Dort wo, sich die Optionen bei Verwendung einer anderen Datenbankart als Zieldatenbank unterscheiden, wird dies erwähnt.

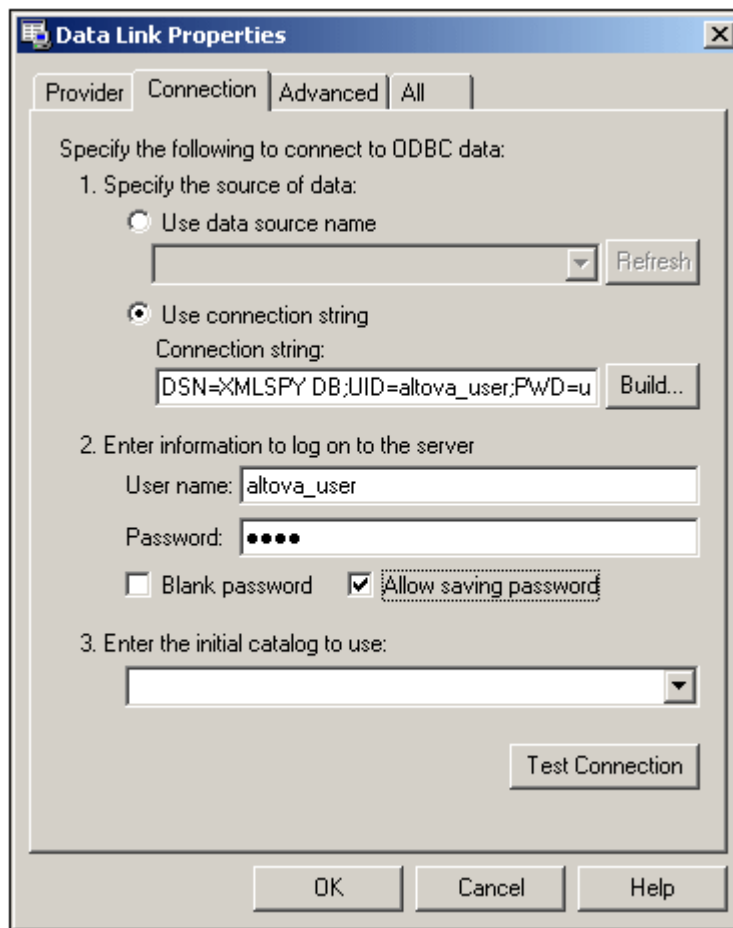
1. Wählen Sie im Dialogfeld "Verbindung zur Datenquelle herstellen" die Option "ADO-Verbindungen" aus. Daraufhin wird das Dialogfeld "ADO-Verbindungen" angezeigt (*Abbildung unten*).



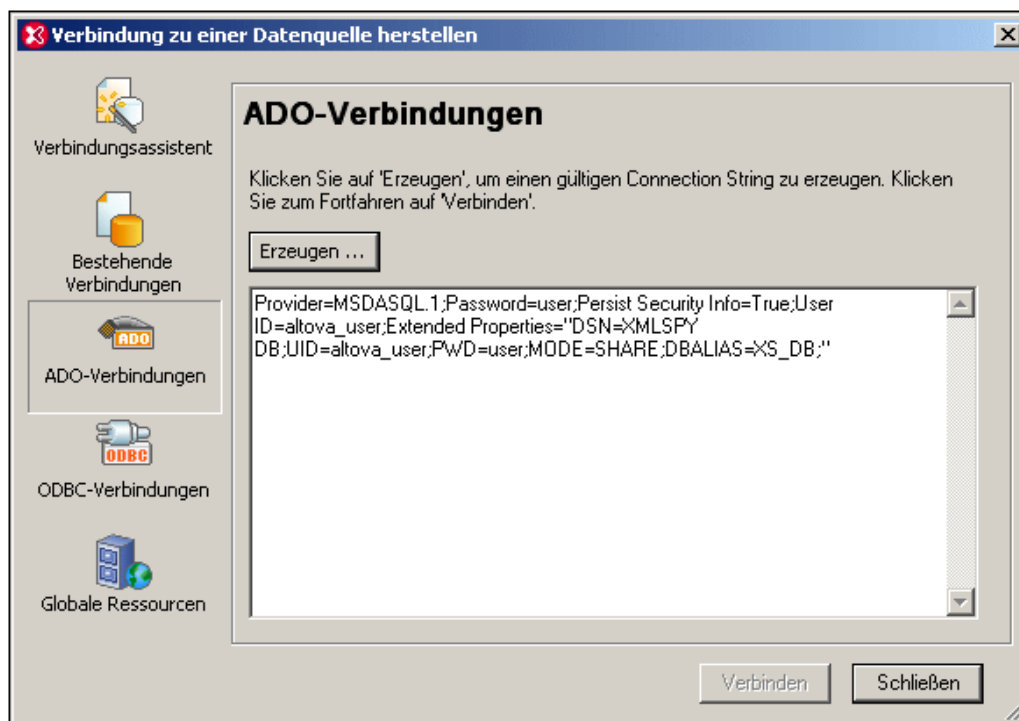
2. Klicken Sie auf die Schaltfläche **Erzeugen**. Das Dialogfeld "Datenverknüpfungseigenschaften" (*Abbildung unten*) wird geöffnet.



3. Wählen Sie "Microsoft OLE DB Provider for ODBC Drivers" in der Drowdown-Liste aus und klicken Sie auf **Weiter**. Das Register "Verbindung" ist aktiv. (Bei Microsoft SQL-Datenbank empfehlen wir die Auswahl von *Microsoft OLE DB Provider for SQL Server*; bei Oracle, MySQL, Sybase und IBM DB2-Datenbanken empfehlen wir *Microsoft OLE DB Provider for ODBC Drivers*.)



4. Geben Sie über die Schaltfläche **Erzeugen** einen Connection String ein. (In dem Dialogfeld, das erscheint, werden Sie aufgefordert die Datenquelle und Benutzerdaten einzugeben). Bei einigen Datenbanken wird kein Connection String erstellt, sondern der Server und die Datenbank werden aus Auswahllisten ausgewählt.
5. Falls Login-Informationen benötigt werden, geben Sie den Benutzernamen und das Passwort ein und aktivieren Sie das Kontrollkästchen **"Speichern des Kennworts zulassen"**.
6. Klicken Sie auf **OK**. Das Dialogfeld "Datenverknüpfungseigenschaften" wird geschlossen. Der Connection String wird im Feld "ADO-Verbindungen" angezeigt (Abbildung unten).

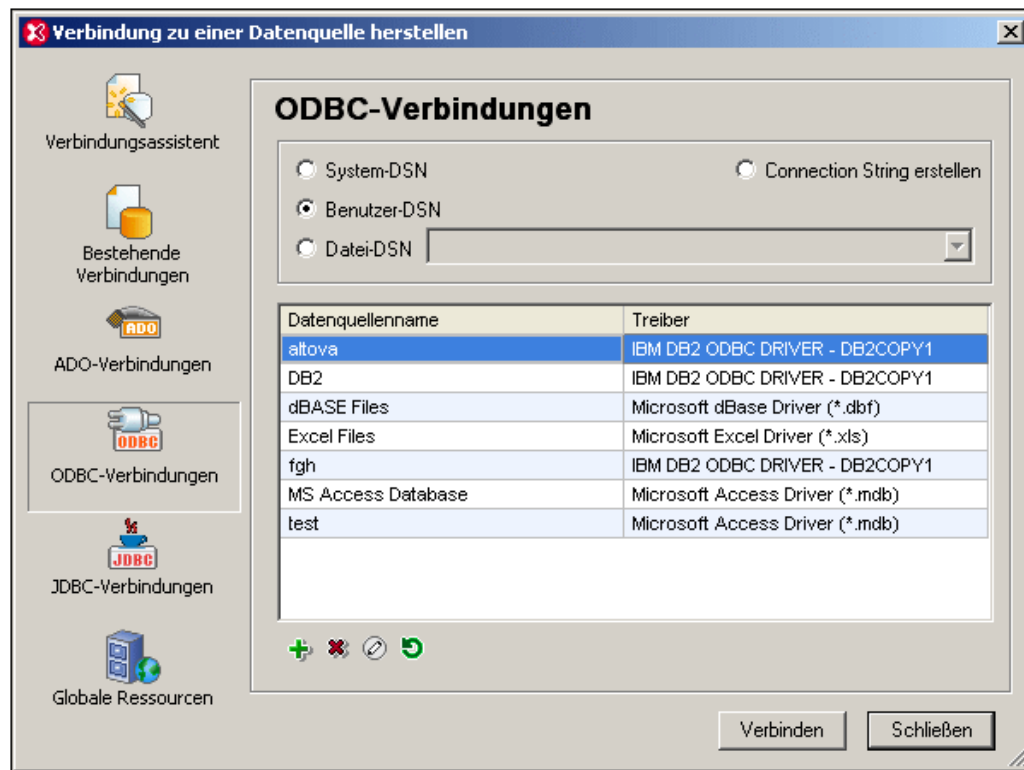


7. Klicken Sie auf die Schaltfläche **Verbindung herstellen**. Daraufhin wird eine Verbindung zur Datenquelle hergestellt. Informationen über Dialogfelder, die in der Folge angezeigt werden, finden Sie unter der Beschreibung des ausgeführten Befehls.

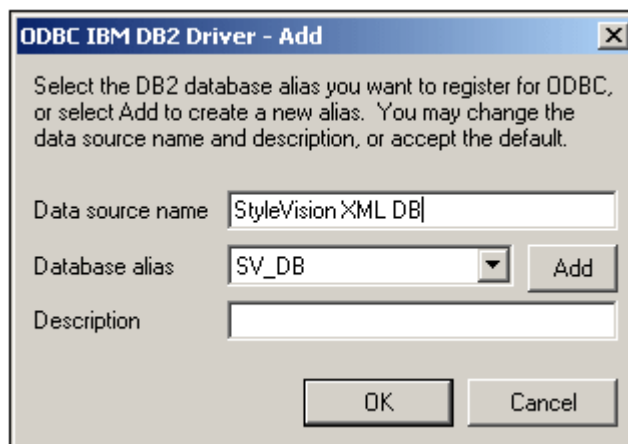
ODBC-Verbindungen

In diesem Abschnitt wird beschrieben, wie Sie über ODBC eine Datenbankverbindung herstellen. In der Anleitung unten wird eine Verbindung zu einer IBM DB2-Datenbank hergestellt. Dieselben Schritte werden aber auch bei anderen Datenbankarten, zu denen über ODBC eine Verbindung hergestellt werden kann, ausgeführt. Um eine Verbindung zu einer IBM DB2-Datenbank über ODBC herzustellen, gehen Sie folgendermaßen vor:

1. Wählen Sie im Dialogfeld "Connect to a Data Source" die Option **ODBC-Verbindungen** aus.

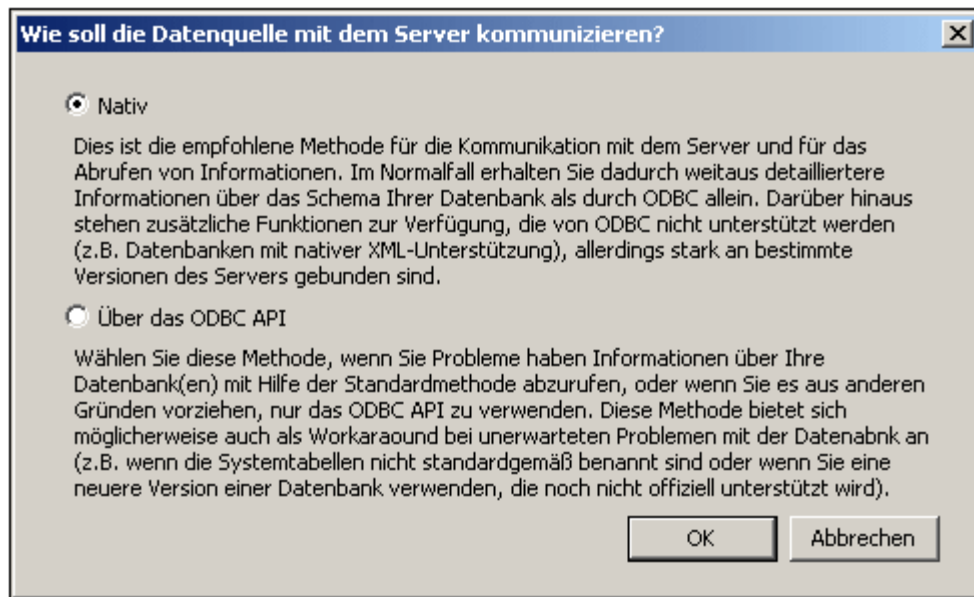


2. Wählen Sie eine der Optionen zur Definition eines Datenquellennamens (DSN). Wenn Sie "System-DSN" oder "Benutzer-DSN" wählen, werden die verfügbaren DSNs im Fenster "Datenquelle" angezeigt. Bei Auswahl von "Datei-DSN" werden Sie aufgefordert, die DSN-Datei auszuwählen. Alternativ dazu können Sie einen Connection String zur DB erzeugen, indem Sie die Option "Connections String erstellen" auswählen.
3. Wenn Sie einen DSN zu den DSNs in der Liste der Datenquellen hinzufügen möchten, klicken Sie auf die Schaltfläche "Neuen DSN erstellen" .
4. Wählen Sie im Dialogfeld "ODBC DSN erstellen" den gewünschten Treiber aus und klicken Sie anschließend auf die Schaltfläche **Benutzer DSN** oder **System-DSN**.
5. Wählen Sie im Dialogfeld, das daraufhin angezeigt wird (*Abbildung unten*), den DB-Alias aus und geben Sie ihm einen DSN.



6. Klicken Sie zum Fertigstellen auf **OK**. Die DB wird als Datenquelle zur Liste im Bereich

- "Datenquelle" hinzugefügt.
7. Nachdem Sie eine Datenquelle (über die Option "System-DSN" oder "Benutzer-DSN") oder eine DSN-Datei (über die Option "Datei-DSN") ausgewählt, oder einen Connection String (über die Option "Connection String") erstellt haben, klicken Sie auf "Verbindung herstellen".
 8. Wenn Sie nach Ihrer Benutzer-ID und dem Passwort gefragt werden, geben Sie diese ein und klicken Sie auf **OK**. Die Verbindung zur Datenquelle wird hergestellt. Informationen über Dialogfelder, die in der Folge angezeigt werden, finden Sie unter der Beschreibung des ausgeführten Befehls.
 9. Sie werden aufgefordert zwischen einer nativen Verbindung oder einer Verbindung zur Datenbank über das ODBC API (*siehe Abbildung unten*) zu wählen.



Wählen Sie die Option *Nativ*, außer wenn es Probleme mit der Verbindung gibt oder Sie lieber das ODBC API verwenden. Klicken Sie dann auf **OK**. Die Verbindung wird hergestellt. Informationen zu den weiteren Dialogfeldern finden Sie in der Beschreibung des jeweiligen ausgeführten Befehls.

Anmerkung: Die Liste im Fenster "Datenquelle" kann mittels der Schaltflächen *Den ausgewählten DSN bearbeiten*, *Den ausgewählten DSN entfernen* und *Aufgelistete DSNs aktualisieren* am unteren Rand des Fensters "ODBC-Verbindungen" bearbeitet werden (wenn die Option "System-DSN" oder "Benutzer-DSN" ausgewählt ist).



Ausgewählten DSN bearbeiten.



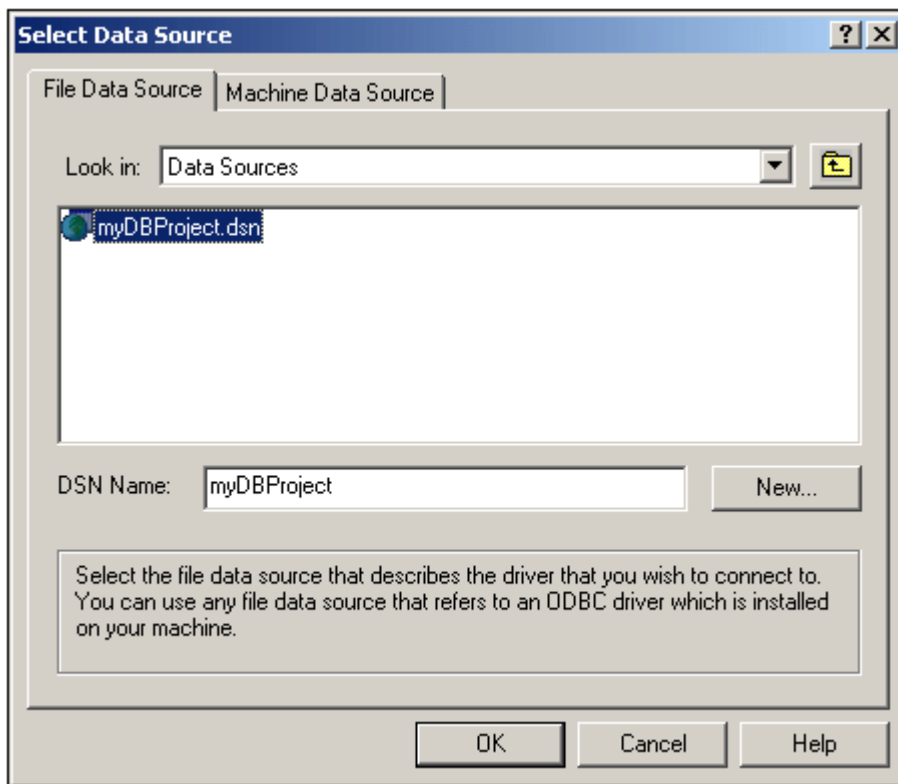
Ausgewählten DSN entfernen.



Aufgelistete DSNs aktualisieren.

Erstellung eines Connection String

Aktivieren Sie im Bereich "ODBC-Verbindungen" das Optionsfeld "Connection String erstellen" und klicken Sie auf die Schaltfläche "Erzeugen". Daraufhin erscheint das Dialogfeld "Select Data Source" (*Abbildung unten*). Sie können entweder einen Datei-DSN (auf dem Register "File Data Source") oder eine Datenquelle auswählen, die auf Ihrem Rechner zur Verfügung steht (wird auf dem Register "Machine Data Source" aufgelistet).



Wenn Sie auf **OK** klicken, erscheint ein Dialogfeld, in dem Sie nach Benutzerdaten wie u.a. Benutzer-ID und Passwort gefragt werden. Wenn Sie in diesem Dialogfeld auf **OK** klicken, wird der Connection String in den Bereich "ODBC-Verbindungen" eingegeben.

JDBC-Verbindungen

In diesem Abschnitt wird beschrieben, wie Sie über JDBC eine Verbindung zu einer Datenbank herstellen. In der Anleitung unten, wird beschrieben, wie Sie eine Verbindung zu einer IBM DB2-Datenbank herstellen. Die Anleitung gilt aber auch für andere Arten von Datenbanken, die über JDBC verbunden werden können.

Schritte vor dem Herstellen der Verbindung

Bevor Sie über JDBC eine Verbindung zu einer Datenbank herstellen, gehen Sie folgendermaßen vor:

1. Vergewissern Sie sich, dass das Java Runtime Environment (JRE) installiert ist. Ist dies nicht der Fall, dann installieren Sie es. Verwenden Sie für einen 32-Bit-Rechner ein 32-Bit JRE und für einen 64-Bit-Rechner ein 64-Bit JRE.
2. Installieren Sie einen JDBC-Treiber. Dazu ist keine spezielle Installation vonnöten. Kopieren Sie einfach den Treiber in ein lokales Verzeichnis, z.B. `c:\jdbc`. Beachten Sie, dass JDBC-Treiber (d.h. Jar-Dateien) plattformunabhängig sind.
3. Richten Sie den `CLASSPATH` ein, sodass er den Pfad enthält, unter dem der JDBC-Treiber gespeichert ist. (Die Applikation liest die `CLASSPATH`-Umgebungsvariable um den JDBC-Treiber zu finden.) Um den `CLASSPATH` aufzurufen und zu bearbeiten, gehen Sie folgendermaßen vor: Klicken Sie auf **Start | Systemsteuerung | System | Erweitert | Umgebungsvariablen**. Wählen Sie im Dialogfeld "Umgebungsvariablen", das daraufhin angezeigt wird, entweder die Benutzerumgebungsvariable `CLASSPATH`

oder die Systemumgebungsvariable `CLASSPATH` aus und klicken Sie auf die Schaltfläche **Bearbeiten**. Fügen Sie den Pfad zum JDBC-Treiber zum `CLASSPATH` hinzu. Beispiel: `CLASSPATH=C:\jdbc\sqljdbc.jar; C:\jdbc\db2jcc.jar; .`

4. Melden Sie sich beim System ab und dann wieder an, damit die `CLASSPATH` Änderungen aktiv werden.

Anmerkung: Wenn Sie den kompletten IBM DB2- oder Oracle Client installieren, wird der `CLASSPATH` automatisch mit den JDBC-Treibern der entsprechenden Pakete befüllt. Nähere Informationen finden Sie in den Beschreibungen weiter unten.

Herstellen einer Verbindung über JDBC

Um über JDBC eine Verbindung herzustellen, gehen Sie folgendermaßen vor:

1. Wählen Sie im Dialogfeld "Verbindung zu einer Datenquelle herstellen" die Option **JDBC-Verbindungen** aus. Daraufhin wird das Fenster "JDBC-Verbindungen" angezeigt (Abbildung unten).



2. Wählen Sie aus der Dropdown-Liste *Treiber* einen JDBC-Treiber aus (alle gefundenen Treiber, d.h. alle, die sich in der `CLASSPATH` Umgebungsvariablen befinden, werden aufgelistet). Geben Sie den Connection String in das Textfeld *Datenbank-URL* ein und geben Sie gegebenenfalls einen Benutzernamen und ein Passwort ein. Im Folgenden finden Sie die Connection String-Syntax für gebräuchliche Datenbanken. Zu jeder Datenbank ist ein Beispiel-Connection String aufgelistet.

Oracle `jdbc:oracle:thin:[ user/password]@//[ host][: port]/SID`
 `jdbc:oracle:thin:@//abcd234/ORA11`

IBM DB2	jdbc:db2://host_name:port/dbname jdbc:db2://MyDB2:50000/boz
MySQL	jdbc:mysql://host_name:port/dbname jdbc:mysql://MyDB2:3306/moz
MSSQL	jdbc:sqlserver://host:port;databasename=name;user=name;password=Pwd jdbc:sqlserver://abcd38:1433;databasename=coz jdbc:sqlserver://Q5;DatabaseName=coz;SelectMethod=Cursor
PostgreSQL	jdbc:postgresql://host:port/database jdbc:postgresql://abc993:5432/qanoz
Sybase	jdbc:sybase:Tds:host:port/dbname jdbc:sybase:Tds:abc12:2048/QUE

3. Klicken Sie auf die Schaltfläche **Verbinden**. Daraufhin wird die Verbindung zur Datenquelle hergestellt. Informationen über die nachfolgenden Dialogfelder finden Sie in der Beschreibung zum ausgeführten Befehl.

Schritt für Schritt: MSSQL, MySQL, PostGre und andere Nicht-XML-Datenbanken

Unten finden Sie eine schrittweise Anleitung für die Herstellung einer Verbindung zu einer Nicht-XML-Datenbank über JDBC.

1. JDBC JAR-Dateien (Treiberdateien): Kopieren Sie die Dateien in einen lokalen Ordner auf Ihrem Rechner. Fügen Sie anschließend zur Windows `CLASSPATH` Variablen den vollständigen Pfad und Dateinamen hinzu, z.B.: `C:\jdbc\sqljdbc.jar`; `C:\jdbc\db2jcc.jar`;
2. Melden Sie sich ab und wieder an, um die Änderungen am `CLASSPATH` aktiv zu machen.
3. Starten Sie XMLSpy und rufen Sie das Dialogfeld "Verbindung zu einer Datenbank herstellen" auf.
4. Im Fenster "JDBC-Verbindungen" werden die gefundenen JDBC-Treiber in der Dropdown-Liste *Treiber* aufgelistet. Wenn die Dropdown-Liste leer ist, vergewissern Sie sich, dass die Datei `altovadb.jar` im Ordner `C:\Program Files\Altova\Common 2012\jar` vorhanden ist.
5. Stellen Sie wie im Abschnitt oben unter *Herstellen einer Verbindung über JDBC* beschrieben, eine Verbindung zu einer Datenbank her.

Schritt für Schritt: Oracle

Unten finden Sie eine schrittweise Anleitung für die Herstellung einer Verbindung zu einer

Oracle-Datenbank über JDBC. Wenn Sie die XML- und Datenbankfunktionen der Oracle-Datenbank nicht benötigen, befolgen Sie die Anleitung für Nicht-XML-Datenbanken. Der Installationsordner des Oracle Client wird in der Anleitung unten durch den Platzhalter `%ORACLE_HOME%` angegeben.

1. Installieren Sie Oracle Client Software, wobei die OCI- und ODBC-Funktionen aktiviert sind. Wenn bereits ein Oracle Client installiert ist, überprüfen Sie, ob die zwei unten aufgelisteten jar-Dateien vorhanden sind:

`%ORACLE_HOME%\LIB\xmlparserv2.jar`
`%ORACLE_HOME%\RDBMS\jlib\xdb.jar`
2. Fügen Sie die folgenden Dateien zur Windows `CLASSPATH` Umgebungsvariablen hinzu:

`%ORACLE_HOME%\jdbc\lib\ojdbc6.jar`
`%ORACLE_HOME%\LIB\xmlparserv2.jar`
`%ORACLE_HOME%\RDBMS\jlib\xdb.jar`
3. Melden Sie sich ab und wieder an, um die `CLASSPATH` Änderungen aktiv zu machen.
4. Starten Sie XMLSpy und rufen Sie das Dialogfeld "Verbindung zu einer Datenbank herstellen" auf.
5. Im Fenster "JDBC-Verbindungen" werden die gefundenen JDBC-Treiber in der Dropdown-Liste *Treiber* aufgelistet. Wenn die Dropdown-Liste leer ist, vergewissern Sie sich, dass die Datei `altovadb.jar` im Ordner `C:\Program Files\Altova\Common 2012\jar` vorhanden ist.
6. Stellen Sie wie im Abschnitt oben unter *Herstellen einer Verbindung über JDBC* beschrieben, eine Verbindung zu einer Datenbank her.

Schritt für Schritt: IBM DB2

Unten finden Sie eine schrittweise Anleitung für die Herstellung einer Verbindung zu einer IBM DB2-Datenbank über JDBC.

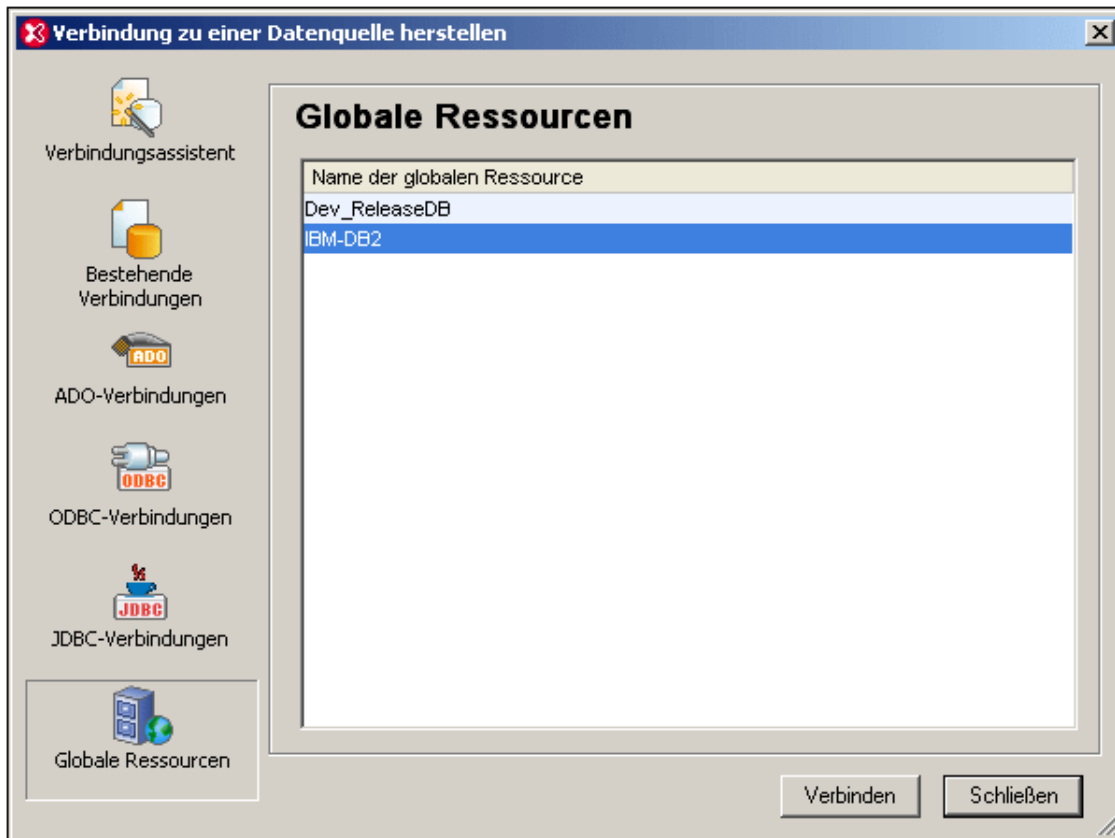
1. Wenn bereits ein IBM DB2 Client installiert wurde, müssen Sie nichts unternehmen, da der `CLASSPATH` bereits während der Installation eingerichtet wurde.
2. Wenn kein IBM DB2 Client installiert wurde, fügen Sie die IBM DB2 JDBC jar-Treiberdateien `db2jcc.jar` und `db2jcc_license_cu.jar` zum Windows `CLASSPATH` hinzu. Melden Sie sich ab und wieder an, um die `CLASSPATH` Änderungen aktiv zu machen.
3. Starten Sie XMLSpy und rufen Sie das Dialogfeld "Verbindung zu einer Datenbank herstellen" auf.
4. Im Fenster "JDBC-Verbindungen" werden die gefundenen JDBC-Treiber in der Dropdown-Liste *Treiber* aufgelistet. Wenn die Dropdown-Liste leer ist, vergewissern Sie sich, dass die Datei `altovadb.jar` im Ordner `C:\Program Files\Altova\Common 2012\jar` vorhanden ist.
5. Stellen Sie wie im Abschnitt oben unter *Herstellen einer Verbindung über JDBC* beschrieben, eine Verbindung zu einer Datenbank her.

Anmerkung: Wenn eine Datenbankverbindung über JDBC erfolgt, ist aufgrund mangelnder vom Treiber zurückgegebenen Informationen (i) die Bearbeitung von Daten in Tabellen ohne Primärschlüssel nicht möglich; (ii) und die Ausführung einer Datenbearbeitung im SQL-Editor funktioniert nicht.

Globale Ressourcen

Wenn Sie im Dialogfeld "Verbindung zu Datenquelle herstellen" auf die Schaltfläche **Globale Ressourcen** klicken, wird der Bereich "Globale Ressourcen" (*Abbildung unten*) geöffnet. Sie sehen darin alle globalen Ressourcen vom Typ Datenbank, die in der derzeit aktiven XML-Datei

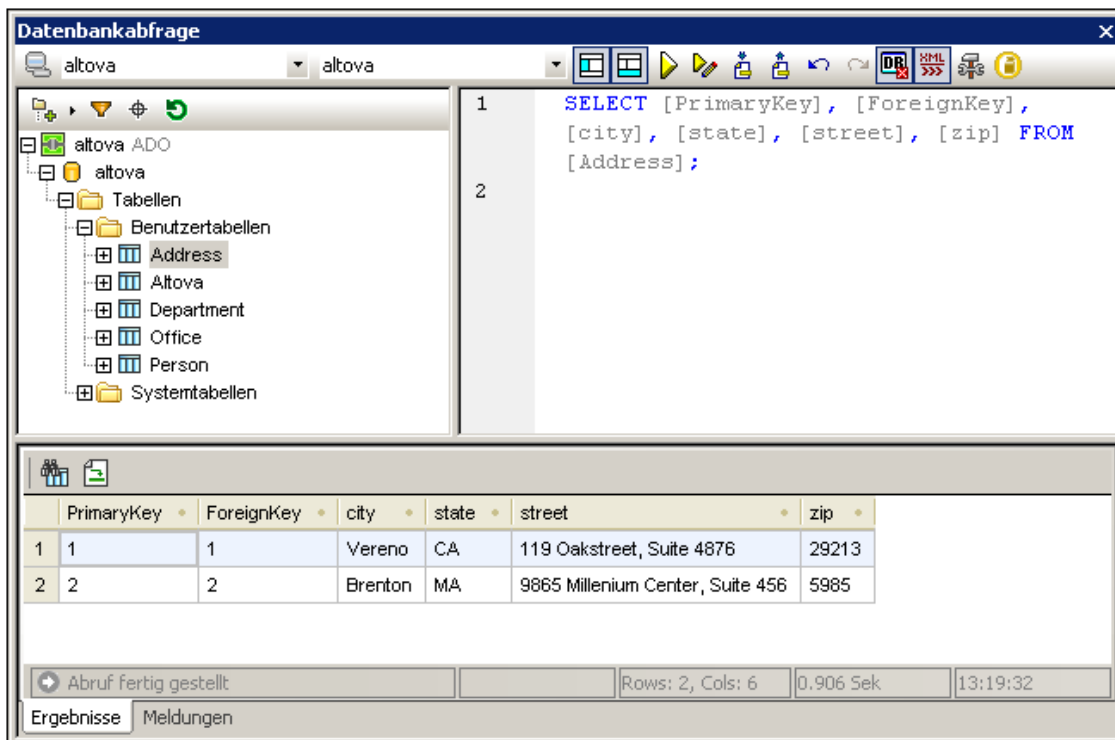
für globale Ressourcen definiert sind.



Wählen Sie die gewünschte globale Ressource aus und klicken Sie auf **Verbindung herstellen**. Die Verbindung zur Datenbank wird hergestellt. Informationen zu Dialogfeldern, die in der Folge angezeigt werden, finden Sie in der Beschreibung zum gerade ausgeführten Befehl.

18.9.2 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 Symbolleiste-Schaltfläche "Datenbank abfragen"  ein- und ausblenden.



Übersicht über das Fenster "Datenbankabfrage"

Das Fenster "Datenbankabfrage" besteht aus drei Teilen:

- einem [Browser-Fenster](#) links oben, in dem Verbindungsinformationen und Datenbanktabellen angezeigt werden.
- einem [Abfragefenster](#) rechts oben, in das die Abfrage eingegeben wird.
- einem mit Registern versehenen [Ergebnis-/Meldungsfenster](#). Im Fenster "Ergebnisse" werden die Ergebnisabfrage im sogenannten Ergebnistraster 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](#) 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](#). 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](#) angezeigt. Dieses Fenster kann je nach gewünschter

- Ansicht angepasst werden.
3. Im [Abfragefenster](#) wird eine [Abfrage](#) gemäß den für die gewünschte Datenbank benötigten Syntaxregeln eingegeben und ausgeführt.
 4. Die [Ergebnisse der Abfrage](#) können mit verschiedenen Filtern gefiltert, bearbeitet und wieder in der Datenbank gespeichert werden.

Datenquellen

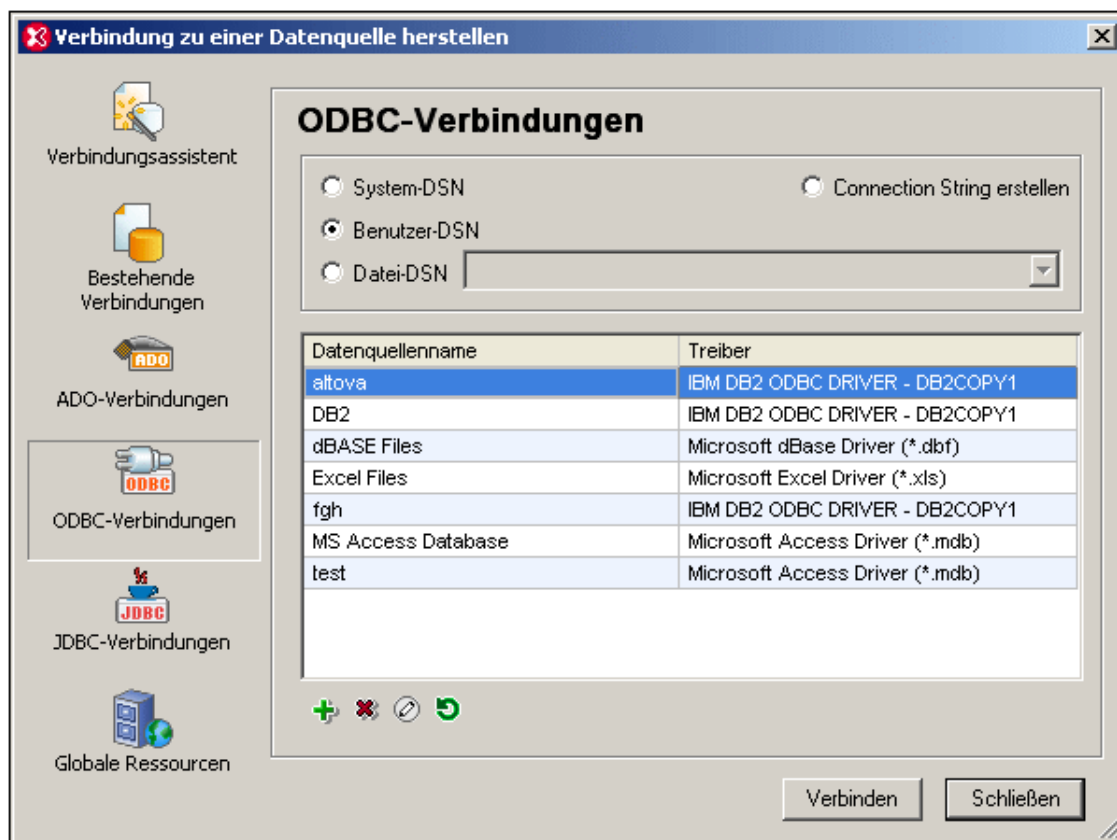
Zur Ausführung einer Datenbankabfrage müssen Sie zuerst eine Verbindung zur gewünschten Datenbank herstellen. In diesem Abschnitt wird beschrieben, wie Sie:

- 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" .



Eine Beschreibung zum Herstellen einer Verbindung zu einer Datenbank über das Dialogfeld "Verbindungsassistent" wird im Abschnitt [Herstellen einer Verbindung zu einer Datenquelle](#) beschrieben.

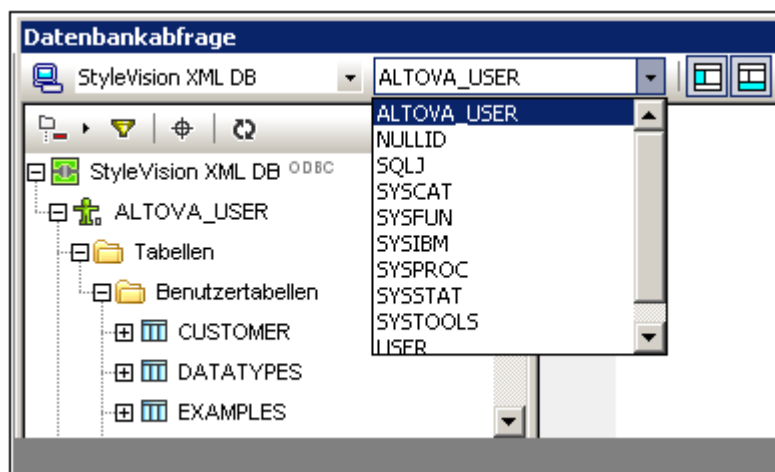
Es werden die folgenden Datenbanken unterstützt. Nachdem Sie eine Verbindung zur Datenbank hergestellt haben, stehen für die einzelnen unterstützten Datenbanken die folgenden Root-Objekte zur Verfügung:

Datenbank (nativ unterstützt)	Root-Objekt
MS SQL Server 2000, 2005 und 2008	database
MS SQL Server 2005 und 2008	schema
Oracle 9i, 10g und 11g	schema
MS Access 2003 und 2007	database
MySQL 4.x und 5.x	database
IBM DB2 8.x und 9	schema
Sybase 12	database
IBM DB2 for iSeries v 5.4	schema
PostgreSQL 8.0, 8.1, 8.2, 8.3	database

Hinweis: XMLSpy bietet vollständige Unterstützung für die oben aufgelisteten Datenbanken. Altova ist bestrebt, auch Unterstützung für andere ODBC/ADO-Datenbanken zu gewährleisten, doch wurde die erfolgreiche Herstellung der Verbindung und Verarbeitung der Daten nur bei den aufgelisteten Datenbanken getestet.

Auswahl der erforderlichen Datenquelle

Alle bestehenden Verbindungen und ihre Root-Objekte werden in zwei Auswahllisten in der Symbolleiste des Fensters "Datenbankabfrage" (*Abbildung unten*) aufgelistet.



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.

Browser-Fenster: Anzeigen der DB-Objekte

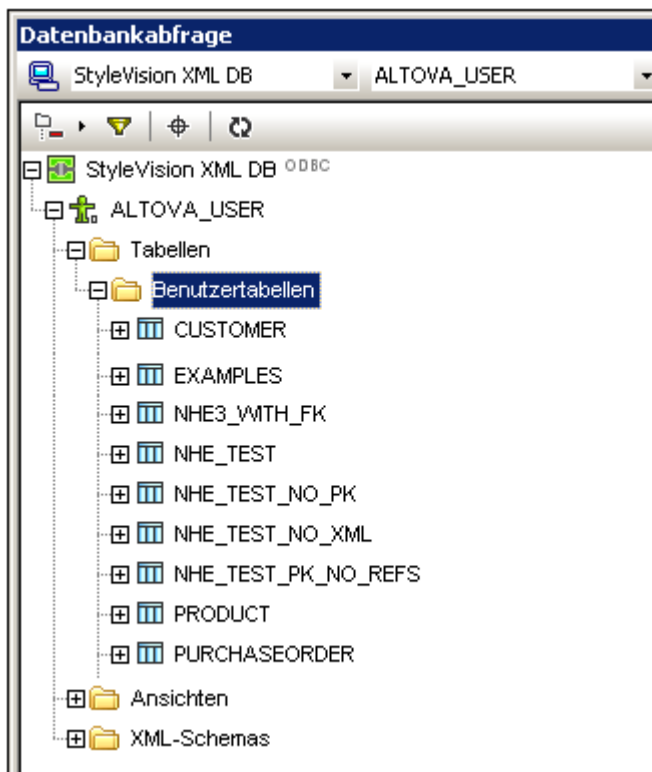
Im Browser-Fenster sehen Sie eine Übersicht über die Objekte in der ausgewählten Datenbank. Diese enthält Datenbank-Constraint-Informationen, z.B. ob es sich bei einer Spalte um einen Primär- oder Sekundärschlüssel handelt. In Datenbanken vom Typ IBM DB2 Version 9 sehen Sie im Browser zusätzlich registrierte XML-Schemas in einem separaten Ordner.

In diesem Abschnitt wird Folgendes beschrieben:

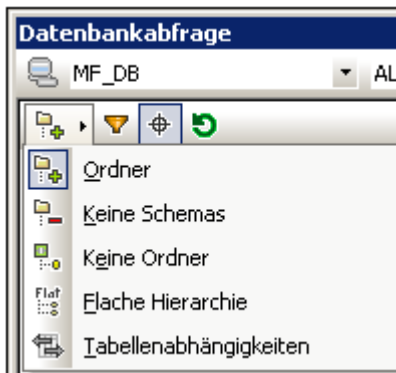
- Die im Browser-Fenster verfügbaren [Layouts](#).
- [Wie Datenbankobjekte gefiltert werden](#).
- [Wie Datenbankobjekte gesucht werden](#).

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.
- **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.

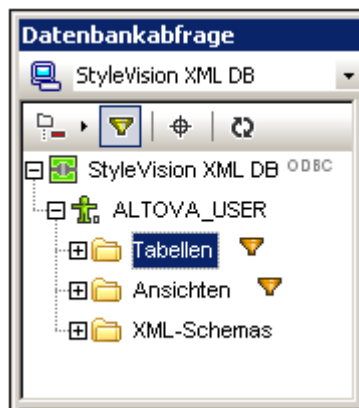
Um Tabellen nach Benutzer- und Systemtabellen zu sortieren, wechseln Sie ins Layout "Ordner", "Keine Schemas" oder "Flach", rechtsklicken Sie auf den Ordner "Tables" und wählen Sie den Befehl **Nach Benutzer- und Systemtabellen sortieren**. Die Tabellen werden in den Ordner "Benutzertabellen" und "Systemtabellen" in alphabetischer Reihenfolge sortiert.

Filtern von Datenbankobjekten

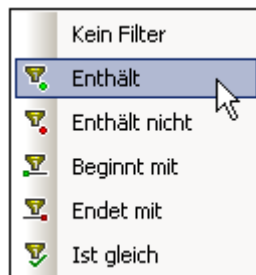
Im Browser-Fenster können Schemas, Tabellen und Ansichten (in allen Layouts mit Ausnahme des Layouts "Keine Ordner" und "Tabellenabhängigkeiten") nach Namen oder dem Teil eines Namens gefiltert werden. Die Objekte werden während der Eingabe der Zeichen gefiltert, dabei wird die Groß- und Kleinschreibung standardmäßig nicht beachtet.

So filtern Sie Objekte im Browser:

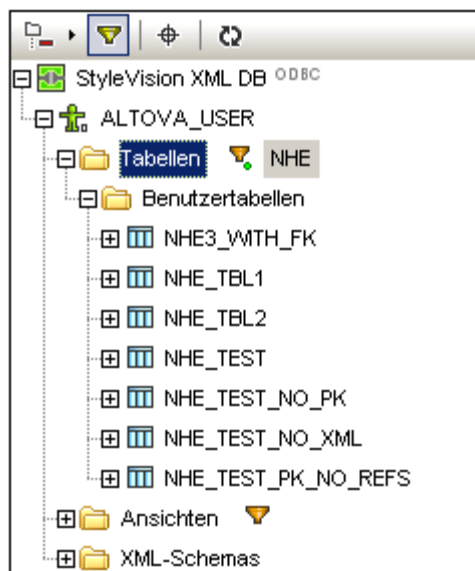
1. Klicken Sie in der Symbolleiste des Browser-Fensters auf die Schaltfläche "Ordnerinhalt filtern". Neben den Ordnern "Tables" und "Views" im ausgewählten Layout erscheinen Filtersymbole (*Abbildung unten*).



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.

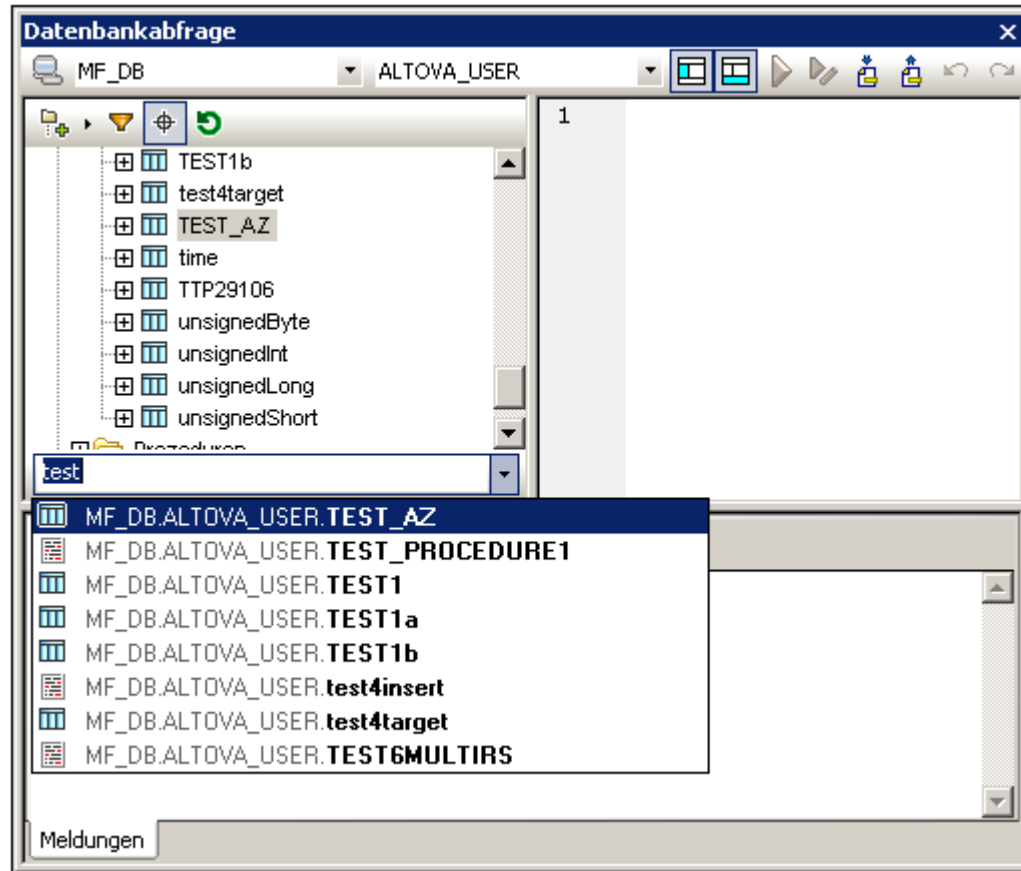


Suchen von Datenbankobjekten

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.



3. Klicken Sie auf das Objekt in der Liste, um es im Browser zu sehen.

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](#). Eine Anleitung zum Arbeiten mit Abfragen finden Sie im nächsten Abschnitt, [Abfragefenster: Arbeiten mit Abfragen](#). 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

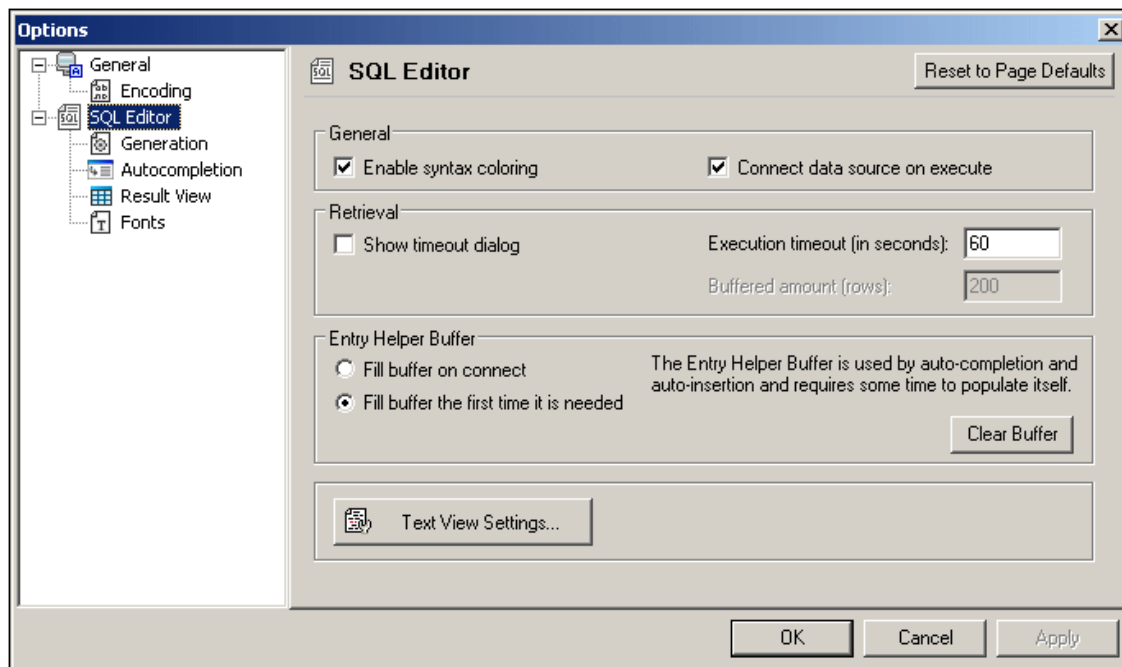
Für die Arbeit mit dem SQL Editor werden die folgenden Schaltflächen in der Symbolleiste des

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 (auf dem Register "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 Ergebnistraster 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

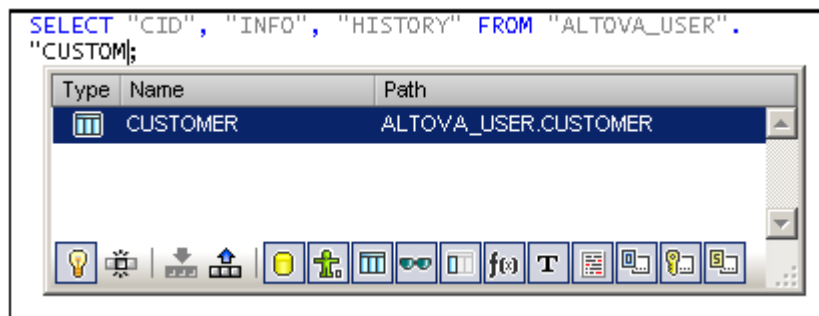
Wenn Sie in der Datenbankabfrage-Symbolleiste auf die Schaltfläche "Optionen" klicken, wird das Dialogfeld "Optionen" angezeigt (*Abbildung unten*). Im linken Bereich können Sie eine Einstellungsseite auswählen. Auf der jeweiligen Seite können Sie Optionen ausgewählt werden. Klicken Sie auf die Schaltfläche **Seite auf Standardeinstellungen zurücksetzen**, um die Standardeinstellungen für die jeweilige Seite wiederherzustellen.



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. 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" 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 *ASemikola 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 | Autokomplettierung:** Die Autokomplettierungsfunktion schlägt Ihnen während der Eingabe SQL-Anweisungen aus verschiedenen SQL-Syntaxkategorien vor. Sie steht für die folgenden Datenbanken zur Verfügung: MS SQL Server 2000, 2005 und 2008, MS Access 2003 und 2007 und IBM DB2 v.9. Wenn die Option "Autokomplettierung" aktiviert ist, erscheint das Fenster "Autokomplettierung" mit Vorschlägen für die Autokomplettierung (*Abbildung unten*). Wählen Sie den

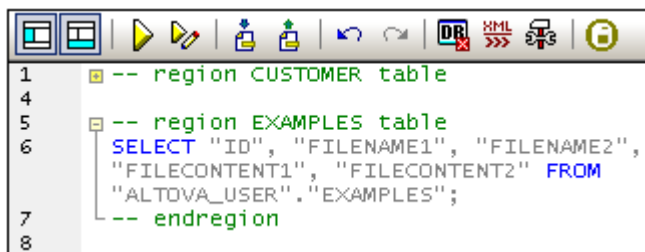
gewünschten Eintrag aus, um ihn einzufügen. Sie können festlegen, ob das Autokomplettierungs-Popup-Fenster nach einer bestimmten Zeitspanne, die Sie im Fenster *Triggering Auto-completion* einstellen können, automatisch aufgerufen werden soll, oder ob Sie das Popup-Fenster manuell aufrufen müssen. Außerdem können Sie auch auswählen, mit welchen Tasten der ausgewählte Vorschlag eingefügt werden soll. Der SQL-Editor kann anhand von Sprachstatistiken intelligente Autokomplettierungsvorschläge machen. Wenn diese Funktion aktiviert ist, werden häufig verwendete Einträge ganz oben in der Liste der Vorschläge angezeigt. Beachten Sie die Schaltflächen am unteren Rand des Autokomplettierungsfensters. Mit der Schaltfläche *Kontextsensitiver Vorschlag* legen Sie fest, ob nur Einträge angezeigt werden sollen, die in diesem Kontext relevant sind, oder ob alle möglichen Einträge mit dieser Schreibweise angezeigt werden sollen. Mit der Schaltfläche *Einzelmodus* können Sie auf eine Categorieschaltfläche klicken, um nur diese Kategorie auszuwählen. Mit der Schaltfläche *Alle Kategorien definieren* werden alle Kategorien ausgewählt. Um eine Kategorie von der Auswahl auszunehmen, klicken Sie anschließend auf die Schaltfläche für diese Kategorie. Mit der Schaltfläche *Alle Kategorien löschen* wird die Auswahl für alle Kategorien aufgehoben. Die anderen Schaltflächen sind die Schaltflächen für die verschiedenen Kategorien.



- **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, rechteckigen 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 Zeilenkommentar 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 Lesezeichen 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**.

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](#). Wenn Sie den Befehl **Zur Datenbearbeitung ausführen** gewählt haben, können die abgerufenen Daten im Ergebnisregister [bearbeitet werden](#). Meldungen über die Ausführung der Anweisung werden auf dem [Meldungsregister](#) angezeigt.

Ergebnisse und Meldungen

Das Ergebnis-/Meldungsfenster hat zwei Register:

- Auf dem [Register "Ergebnisse"](#) werden die von der Abfrage abgerufenen Daten angezeigt.
- Auf dem Register [Register "Meldungen"](#) 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.

	CID	INFO	HISTORY
1	1000	 <?xml version="1.0" encoding="UTF-8" ?><n1:customerinfo ...	 [NULL]
2	1001	 <?xml version="1.0" encoding="UTF-8" ?><customerinfo ...	 [NULL]
3	1002	 <?xml version="1.0" encoding="UTF-8" ?><customerinfo ...	 [NULL]
4	1003	 <?xml version="1.0" encoding="UTF-8" ?><customerinfo ...	 [NULL]
5	1004	 <?xml version="1.0" encoding="UTF-8" ?><customerinfo ...	 [NULL]
6	1005	 <?xml version="1.0" encoding="UTF-8" ?><customerinfo ...	 [NULL]

 Abruf fertig gestellt Rows: 6, Cols: 3 0.110 sec 15:42:49

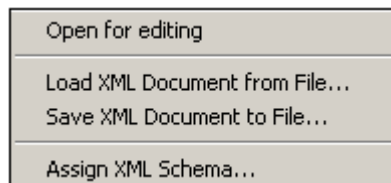
Ergebnisse Meldungen

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.
- **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:** Wenn die Abfrage zur Datenbearbeitung ausgeführt wurde und wenn es sich bei einem editierbaren Feld um ein XML-Feld handelt (derzeit werden nur IBM DB2 XML-Datenbanken unterstützt), wird bei Klick auf das Symbol für editierbares XML  im Ergebnistraster 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"](#) 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](#) näher behandelt.

- **NULL definieren, Standard definieren, Änderungen für diese Zelle rückgängig machen:** Wenn die Abfrage zur Bearbeitung ausgeführt wurde, erhalten Sie bei Rechtsklick in

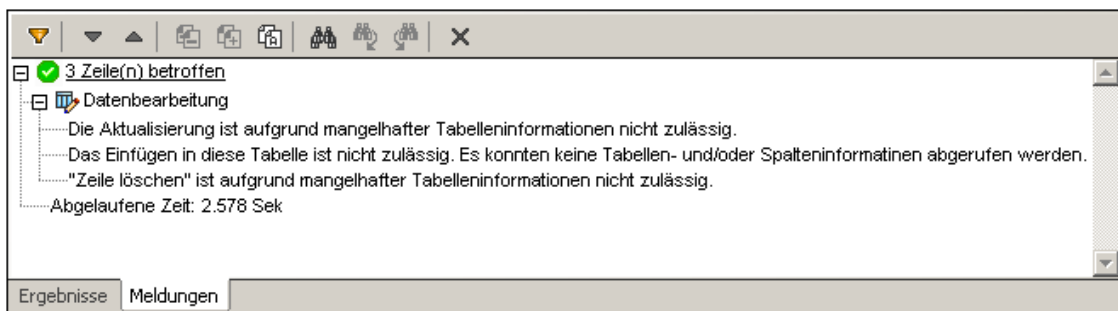
eine Zelle Zugriff auf Befehle zum Setzen eines Null-Werts oder, falls definiert, eines Spalten-Standardwerts für diese Zelle. An einer Zelle vorgenommene Änderungen können mit dem Befehl **Änderungen für diese Zelle rückgängig machen** rückgängig gemacht werden; der aktuelle bearbeitete Wert wird durch den Wert aus der Datenbank ersetzt.

Das Ergebnisregister hat die folgenden Symboleleistenschaltflächen:

	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 Ergebnistraster hinzu.
	Zeile löschen	Löscht die aktuelle Zeile im Ergebnistraster.
	Änderungen am Ergebnistraster rückgängig machen	Macht alle Änderungen am Ergebnistraster rückgängig.
	Übernehmen	Übernimmt die im Ergebnistraster 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.

18.9.3 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](#)), und (ii) zum Zuweisen von Schemas für die XML-Dateivalidierung ([Zuweisen eines XML-Schemas](#)).

Für beide Funktionen muss eine Verbindung zur jeweiligen IBM DB2-Datenbank bestehen. Wie man eine Verbindung zur Datenbank herstellt, ist im Abschnitt [Herstellen einer Verbindung zur Datenquellen](#) 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"](#) enthält wichtige Funktionen für die Arbeit mit XML-Dateien in IBM DB2-Datenbanken. Dazu gehören Funktionen zum Bearbeiten, Laden von XML-Dateien in Datenbank-XML-Dateien, externen Speichern von DB-XML-Dateien und Zuweisen von XML-Schemas zu DB-XML-Dateien.

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*).

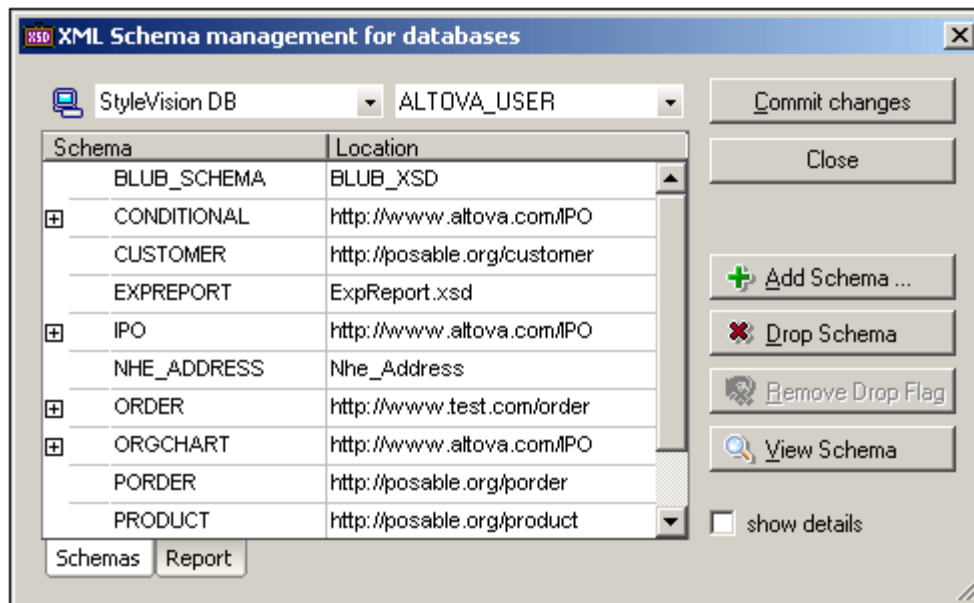


Um die Schemas einer IBM DB2-Datenbank zu verwalten, müssen Sie zuerst eine Verbindung

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 [Menü "DB" | Herstellen einer Verbindung zu einer Datenquelle](#) 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.

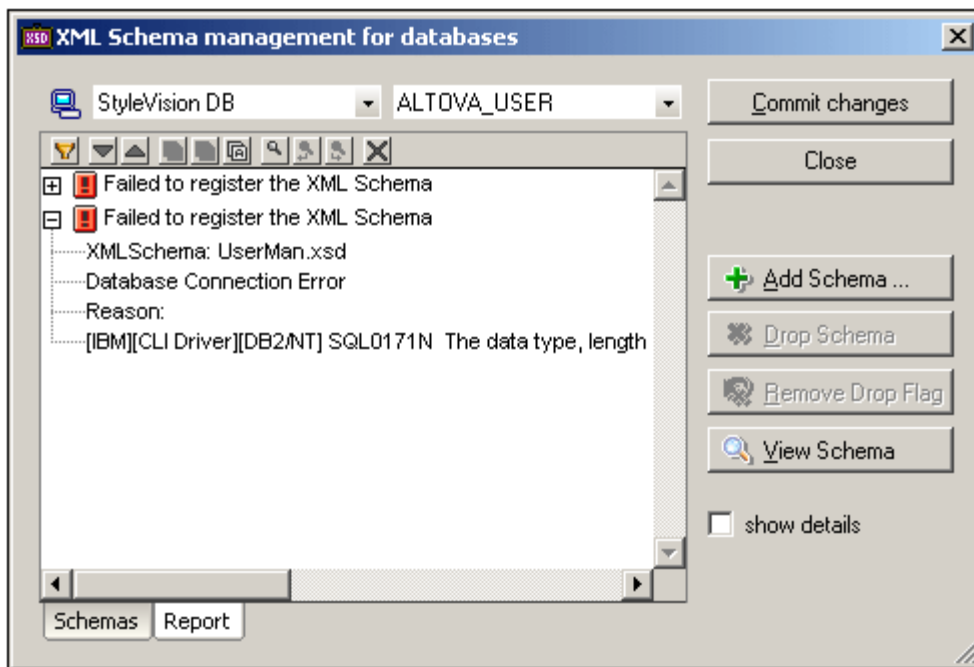
Um ein Schema zu löschen, wählen Sie das Schema im Hauptfenster aus und klicken Sie auf die Schaltfläche **Schema löschen**. Dem Schema wird ein Drop Flag zugewiesen, um anzuzeigen, dass es bei der Übernahme der nächsten Änderungen gelöscht wird. Sie können das Drop Flag löschen, indem Sie das so gekennzeichnete 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.

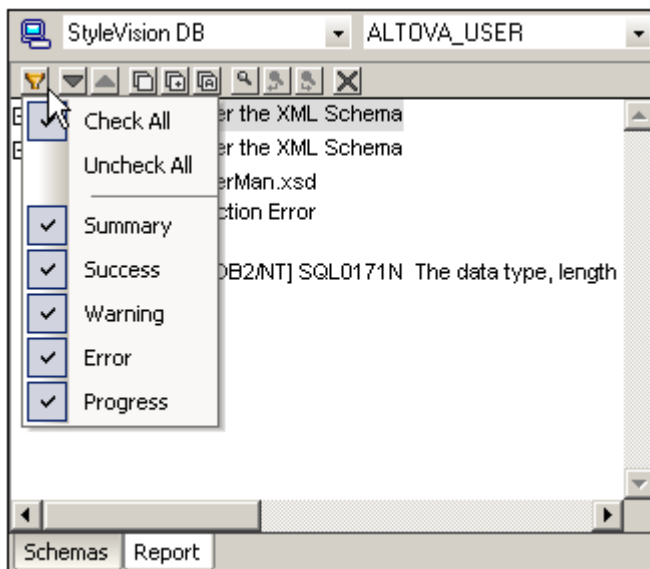
Wenn Sie auf die Schaltfläche "Schema anzeigen" klicken, wird das Schema in XMLSpy geöffnet. Um das Dialogfeld "XML-Schemaverwaltung" zu schließen, klicken Sie auf die Schaltfläche **Schließen**.

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*).



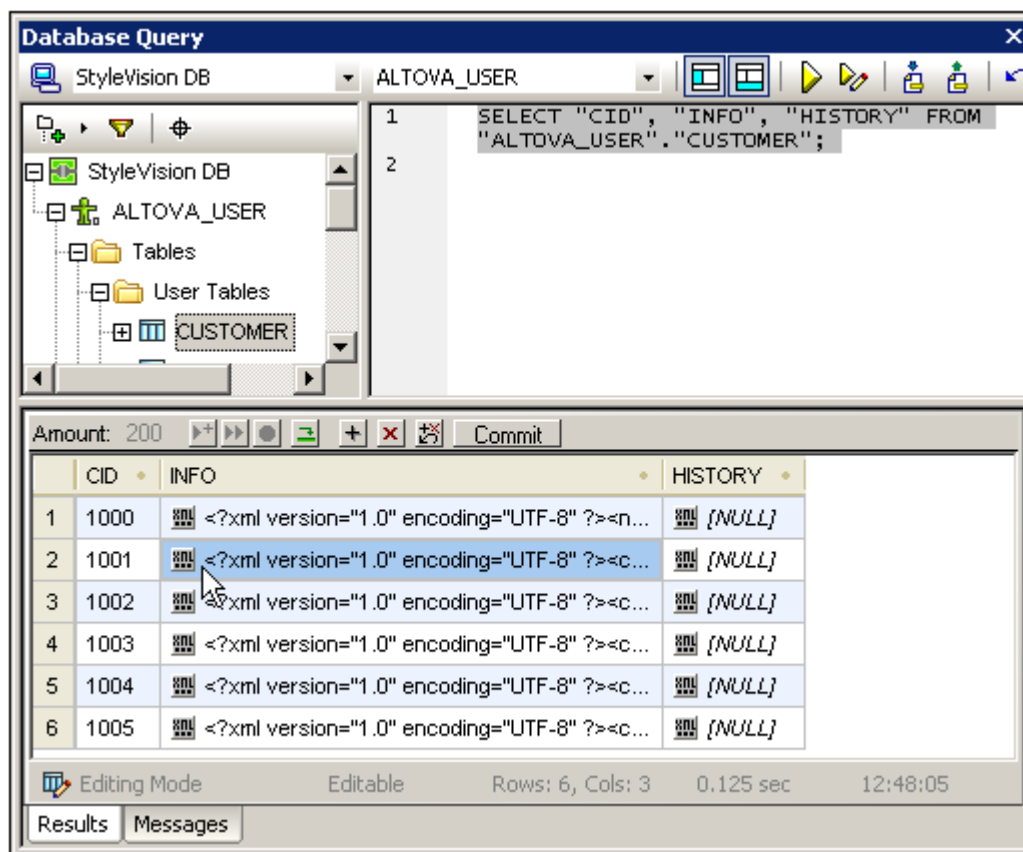
Mit Hilfe der Schaltfläche **Filter** können Sie bestimmte Arten von Meldungen ein oder ausblenden. Mit Hilfe der Schaltflächen **Nächste** und **Vorherige** wird die jeweils vorige bzw. nächste Meldung in der Liste markiert. Meldungen können auch mit oder ohne Ihre Sub-Einträge in die Zwischenablage 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.

XML-Schema zuweisen

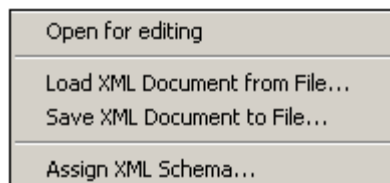
Mit dem Befehl "XML-Schema zuweisen" wird einer XML-Datei, die zur Bearbeitung über das Ergebnisraster des Fensters "Datenbankabfrage" geöffnet wurde, ein XML-Schema zugewiesen. Nach Zuweisung des Schemas kann die XML-Datei gegen das zugewiesene Schema validiert werden. Die Zuweisung wird in die Datenbank 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*).



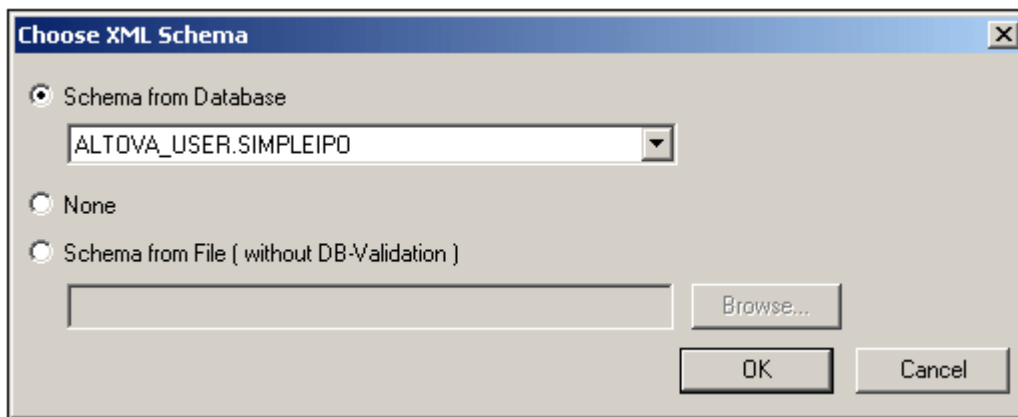
Wenn Sie auf die Schaltfläche "XML"  klicken, erscheint das folgende Menü.



Wenn Sie den Befehl **Zur Bearbeitung öffnen** auswählen, wird das XML-Dokument in XMLSpy geöffnet, wo es bearbeitet werden kann.

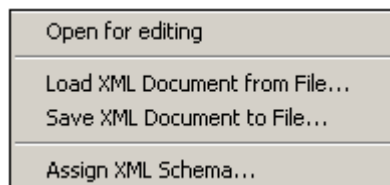
Zuweisen eines Schemas zur Datenbank-XML-Datei

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 Ergebnistraster 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.

18.9.4 SQL Server

Der Befehl **SQL-Server** enthält ein Untermenü, das den Befehl "XML-Schemas verwalten" enthält.

XML-Schemas verwalten

Mit dem Befehl **XML-Schemas verwalten** können Schemas aus einzelnen Datenbank-Schemas in einer SQL-Server-Datenbank hinzugefügt und gelöscht werden. Dazu müssen folgendes tun:

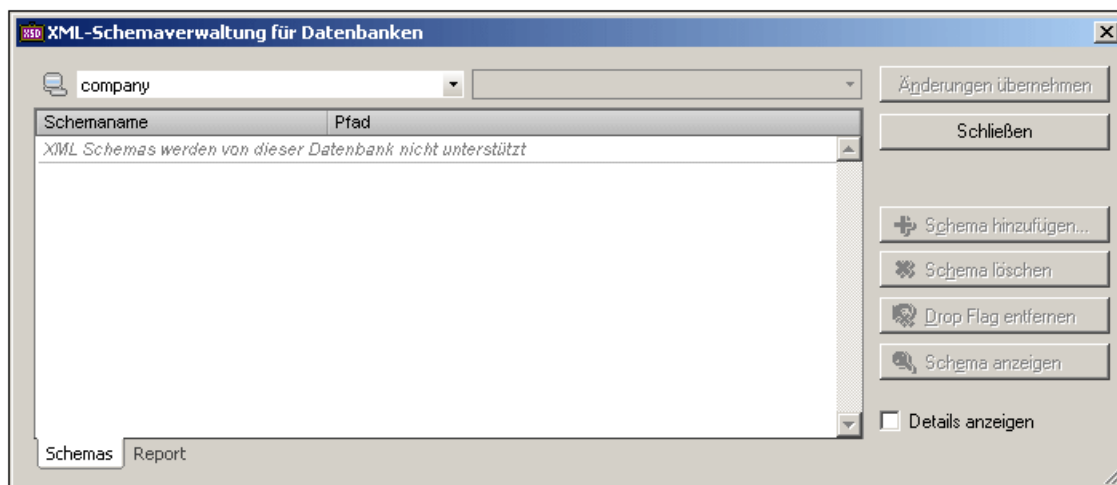
- eine Verbindung zu einer SQL Server-Datenbank herstellen
- das/die Datenbank-Schema(s) auswählen, für das/die XML-Schemas hinzugefügt oder gelöscht werden sollen und

- anschließend die Schemaverwaltungsaktionen durchführen.

Diese Schritte sind im Folgenden beschrieben.

Herstellen einer Verbindung zur IBM DB2-Datenbank

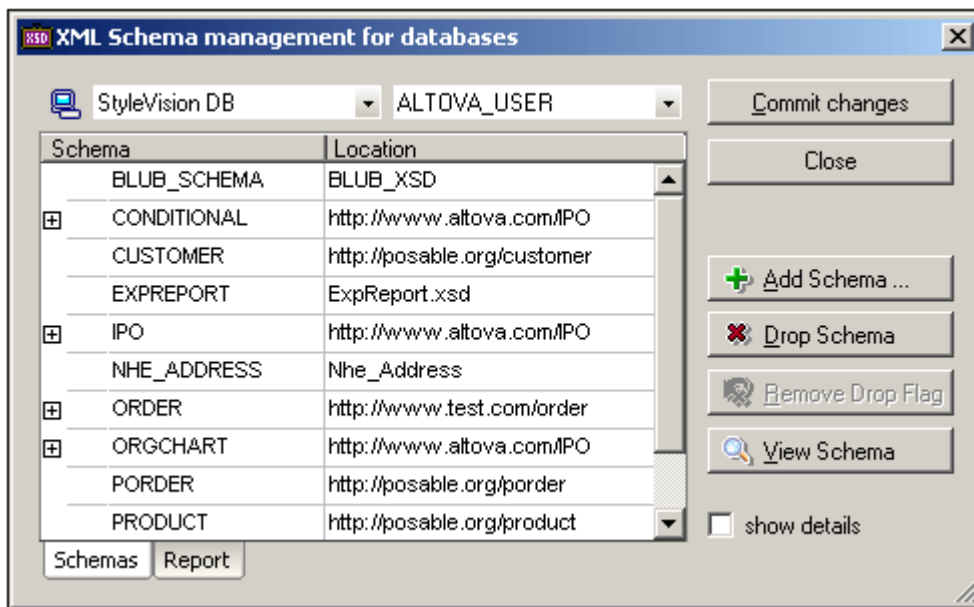
Wenn Sie den Befehl **XML-Schemas verwalten** auswählen, wird das Dialogfeld "XML-Schemaverwaltung für Datenbanken" aufgerufen (*Abbildung unten*).



Um die Schemas einer SQL Server-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 [Menü "DB" | Herstellen einer Verbindung zu einer Datenquelle](#) beschrieben.

Anzeigen einer Liste von XML-Schemas

Nachdem die Verbindung zur SQL Server-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 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.

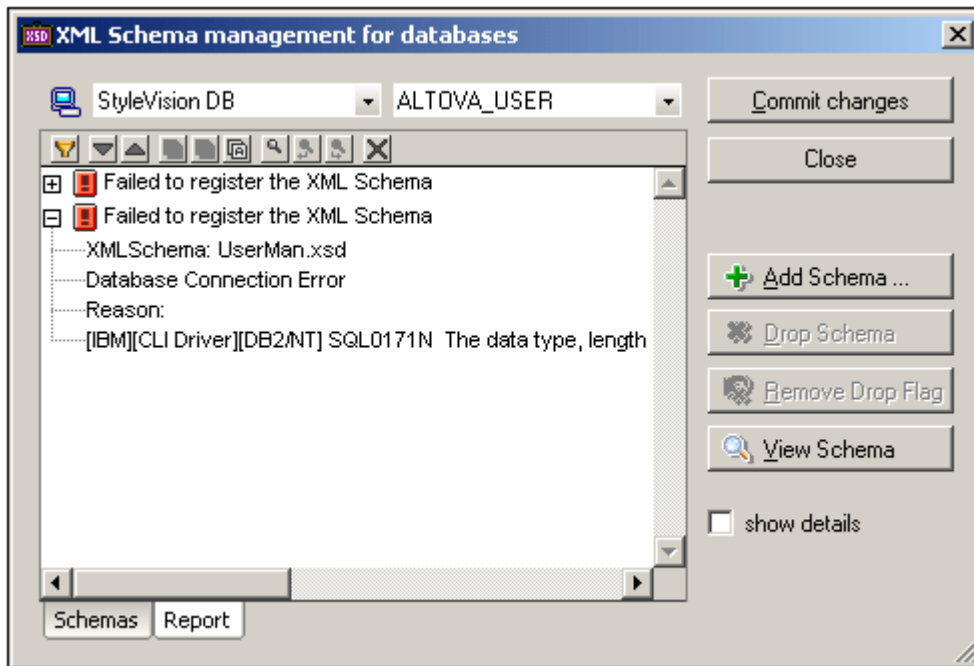
Um ein Schema zu löschen, wählen Sie das Schema im Hauptfenster aus und klicken Sie auf die Schaltfläche **Schema löschen**. Dem Schema wird ein Drop Flag zugewiesen, um anzuzeigen, dass es bei der Übernahme der nächsten Änderungen gelöscht wird. Sie können das Drop Flag löschen, indem Sie das so gekennzeichnete 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**.

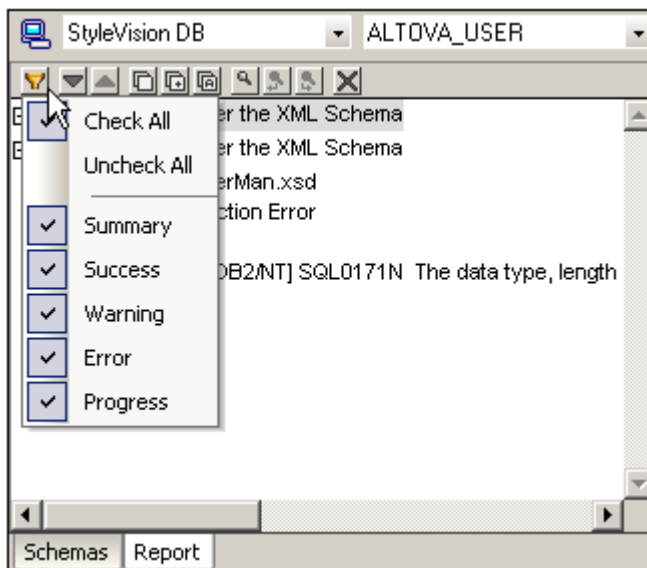
Berichte

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.

18.9.5 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:

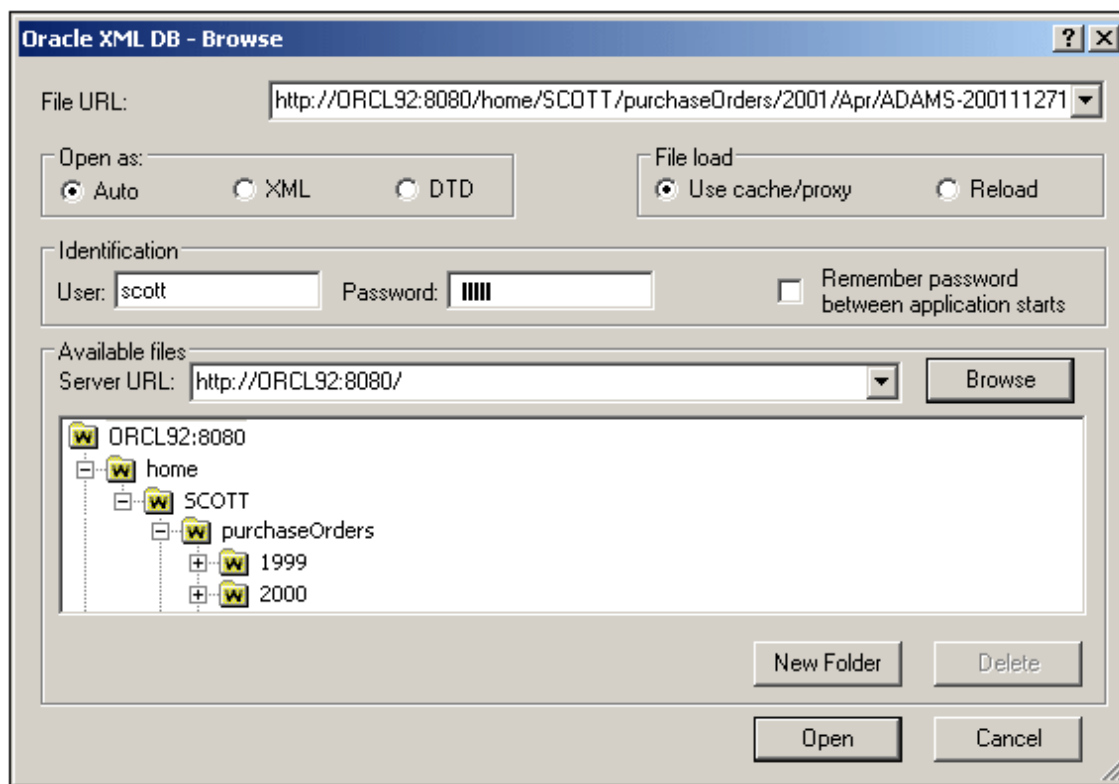
- Downloaden und installieren Sie XMLSpy
- Installieren Sie Oracle Server (falls nötig)
- Erzeugen Sie eine Oracle Datenbank

Unter http://otn.oracle.com/sample_code/tech/xml/xmlldb/XDBBasicDemo.pdf finden Sie eine Demoinstallation für eine Oracle XML DB. In diesem Dokument wird beschrieben, wie die Oracle Software installiert und für XMLSpy konfiguriert wird.

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.



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.

18.10 Menü "Konvertieren"

Mit XMLSpy stehen Ihnen leistungsfähige Datenaustauschfunktionen zur Verfügung:

- Importieren und Exportieren von Text-, Word-Prozessor-, Datenbanken- und XML-Dateien.
- [Importieren von Datenbankdaten](#) auf Basis eines vorhandenen Schemas.
- Erzeugen von [XML-Schemas](#), die auf einer bereits existierenden Datenbank **basieren**.
- Erzeugen einer [Datenbankstruktur](#), die auf einem existierenden Schema basiert.

Diese Funktionen stehen im Menü **Konvertieren** zur Verfügung.

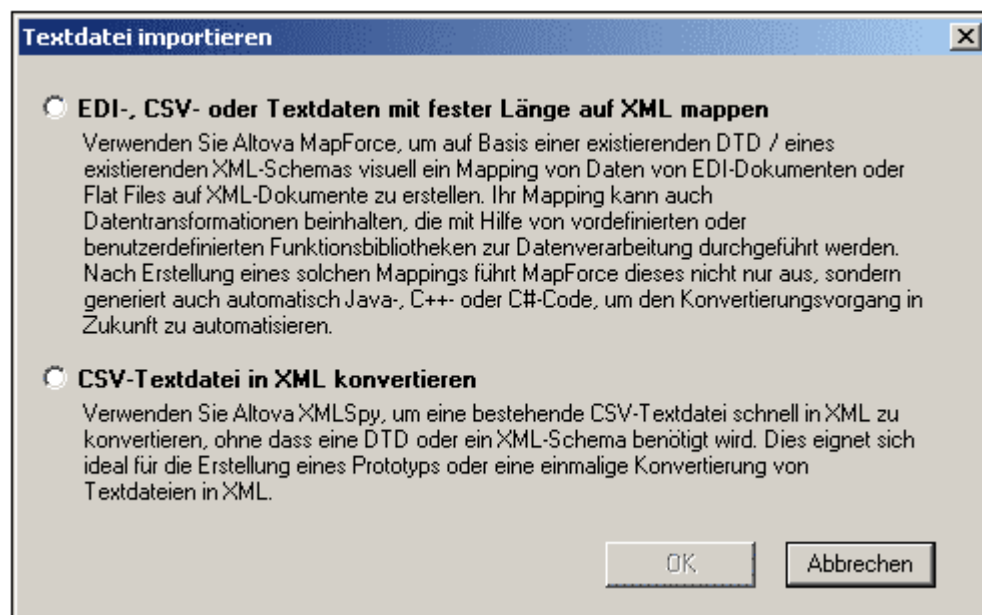


18.10.1 Textdatei importieren

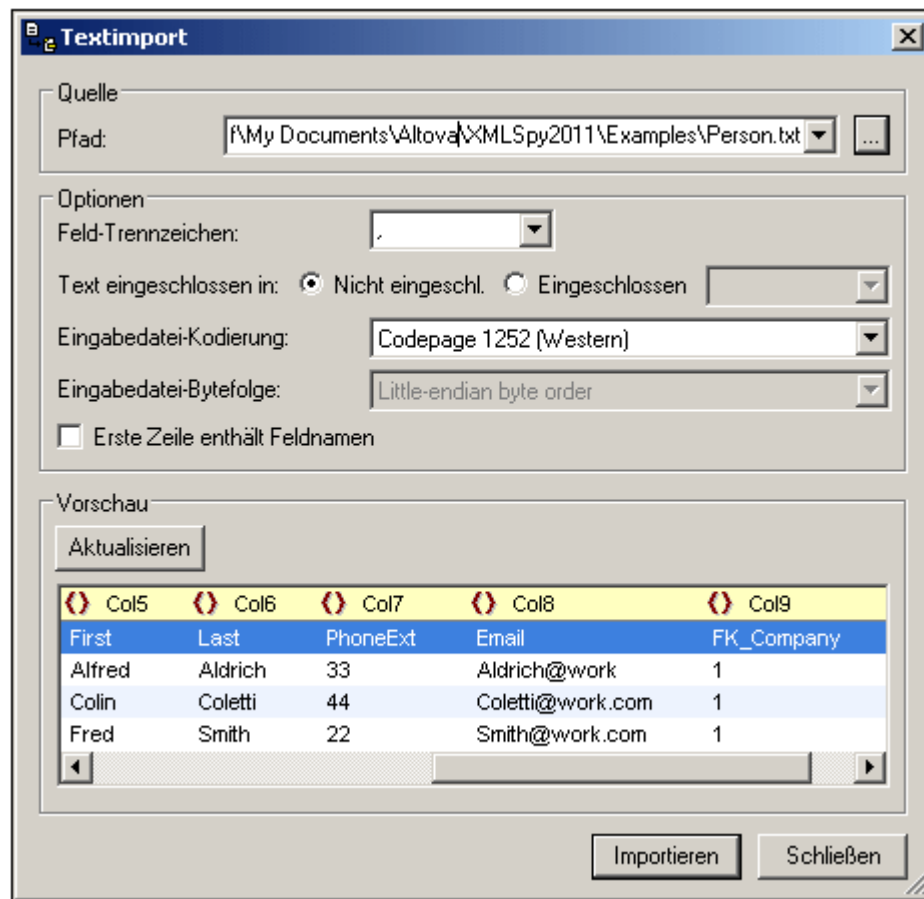


Mit diesem Befehl lässt sich jede Datei mit **strukturierten Text** in XMLSpy importieren und sofort ins XML-Format konvertieren. Diese Funktion ist nützlich, um Altdatenbestände aus älteren Systemen zu importieren. Die Schritte zum Importieren von Daten in einer Textdatei als XML-Dokument sind unten beschrieben.

1. Wählen Sie den Menübefehl **Konvertieren | Textdatei importieren**. Das folgende Dialogfeld wird angezeigt:



2. Wählen Sie eine der folgenden Optionen:
 - EDI-, CSV- oder Textdaten mit fester Länge auf XML mappen (Sie müssen Altova MapForce installiert haben, um diese Option auszuwählen)
 - CSV-Textdatei in XML konvertieren
3. Klicken Sie auf **OK**. Daraufhin wird das Dialogfeld "Textimport" (*Abbildung unten*) angezeigt. Die Verwendung dieses Dialogfelds wird unten 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 Felder 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.

Eingabedatei-Encoding

Die Daten werden ins [Unicode](#) 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.

Eingabedatei-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 werden im XML-Dokument die Element- oder Attributnamen sein. 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 ist `Col1` ein Element, `Col2` ein Attribut und `Col3` wird nicht importiert.

Wenn Sie mit den Optionen zufrieden sind, klicken Sie auf **Import**. Die importierten Daten werden in ein XML-Dokument konvertiert, das in der Grid-Ansicht angezeigt wird.

18.10.2 Datenbankdaten importieren

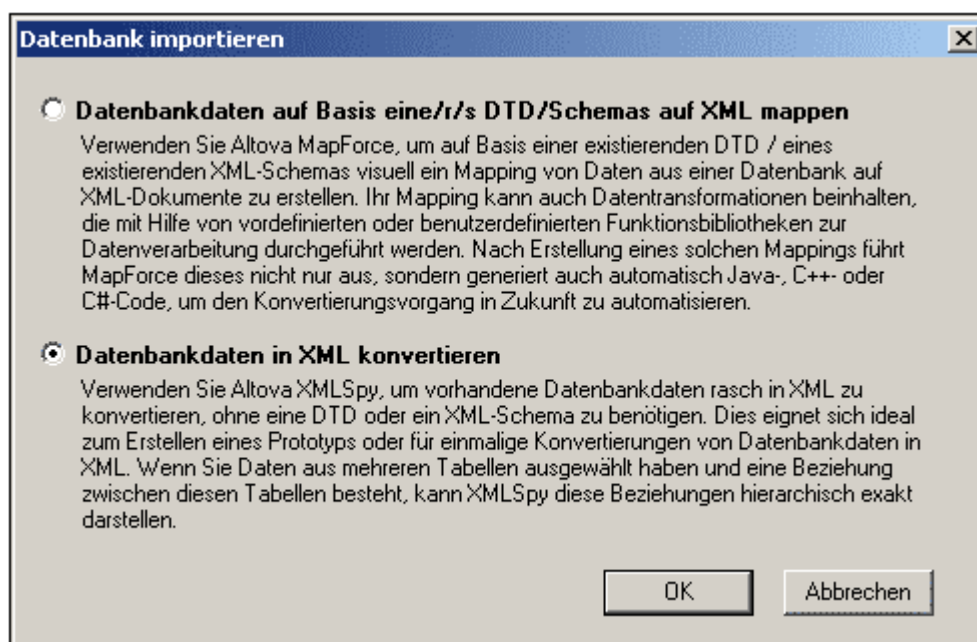


Mit Hilfe des Befehls **Datenbankdaten importieren** können Sie Daten aus verschiedenen Datenbanktypen in eine XML-Datei importieren. Der Import erfolgt in zwei Schritten:

1. [Aufbau einer Verbindung zur Datenbank](#)
2. [Auswahl der zu importierenden Daten](#)

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.

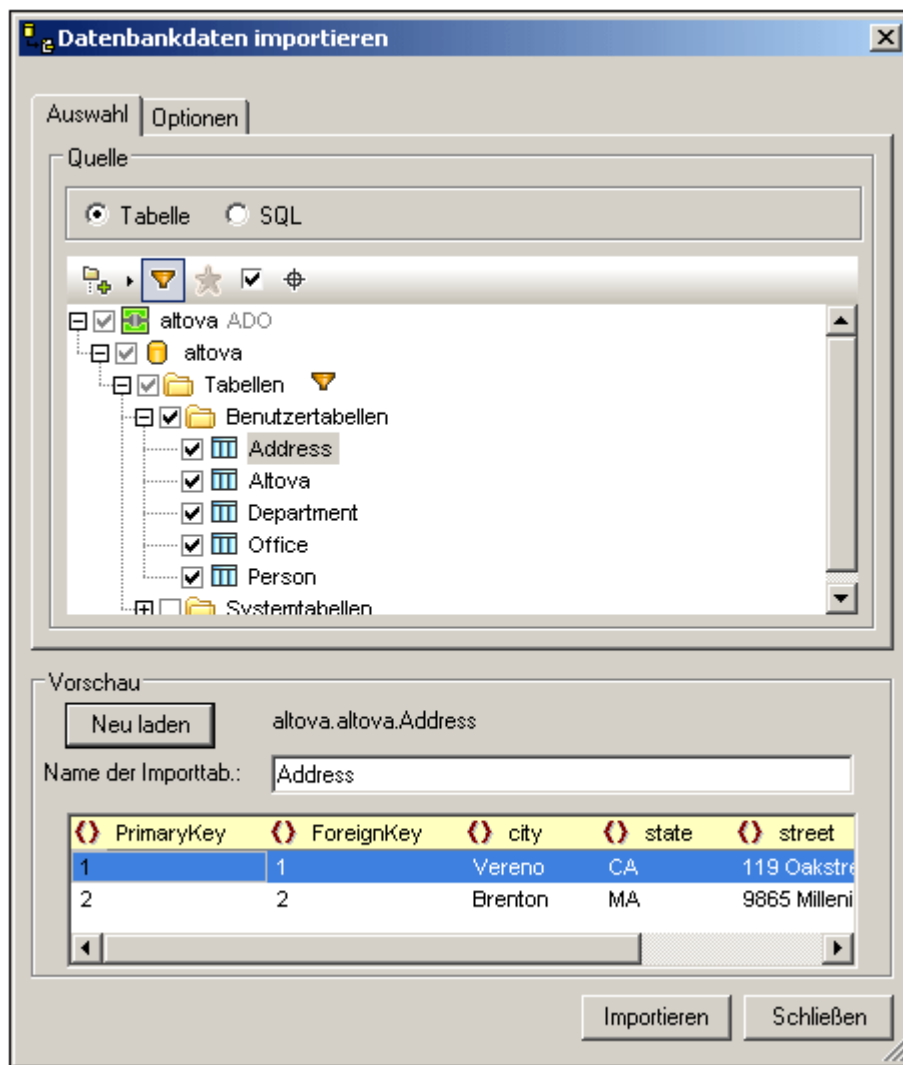


2. Wählen Sie die Option "Datenbankdaten in XML konvertieren" aus und klicken Sie auf **OK**. Daraufhin wird das Dialogfeld "Connect to a Data Source" geöffnet. (Wenn Sie die Option "Datenbankdaten auf Basis eine/r/s DTD/Schemas auf XML mappen" auswählen, muss Altova MapForce installiert sein, damit das Mapping durchgeführt werden kann.)
3. Stellen Sie im Dialogfeld [Connect to Data Source](#) eine Verbindung zur Datenbank her. Eine Anleitung zum Herstellen einer Verbindung zu unterschiedlichen Datenbanktypen finden Sie im Abschnitt [Herstellen einer Verbindung zu einer Datenquelle](#).
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

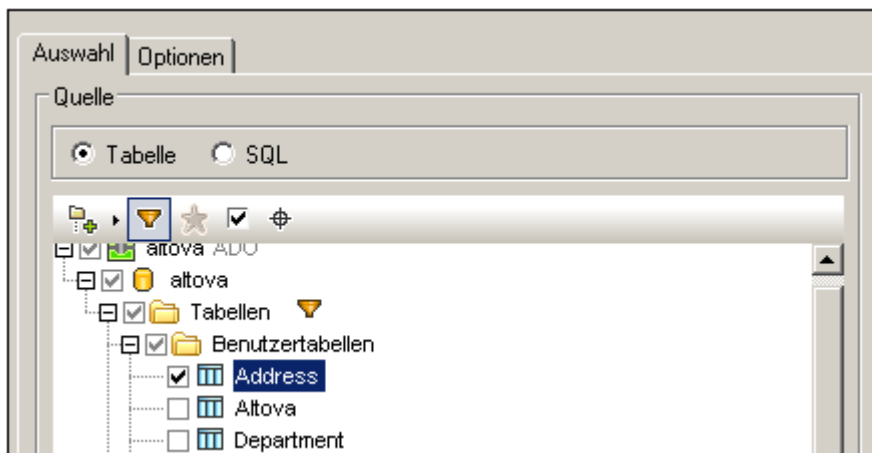
Das Dialogfeld "Datenbankdaten importieren" zum Auswählen der Daten und Einstellen der Importoptionen besteht aus zwei Teilen (*die in den Abbildungen unten separat gezeigt werden*):

- 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.



Register "Auswahl"

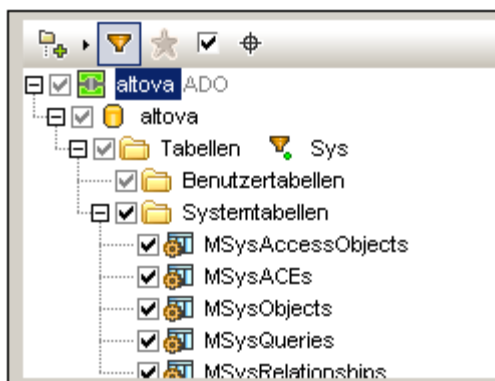
Auf dem Register "Auswahl" (*Abbildung oben*) werden im Bereich Quelle (*Abbildung unten*) entweder die Tabellen der Datenbank angezeigt oder Sie sehen eine editierbare SQL-Anweisung zur Auswahl der gewünschten Tabellen, wobei die einzelnen Ansichten durch Aktivieren des entsprechenden Optionsfelds ausgewählt werden.



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 wird im Vorschauenster angezeigt. Die Tabellenauswahl kann im Vorschaubereich noch weiter gefiltert werden (*siehe unten*).

Die Datenbankstruktur kann unterschiedlich angezeigt und gefiltert werden. Über die Schaltfläche "Layout"  im Bereich "Quelle" können Sie die Datenbankobjekte in: (i) Ordner nach Objekttyp; (ii) Ordner nach Objekttyp oder ohne Schemaordner; (iii) in eine Hierarchie aber ohne Ordner; und (iv) in Tabellenkategorien basierend auf Beziehungen mit anderen Tabellen gliedern.

Wenn Sie auf die Schaltfläche "Ordnerinhalt filtern"  klicken, wird ein Filter auf den ausgewählten Ordner angewendet (*in der Abbildung unten auf den Ordner "Tabellen"*). Wenn Sie in der Tabelle auf das Filtersymbol klicken, erscheint ein Menü mit einer Liste von Filteroptionen. In der Abbildung unten wurde der Filter so eingestellt, dass Objekte angezeigt werden, die in ihrem Namen den Text `sys` enthalten. Die Ansicht wird entsprechend gefiltert.



Wenn Sie auf die Schaltfläche "Objektsuche"  klicken, wird ein Textfeld angezeigt, das wie ein Suchfeld funktioniert. 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.

	Primary	Foreign	city	st	street	zip
1	1	Vereno	CA	119 Oakstreet, Suite 4876	29213	
2	2	Brenton	MA	9865 Millenium Center, Suite	5985	

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](#).

18.10.3 MS Word Dokument Importieren



Mit diesem Befehl können Sie jedes beliebige **Word Dokument importieren und in ein XML-Format konvertieren**, wenn Sie Absatzformatvorlagen in Microsoft Word verwenden. Für diese Option wird Microsoft Word oder Microsoft Office (Version 97 oder 2000) benötigt.

Mit diesem Befehl öffnen Sie das Dialogfeld "Öffnen". Wählen Sie das zu importierende Word Dokument aus.

XMLSpy erzeugt automatisch ein XML-Dokument, das CSS Stylesheet enthält. Jeder Absatz in Word generiert ein XML-Element, dessen Name der Name des entsprechenden Absatzformats in Microsoft Word ist.

18.10.4 XML-Schema von DB-Struktur erzeugen

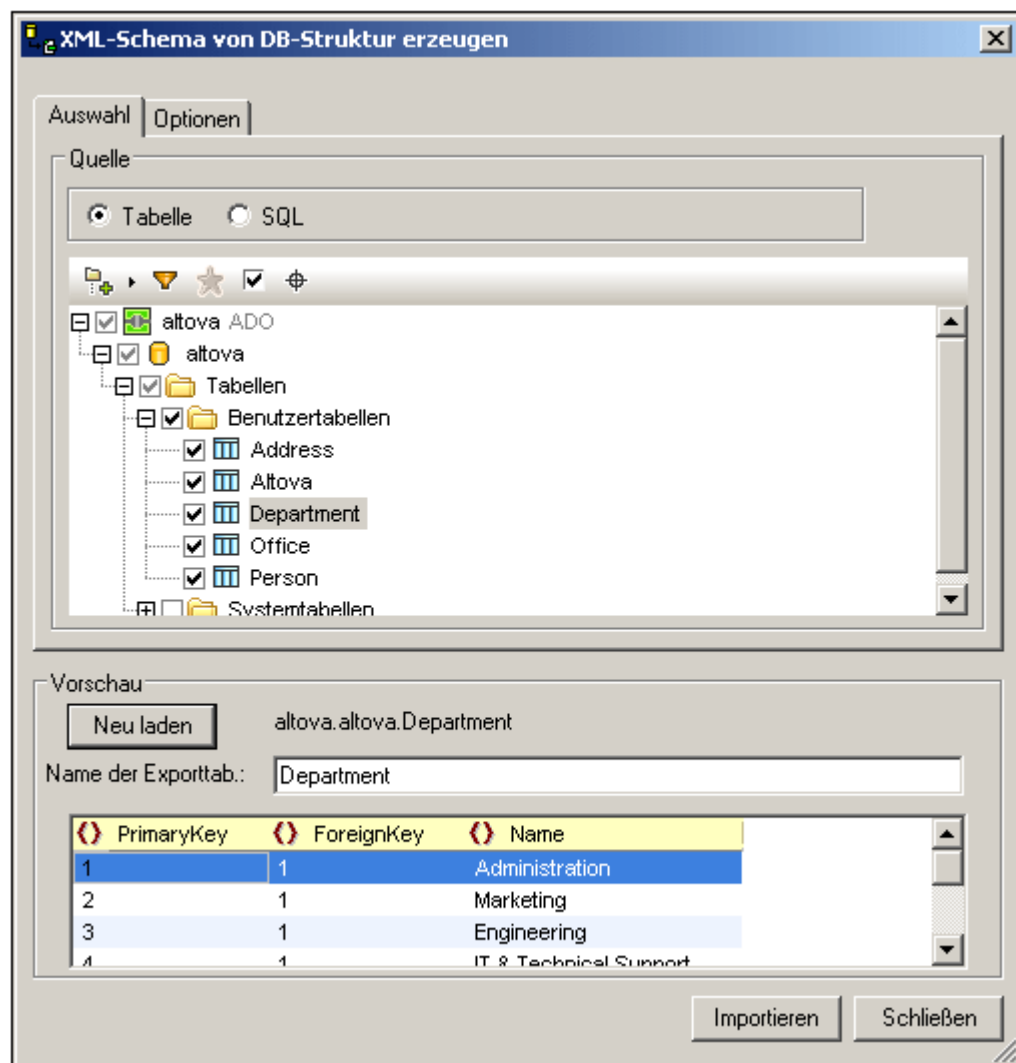


Mit Hilfe des Befehls **XML-Schema von DB-Struktur erzeugen** können Sie anhand der Struktur einer Datenbank ein XML-Schema erzeugen. Es werden die verschiedensten Datenbanktypen unterstützt. Die Erstellung des XML-Schemas erfolgt in zwei Schritten:

1. Aufbau einer Verbindung zur Datenbank
2. Auswahl der Optionen für die Datenbankdaten und Angabe des XML-Schemas

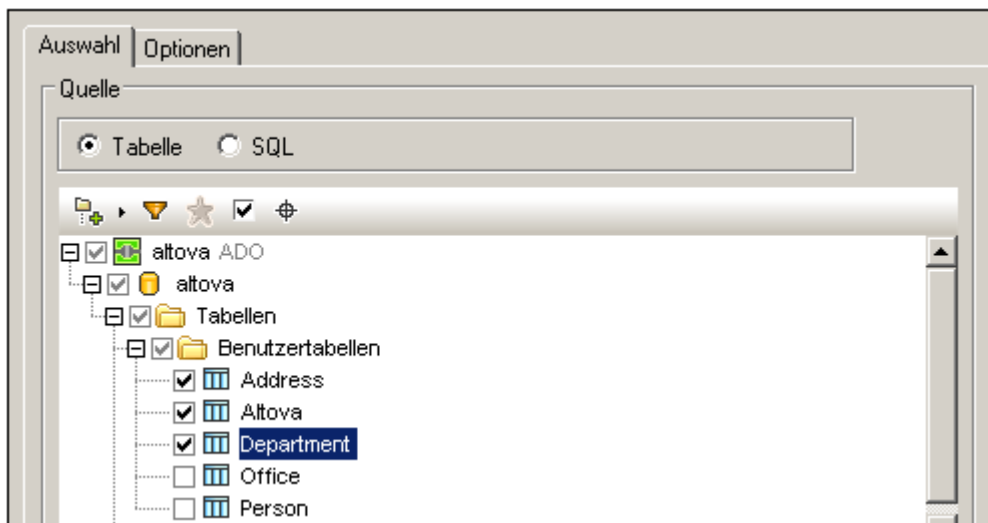
Um ein XML-Schema anhand einer DB-Struktur zu erzeugen, gehen Sie folgendermaßen vor:

1. Wenn Sie auf den Befehl **XML-Schema von DB-Struktur erzeugen** klicken, wird das Dialogfeld [Verbindung zur Datenquelle herstellen](#) aufgerufen.
2. Stellen Sie in diesem Dialogfeld eine Verbindung zur Datenbank her. Eine Anleitung zum Herstellen einer Verbindung zu verschiedenen Datenbanktypen finden Sie im Abschnitt [Herstellen einer Verbindung zu einer Datenquelle](#).
3. Nachdem Sie die Verbindung hergestellt haben, wird das Dialogfeld "XML-Schema von DB-Struktur erzeugen" (*Abbildung unten*) geöffnet, in dem Register und Fenster angezeigt werden, über die Sie die gewünschte Datenbankstruktur auswählen können. Diese Optionen sind weiter unten beschrieben. Klicken Sie anschließend auf die Schaltfläche **Importieren**, um die Datenbankdaten zu importieren.



Register "Auswahl"

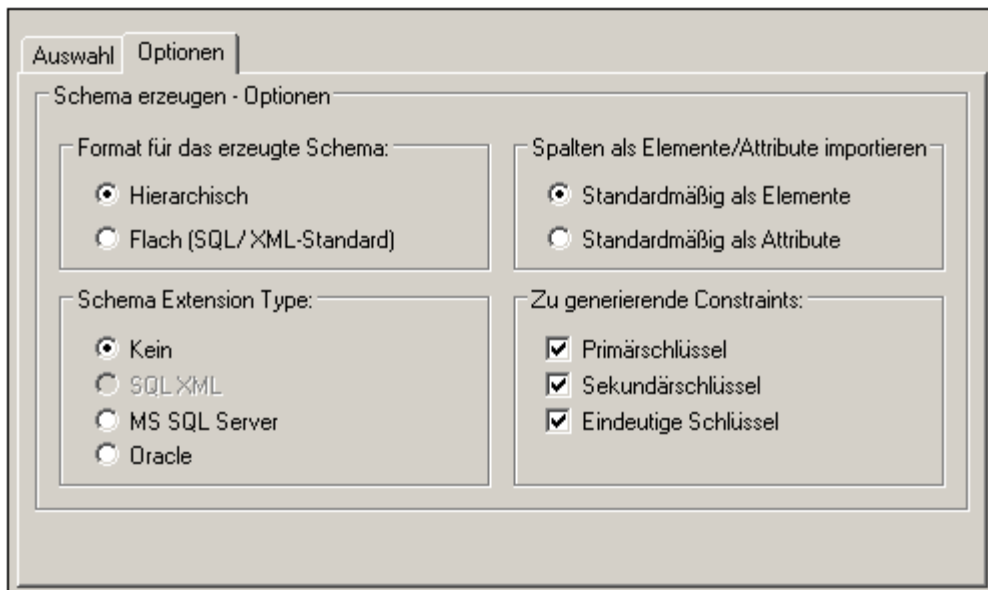
Auf dem Register "Auswahl" (*Abbildung oben*) wird die Datenquelle im Bereich "Datenbankquelle" aufgelistet. Im Bereich "Quelle" (*Abbildung unten*) werden entweder die Tabellen der Datenbank angezeigt oder Sie sehen eine editierbare SQL-Anweisung zur Auswahl der gewünschten Tabellen.



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 angezeigt werden. Die Tabellenauswahl kann im Vorschaubereich noch weiter gefiltert werden (*siehe unten*). Sie können die Anzeige im Bereich "Quelle" wie im Abschnitt [Datenbankdaten importieren](#) beschrieben, konfigurieren.

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 einem ISO-ANSI Working Draft mit dem Titel "XML-Related Specification (SQL/XML)" (im Weiteren als "SQL/XML Working Draft")

bezeichnet). 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. Nähere Informationen zum SQL/XML Working Draft finden Sie unter: <http://www.sqlx.org/SQLXdocs>

- 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.

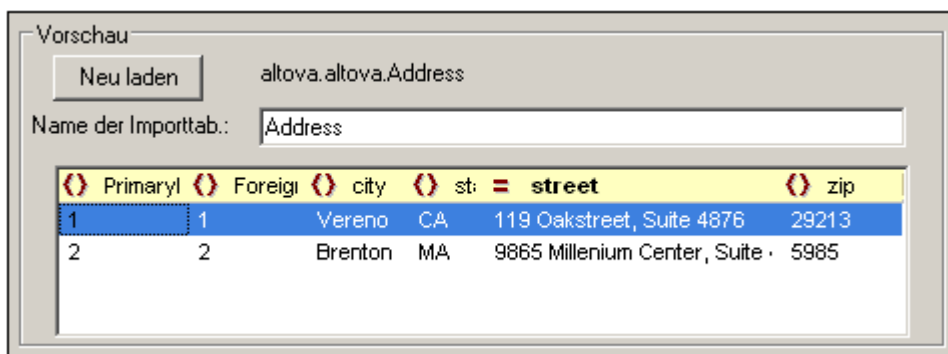
Schema-Extension-Typ: Schema-Extension-Informationen sind zusätzliche Informationen, die aus einer Datenbank ausgelesen werden und die anschließend in Form von Annotationsdaten oder Attributen in das Schema eingebettet werden. Beim Generieren von Schemas gibt es vier Extension-Typ-Optionen: (i) Keine Extension-Informationen; (ii) SQL/XML Extensions; (iii) MS SQL Server Extensions; and (iv) Oracle Extensions. Die folgenden Optionen stehen zur Verfügung:

- **Kein:** 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 im SQL/XML Working Draft beschrieben.
- **MS SQL Server:** Bei Auswahl von Microsoft SQL Server werden SQL Server Extensions generiert. Nähere Informationen dazu finden Sie online in den SQL Server Books unter *Using Annotations in XSL Schemas* (Unter <http://www.microsoft.com/sql/techinfo/productdoc/2000/books.asp>). Im Schema wird die folgende Untergruppe mit Annotationsdaten 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. Eine ausführliche Beschreibung dieser Attribute finden Sie unter http://download-west.oracle.com/docs/cd/B10501_01/appdev.920/a96620/xd05obj.htm#1027227. 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 derzeit auf dem Register "Auswahl" ausgewählten Tabelle angezeigt. Wenn auf dem Register "Auswahl" eine neue Tabelle ausgewählt wird, klicken Sie im Vorschaubereich auf die Schaltfläche "Aktualisieren", 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](#).

18.10.5 DB-Import auf Basis von XML-Schema



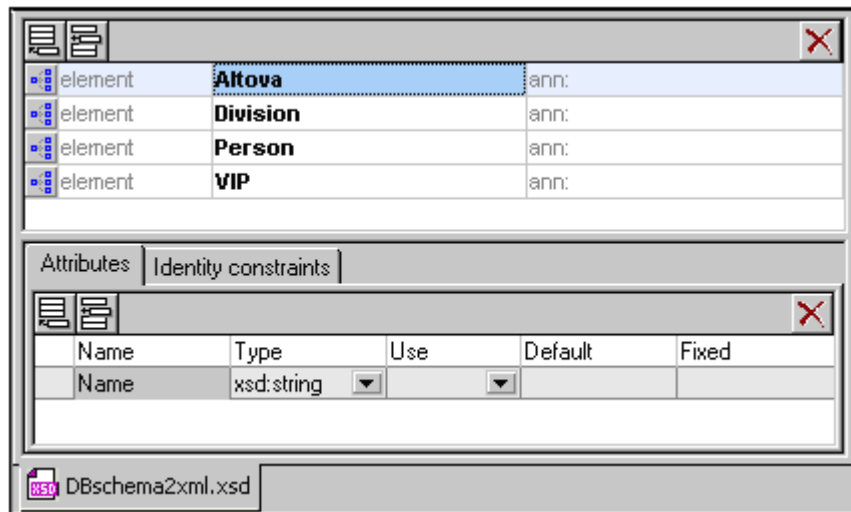
Mit Hilfe des Befehls **DB-Import auf Basis von XML-Schema** wird ein XML-Dokument erstellt, welches gemäß einem bestimmten XML-Schema gültig ist und aus einer Datenbank importierte Daten enthält. Diese Funktion unterstützt die folgenden Datenbanken:

- 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 das Dialogfeld [Verbindung zur Datenquelle herstellen](#) 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](#).

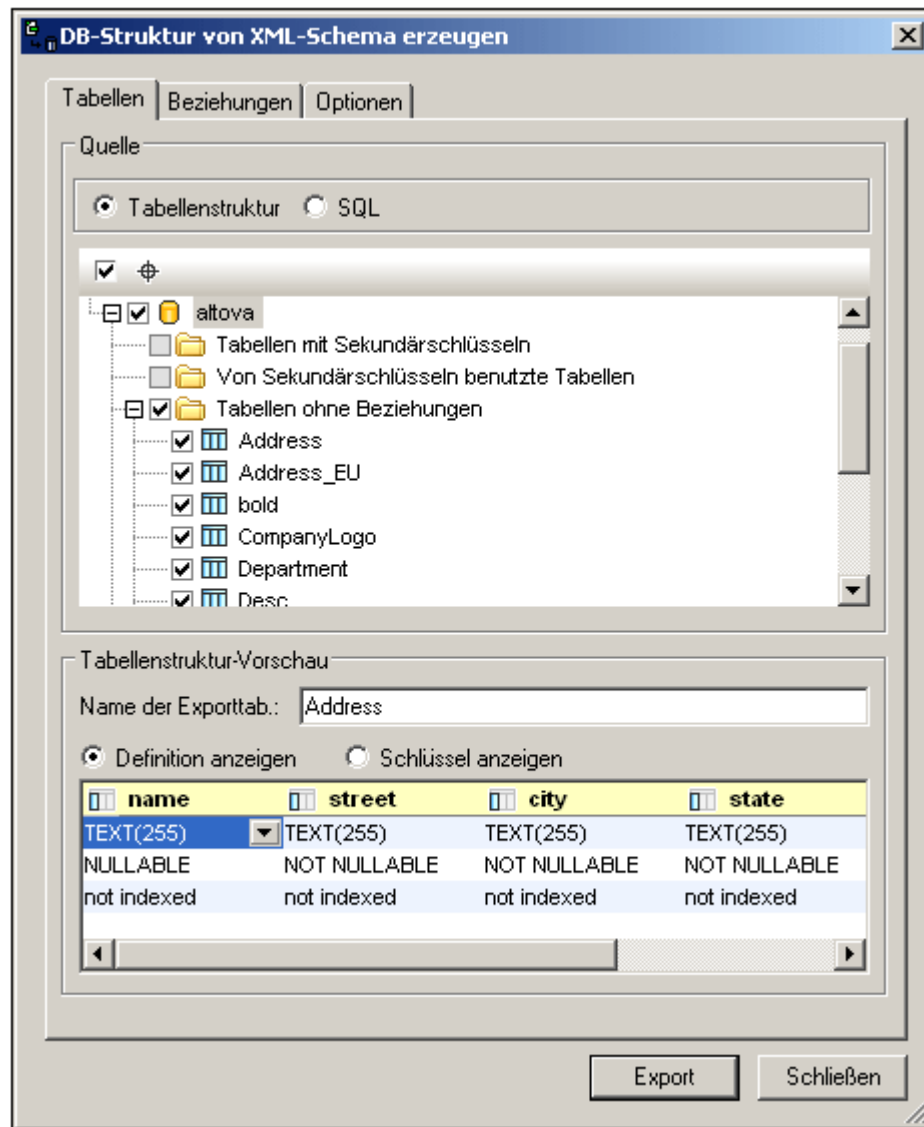
18.10.6 DB-Struktur von XML-Schema erzeugen



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](#), in dem Sie eine Verbindung zu einer Datenbank (DB) herstellen können.
3. Gehen Sie vor, wie unter [Herstellen einer Verbindung zu einer Datenquelle](#) beschrieben, um eine Verbindung zur gewünschten Datenbank herzustellen. Um z.B. eine Verbindung zu einer Microsoft 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, in der die Schemastruktur enthalten sein wird.
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).



	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](#). 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](#).

18.10.7 Export in Textdateien



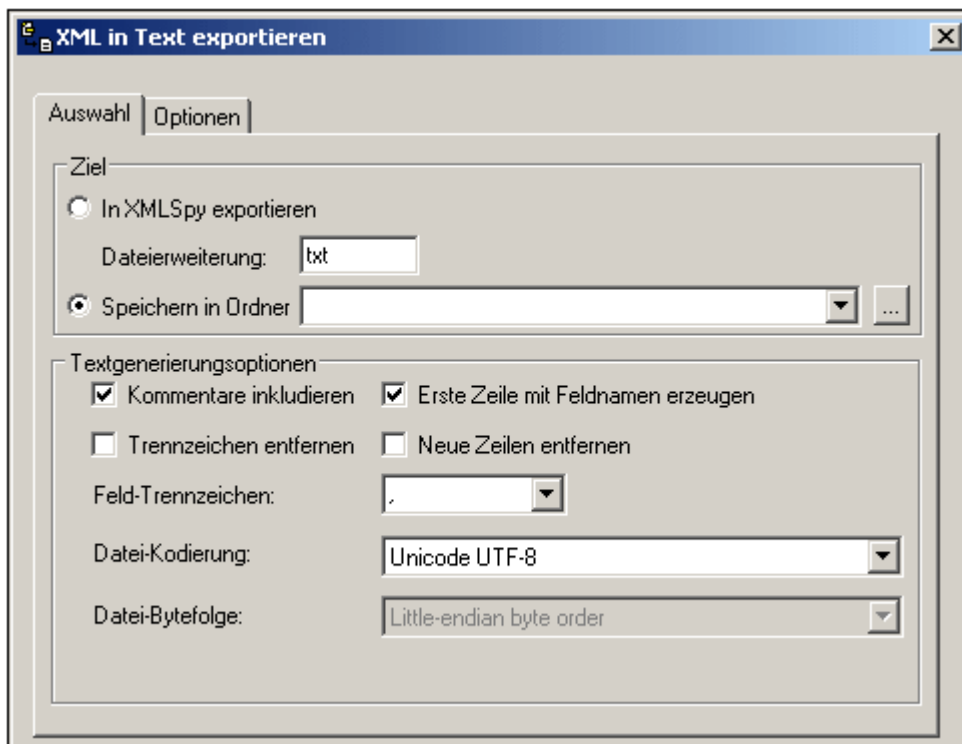
Mit dem Befehl **Konvertieren | Export in Textdateien** werden XML-Daten für den Austausch mit Datenbanken oder Altdatenbeständen in Textformate exportiert. Wenn Sie auf diesen Befehl klicken, wird das Dialogfeld "XML in Text exportieren" aufgerufen. Es besteht aus zwei Teilen (*die in der Abbildung unten separat angezeigt werden*):

- einem oberen Bereich mit zwei Registern: (i) "Selection" und (ii) "Export Options".
- 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 Kommentar in die exportierte XML-Datei zu inkludieren, der die zur Auswahl der Daten verwendete SQL-Abfrage sowie eine Liste mit einem Eintrag für jede Spaltenüberschrift in der Datenbanktabelle anzeigt.

Erste Zeile mit Feldnamen erzeugen: Ist diese Option aktiviert, enthalten die Tabellen die Namen der Datenbankspalten.

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.

Optionen

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.

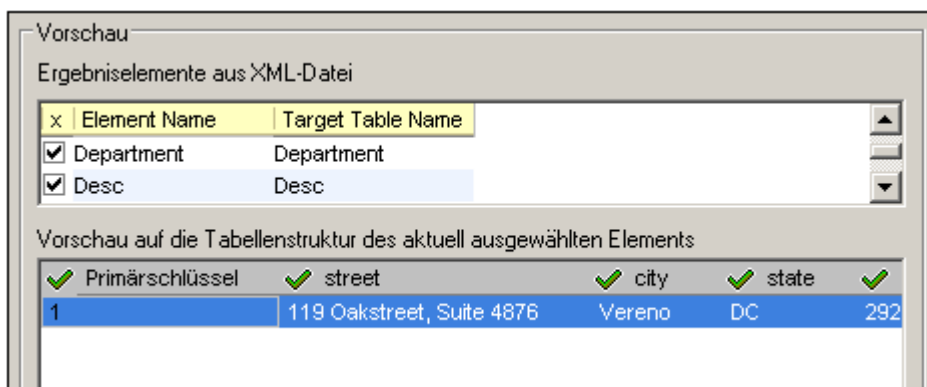
Exportfelder erzeugen aus: Abhängig von Ihren XML-Daten möchten Sie eventuell nur Elemente, Attribute oder den Textinhalt Ihrer Elemente exportieren. Beachten Sie, dass Sie den Export für einzelne Elemente im Vorschauenfenster aufheben können.

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.

Vorschauenfenster

Das Vorschauenfenster (*Abbildung unten*) wird unterhalb der Register "Auswahl" und "Optionen" angezeigt.



Im Bereich "Ergebniselemente aus der XML-Datei" wird der Node-Name angezeigt, der exportiert werden soll sowie der Name in der generierten Datei. 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.

18.10.8 Export in Datenbank



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.

The screenshot shows the 'Optionen' (Options) tab of a dialog box. It is divided into several sections. The 'Export beginnt' section has two radio buttons: 'am Anfang der XML-Datei' (selected) and 'Beginn bei' (with a dropdown menu). The 'Export-Tiefe' section has two radio buttons: 'Alle Sub-Elemente exportieren' (selected) and 'Nur bis:' (with a numeric input field set to 0) and 'Sub-Ebenen'. The 'Exportfelder erzeugen aus:' section has four checkboxes: 'Attributen' (checked), 'Textinhalt' (checked), 'Textwert auf Parent anwenden' (unchecked), and 'Entities in Text konvertieren' (checked). The 'Generiere automatische Felder' section has two checkboxes: 'Primär- /Sekundärschlüssel' (checked) and 'Eigener Primärschlüsselzähler für jedes Element' (checked). The 'Andere Optionen' section has two radio buttons: 'Namespace-Namen ausnehmen' (selected) and 'Unterstrich ersetzt Doppelpunkt' (unchecked). There is an 'Optionen anw.' button at the bottom left.

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.

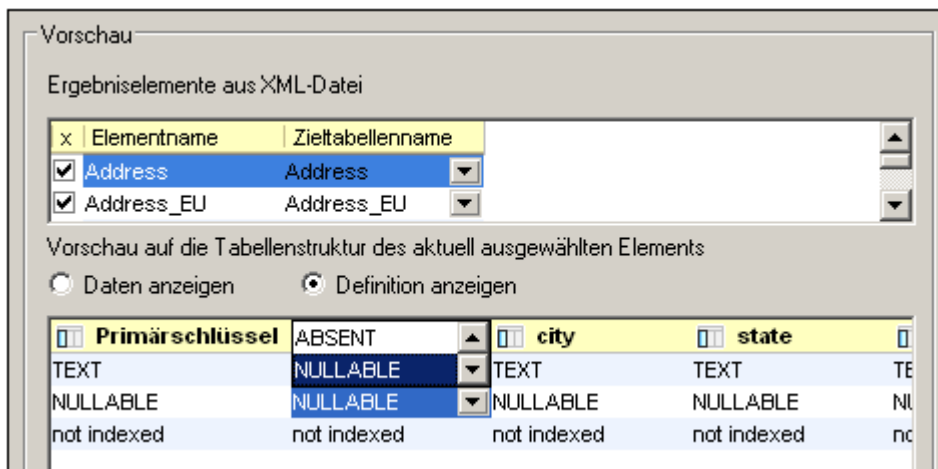
Exportfelder erzeugen aus: Abhängig von Ihren XML-Daten möchten Sie eventuell nur Elemente, Attribute oder den Textinhalt Ihrer Elemente exportieren. Beachten Sie, dass Sie den Export für einzelne Elemente im Vorschaufenster aufheben können.

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.

Vorschaufenster

Das Vorschaufenster (*Abbildung unten*) wird unterhalb der Register "Auswahl" und "Optionen" angezeigt.



Im Bereich "Ergebniselemente aus der XML-Datei" wird der Node-Name angezeigt, der exportiert werden soll sowie der Name in der generierten Datei. 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. In diesem Fenster wird der Export der entsprechenden Spalte bei Klick links vom Spaltennamen aktiviert bzw. deaktiviert. In der Abbildung oben wurde die Spalte `paidnewyear` deaktiviert, sodass sie nicht exportiert wird.

Wenn in der Elementtabelle die Daten angezeigt werden, kann der Key Constraint-Typ durch Klicken links von einem Spaltennamen ausgewählt werden. Wenn in der Struktur der Elementtabelle Felddefinitionen angezeigt werden sollen, können Sie die Definitionen bearbeiten, indem Sie die Definition auswählen und eine Option aus der Auswahlliste der Definition auswählen (siehe Abbildung oben).

18.10.9 XML von/in JSON konvertieren

Mit diesem Befehl wird das aktive Dokument in ein JSON-Dokument konvertiert, falls es sich um ein XML-Dokument handelt. Wenn das aktive Dokument ein JSON-Dokument ist, wird es mit diesem Befehl in ein XML-Dokument konvertiert. Das generierte Dokument kann dann in jedem beliebigen Ordner gespeichert werden.

JSON (JavaScript Object Notation) ist ein Datenaustauschformat, das auf einer Teilmenge von JavaScript basiert und normalerweise mit JavaScript-Code verwendet wird. Code zum Parsen von JSON steht für die meisten Programmiersprachen zur Verfügung.

XMLSpy bietet intelligente Bearbeitungsfunktionen für JSON-Dokumente. Diese Funktionen werden in dieser Dokumentation im Abschnitt [JSON-Bearbeitungsfunktionen](#) näher beschrieben.

Unten sehen Sie ein Beispiel für ein XML-Quelldokument. Unterhalb davon sehen Sie das JSON-Dokument, das mit dem Befehl **XML 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 von/in JSON konvertieren**.

18.11 Menü "Ansicht"

Im Menü Ansicht kontrollieren Sie die Anzeige für das aktive [Hauptfenster](#). So können Sie sich in XMLSpy Ihre XML-Dokumente verschiedenartig anzeigen lassen.

In diesem Abschnitt finden Sie eine vollständige Beschreibung aller Befehle im Menü "Ansicht".

18.11.1 Text-Ansicht



Mit diesem Befehl **wechseln** Sie für das aktuelle Dokument in die [Textansicht](#). In dieser Ansicht können Sie das Dokument in der Textform editieren. Die Text-Ansicht unterstützt eine Reihe von Editierfunktionen, die im Abschnitt [Textansicht](#) näher beschrieben sind.

Bitte beachten Sie: Sie können Aspekte der Textansicht mit Hilfe von Optionen konfigurieren, die auf den verschiedenen Registern des Dialogfelds "Optionen" ([Extras | Optionen](#)) zur Verfügung stehen.

18.11.2 Grid Ansicht



Mit diesem Befehl **wechseln** Sie mit dem aktuellen Dokument in die [Grid-Ansicht](#).

Wenn Sie von der [Textansicht](#) in diese Ansicht wechseln, wird Ihr Dokument vorab auf Wohlgeformtheit überprüft.

ipo:purchaseOrder	
= xmlns:xsi	http://www.w3.org/2001/XMLSchema-instance
= xmlns:ipo	http://www.altova.com/IPO
= orderDate	1999-12-01
= xsi:schema...	http://www.altova.com/IPO ipo.xsd
▼ shipTo export-code=1 xsitype=ipo:EU-Address	
▲ billTo	
= xsi:type	ipo:US-Address
() name	Robert Smith
() street	8 Oak Avenue
() city	Old Town
() state	AK
() zip	95819

Eingebettete Datenbank/Tabelle Ansicht

Mit XMLSpy können Sie sich wiederholende Elemente innerhalb der Enhanced Grid Ansicht in einer [Datenbank/Tabelle Ansicht](#) ansehen. Diese Funktion ist immer dann verfügbar, wenn Sie die Grid Ansicht aktivieren können, und kann beim Editieren jeder beliebigen XML Datei - XML, XSD, XSL usw. verwendet werden.

Weitere Informationen zu diesem Thema finden Sie auch im Abschnitt [Grid-Ansicht](#).

18.11.3 Schema-Ansicht



Mit diesem Befehl **wechseln** Sie mit dem aktuellen Dokument in die Schema-Ansicht. Sie können nur in diese Ansicht wechseln, wenn es sich beim Dokument um ein XML-Schema-Dokument handelt. Nähere Informationen dazu erhalten Sie im Abschnitt [Schema-Ansicht](#).

18.11.4 Authentic-Ansicht



Mit diesem Befehl wechseln Sie mit Ihrem Dokument in die [Authentic](#)-Ansicht.

In der Authentic-Ansicht können Sie XML-Dokumente bearbeiten, die auf in **<%XDESIGN%> erstellten StyleVision Power Stylesheet-Vorlagen basieren!** Die Vorlagen werden in **<%XDESIGN%>** als StyleVision Power Stylesheet (*.sps Dateien) gespeichert, und liefern alle für die Authentic-Ansicht benötigten Informationen.

Öffnen Sie die Vorlagen über den Menübefehl **Datei | Neu** und klicken auf die Schaltfläche "StyleVision Stylesheet auswählen...". Weitere Informationen finden Sie in der Dokumentation zur Authentic-Ansicht.

Bitte beachten Sie: Wenn bei Umschalten in die Authentic-Ansicht eine Meldung angezeigt wird, dass eine temporäre Datei nicht erstellt werden konnte, wenden Sie sich an Ihren Systemadministrator, damit dieser die Standard-Sicherheits-ID für Personen, die nicht als Hauptbenutzer angemeldet sind, ändert, so dass diese Ordner und Dateien erstellen können.

18.11.5 Browser-Ansicht



Mit diesem Befehl wechseln Sie mit dem aktuellen Dokument in die [Browser Ansicht](#).

Diese Ansicht verwendet einen XML-fähigen Browser um das XML-Dokument mit den aus CSS oder XSL Style-Sheets zur Verfügung gestellten Informationen darzustellen.

Wenn Sie in die Browser Ansicht wechseln, wird das Dokument zuerst auf Gültigkeit geprüft, wenn Sie die Option "Validieren beim Speichern der Datei" unter **Extras | Optionen** auf dem Register **Datei** einschalten.

Weitere Informationen zu dieser Ansicht finden Sie im Abschnitt [Hauptfenster](#).

18.11.6 Erweitern



Kürzel: **"+" auf dem Ziffernblock**

Mit diesem Befehl blenden Sie eine weitere Ebene des ausgewählten Elements ein.

Der Befehl kann in der Grid Ansicht ausgeführt werden.

In der Grid Ansicht, bleiben das Element und seine Child-Elemente nach dem Einblenden markiert. So können Sie große Elemente einblenden indem Sie wiederholt die + Taste betätigen.

Sie können Elemente auch durch Klicken auf die graue Leiste links eines jeden Elements einblenden oder ausblenden.

18.11.7 Reduzieren



Kürzel: "-" auf dem Ziffernblock

Mit diesem Befehl **blenden** Sie ein Element um eine Ebene **aus**.

Der Befehl kann in der Grid Ansicht ausgeführt werden.

Sie können Elemente auch durch Klicken auf die graue Leiste links eines jeden Elements einblenden oder ausblenden.

18.11.8 Alles erweitern



Kürzel: "*" auf dem Ziffernblock

Mit diesem Befehl **blenden** Sie alle Child-Objekte des **ausgewählten Elements** ein.

Der Befehl kann in der Grid Ansicht ausgeführt werden.

18.11.9 Unselektierte reduzieren

Kürzel: **CTRL + "-"** auf dem Ziffernblock

Mit diesem Befehl können Sie sich auf ein Element und dessen Children konzentrieren und alle anderen Elemente ignorieren.

Der Befehl kann in der Grid Ansicht ausgeführt werden.

Wählen Sie das Element aus, mit dem Sie arbeiten möchten und blenden Sie mit diesem Befehl alle anderen nicht selektierten Elemente aus.

18.11.10 Optimale Breite



Mit diesem Befehl stellen Sie die Größe der Spalten so ein, dass jede Spalte so breit ist, dass der längste Text einer ihrer Zellen in einer Zeile darin Platz hat. Im Dialogfeld "Optionen" (im Menü "Extras") können Sie auf dem Register "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.

18.11.11 Zeilenumbruch



Mit diesem Befehl schalten Sie den Zeilenumbruch in der **Text-Ansicht** ein oder aus.

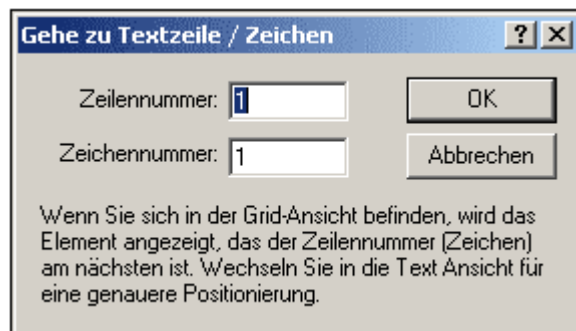
18.11.12 Gehe zu Zeile/Zeichen



Kürzel: **Strg+g**

Mit diesem Befehl gelangen Sie zu einer **spezifischen Zeilennummer** und/oder Zeichenposition in einem XML-Dokument in der Text-Ansicht.

Wenn Sie mit einem externen XSLT Prozessor (weitere Infos: [Register XSL unter Extras | Optionen](#)) arbeiten, werden Sie öfter Fehlermeldungen bezüglich Zeilennummer und/oder Zeichenposition erhalten. Mit XMLSpy können Sie mit diesem Befehl exakt zur gewünschten Position navigieren.



Diese Funktion ist in der [Text-Ansicht](#) und Grid Ansicht verfügbar, zeigt allerdings in der Grid-Ansicht nur die ungefähre Position an, indem das Element markiert wird, welches sich am nächsten zur gewünschten Position befindet.

18.11.13 Gehe zu Datei



Mit diesem Befehl **öffnen** Sie das Dokument, welches auf das von Ihnen momentan bearbeitete Dokument **referenziert**.

Wählen Sie den gewünschten Dateinamen, Pfadnamen, oder die URL und führen Sie diesen Befehl über das [Menü Ansicht](#) aus.

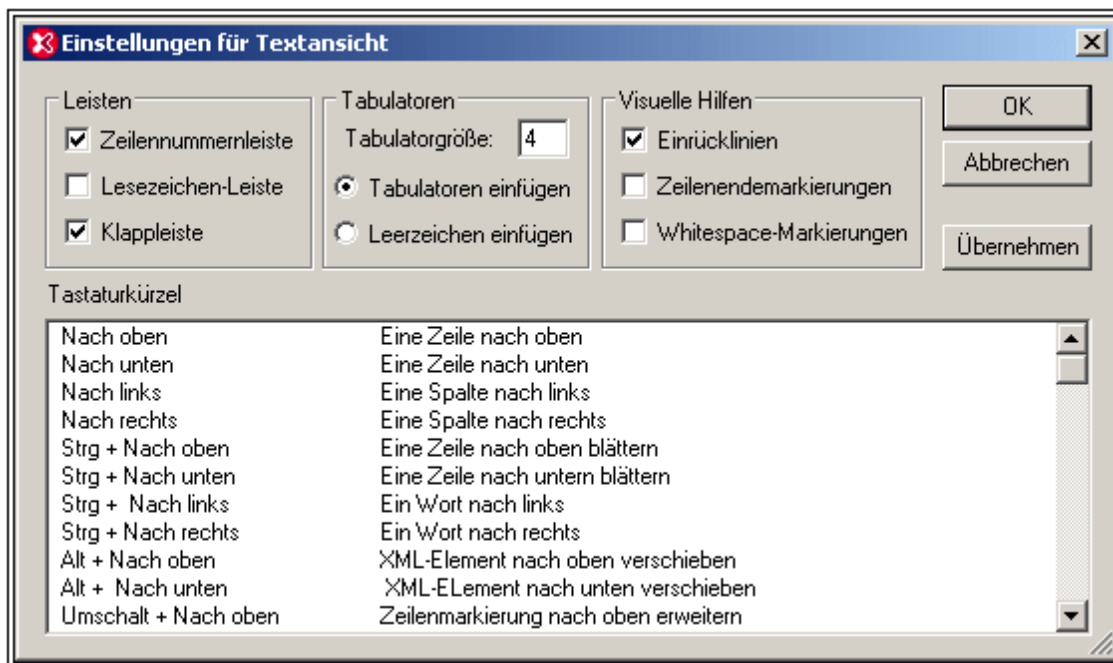
Sie können folgendes auswählen:

- Ein vollständiges Element oder Attribut in der [Grid-Ansicht](#)
- Zeichen innerhalb eines beliebigen Elements in der [Text](#) oder [Grid](#)-Ansicht.
- Einen kompletten String. Wenn sich der Cursor zwischen zwei Anführungszeichen befindet, benutzt XMLSpy automatisch den kompletten String der sich zwischen den Anführungszeichen befindet.

18.11.14 Einstellungen für Textansicht



Mit dem Befehl **Einstellungen für Textansicht** 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.



Leisten

Im Bereich "Leisten" können Sie die Zeilennummerleiste, die Lesezeichenleiste und die Klappleiste ein- und ausblenden. Dabei handelt es sich um separate Leisten, in denen Zeilennummern, Lesezeichen und 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 oder nicht. Die Lesezeichenbefehle befinden sich im Menü **Bearbeiten**. Um Nodes ein- und auszuklappen, muss die Klappleiste eingeblendet sein.

Tabulatoren

Im Bereich "Tabulatoren" können Sie die Tabulatorgröße in Form der Anzahl von Leerzeichen definiert. Mit Hilfe der Optionsfelder zum Einfügen von entweder Tabulatoren oder Leerzeichen, legen Sie fest, ob Dokumente mit Tabulator- oder Leerzeicheneinrückungen angezeigt werden, wenn die [Pretty-Print mit Einrückung](#) 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), Zeilenendezeichen sowie Whitespace-Markierungen.

Tastaturkürzel

Die Liste der Tastaturkürzel enthält die Tastaturkürzel und die damit verknüpften Befehle.

18.12 Menü "Browser"

Dieses Menü enthält Befehle, die nur verfügbar sind, wenn die [Browser Ansicht](#) aktiviert ist.

18.12.1 Zurück



Taste **Zurück (Alt + Pfeil nach links)**

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. Das Tastaturkürzel dafür ist **Alt + Pfeil nach links**. Mit dem Befehl **Zurück** können Sie bis zu 500 zuvor angezeigte Positionen erneut anzeigen.

18.12.2 Vorwärts



(Alt + Pfeil nach rechts)

Dieser Befehl ist nur dann verfügbar, wenn Sie zuvor den Befehl **Zurück** verwendet haben. Mit diesem Befehl gelangen sie (i) in der Browser-Ansicht vorwärts zur vorher angezeigten Seite (ii) und in der Schema-Ansicht zu zuvor angezeigten Komponenten.

18.12.3 Abbrechen



Mit diesem Befehl beenden Sie das Laden eines neuen Dokuments. Dieser Befehl ist nützlich, wenn Sie große externe Dateien oder graphische Darstellungen über eine langsame Internetverbindung laden.

18.12.4 Aktualisieren



F5

Mit diesem Befehl aktualisieren Sie die Browser Ansicht durch erneutes Laden des XML-Dokuments, sowie potentieller externer CSS oder XSL Stylesheets und DTDs.

18.12.5 Schriftgröße

Mit diesem Befehl können Sie die Schriftgröße wählen, die für Text in Ihrem XML-Dokument verwendet werden soll. Die Funktionsweise ist ähnlich wie die, welche in den meisten Browsern zur Anwendung kommt.

18.12.6 Neues Fenster



Mit diesem Befehl öffnen Sie die Browser Ansicht in einem neuen Fenster, sodass Sie

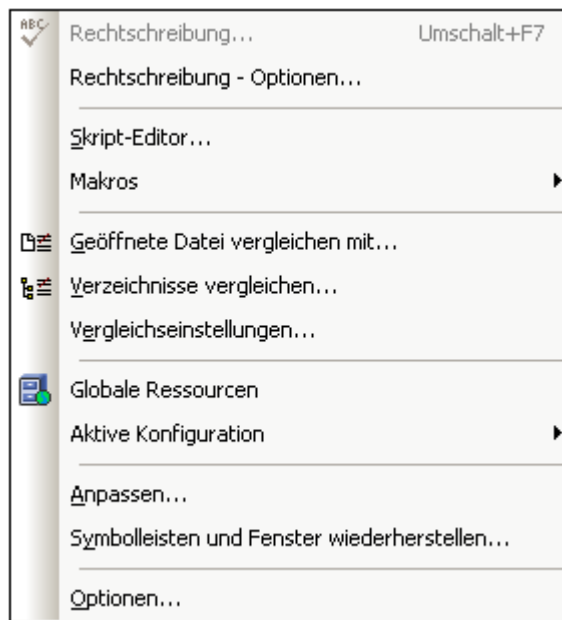
Dokumente nebeneinander ansehen können.

Wenn Sie die Browser Ansicht in einem neuen Fenster geöffnet haben, drücken Sie in der Bearbeitungsansicht die Taste **F5**, um die Browser-Ansicht automatisch zu aktualisieren. Um separate Fenster wieder anzudocken, klicken Sie auf die Maximieren-Schaltfläche in der rechten oberen Ecke des aktiven Fensters.

18.13 Menü "Extras"

Im Menü Extras können Sie:

- Die [Rechtschreibung](#) Ihrer XML-Dokumente überprüfen
- In die [Skripting-Umgebung](#) von XMLSpy wechseln. Sie können Ihre eigenen Formulare, Makros und Event Handler erstellen, bearbeiten und speichern.
- Derzeit zugewiesene Makros [ansehen](#)
- zwei beliebige Dateien vergleichen und auf Unterschiede überprüfen
- zwei beliebige Ordner vergleichen und auf Unterschiede überprüfen
- [globale Ressourcen definieren](#)
- [die aktive Konfiguration für globale Ressourcen](#) in XMLSpy wechseln
- Ihre Version von XMLSpy [anpassen](#): eigene Menüleisten, Tastaturkürzel, Menüs und Makros erstellen
- Globale XMLSpy-[Einstellungen](#) definieren



18.13.1 Rechtschreibung

Die Rechtschreibprüfung von XMLSpy mit integrierten Wörterbüchern (*siehe Anmerkung unten*) kann in folgenden Ansichten verwendet werden: Textansicht Grid-Ansicht und Authentic-Ansicht. 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 Auswahl der integrierten Wörterbücher, die mit Altova Software verfügbar sind, ist nicht Ausdruck einer Präferenz für bestimmte Sprachen durch Altova, sondern ist größtenteils auf der Verfügbarkeit der Wörterbücher begründet, 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 Lizenz einschränkungen wie z.B. im Rahmen von [GPL](#) Lizenzen weitergegeben. Viele dieser Wörterbücher stehen als Teil eines separaten Installationsprogramms unter <http://www.altova.com/de/dictionaries> zur Verfügung. 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.

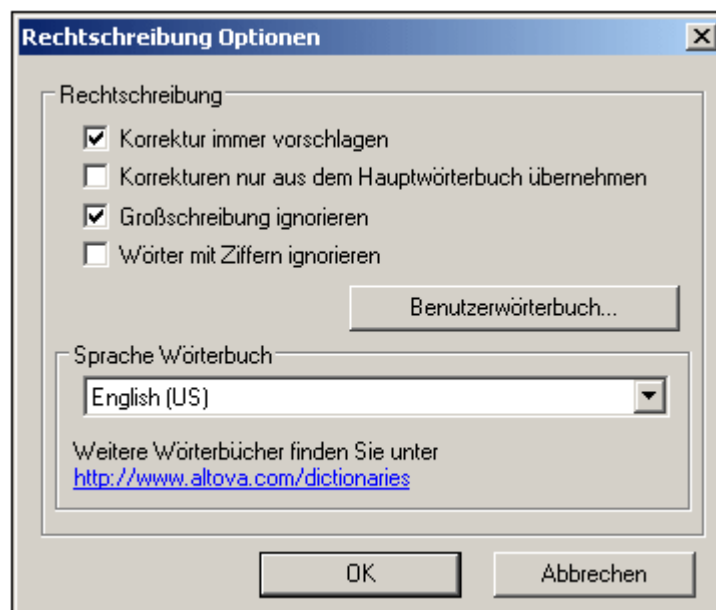
In diesem Abschnitt finden Sie eine Anleitung zur Verwendung der Rechtschreibprüfung. Der Abschnitt ist in die folgenden Unterabschnitte gegliedert:

- [Auswahl der Sprache für die Rechtschreibprüfung](#)
- [Definieren des Rahmens der Rechtschreibprüfung](#)
- [Ausführen der Rechtschreibprüfung](#)

Auswahl der Sprache für die Rechtschreibprüfung

Die Sprache für die Rechtschreibprüfung wird folgendermaßen festgelegt:

1. Wählen Sie den Menübefehl **Extras | Rechtschreibung - Optionen**.
2. Klicken Sie im Dialogfeld "Rechtschreibkontext", das daraufhin angezeigt wird, auf die Schaltfläche **Rechtschreiboptionen**.
3. Wählen Sie im Dialogfeld "Rechtschreibung Optionen" (*Abbildung unten*) eines der installierten Wörterbücher aus der Dropdown-Liste des Auswahlfelds "Sprache Wörterbuch) aus.



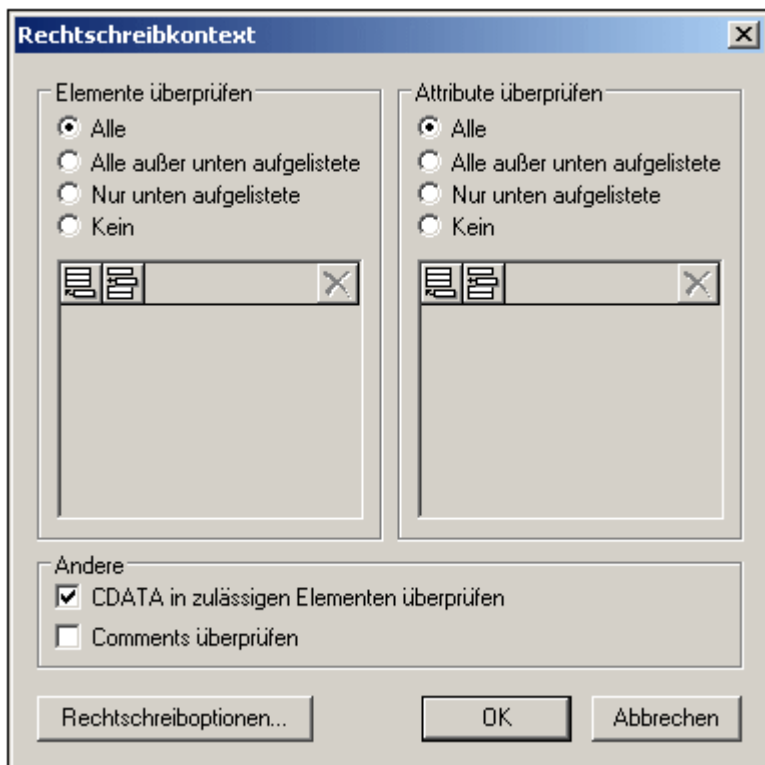
4. Klicken Sie zum Fertigstellen auf **OK**.

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](#).

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 | Rechtschreibung - Optionen** und definieren Sie den gewünschten Rahmen im nun angezeigten Dialogfeld [Rechtschreibkontext](#) (

siehe Abbildung unten).



Sie können auswählen, welche Elemente und Attribute überprüft werden sollen und ob auch CDATA-Abschnitte und Kommentare überprüft werden sollen. Informationen dazu finden Sie auch in der Beschreibung zum Befehl [Rechtschreiboptionen](#).

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](#) 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.



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 den Text 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 Listefeld zeigt eine Wörter an, die Ähnlichkeiten mit dem unbekannten Wort haben (die Wörter stammen aus dem Wörterbuch und dem Benutzerwörterbuch). 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 [Rechtschreibung Optionen](#) 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

Das Dokument wird vom Anfang an neu überprüft.

Optionen

Mit diesem Befehl öffnen Sie ein Dialogfeld in der aktuellen Ansicht:

- Bei Aufruf über die Authentic-Ansicht wird das Dialogfeld [Rechtschreibung - Optionen](#) geöffnet.
- In der Text oder Grid-Ansicht wird das Dialogfeld [Rechtschreibkontext](#) geöffnet.

Nähere Informationen dazu finden Sie auch in der Beschreibung zum Befehl [Rechtschreiboptionen](#).

Schließen

Mit diesem Befehl schließen Sie das Dialogfeld "Rechtschreibung".

18.13.2 Rechtschreibung - Optionen

Mit diesem Befehl öffnen Sie das Dialogfeld [Rechtschreibung - Optionen](#).

Mit diesem Befehl öffnen Sie, je nachdem welche Ansicht aktiv ist, entweder direkt das Dialogfeld [Rechtschreibung - Optionen](#) (Schema-Ansicht, WSDL-Ansicht, XBRL-Ansicht, Authentic-Ansicht, Browser-Ansicht) oder das Dialogfeld [Rechtschreibkontext](#) (Textansicht, Grid-Ansicht, usw.). Das Dialogfeld **Rechtschreibkontext** hat eine Schaltfläche "Rechtschreiboptionen" um das Dialogfeld "Rechtschreibung Optionen" aufzurufen.

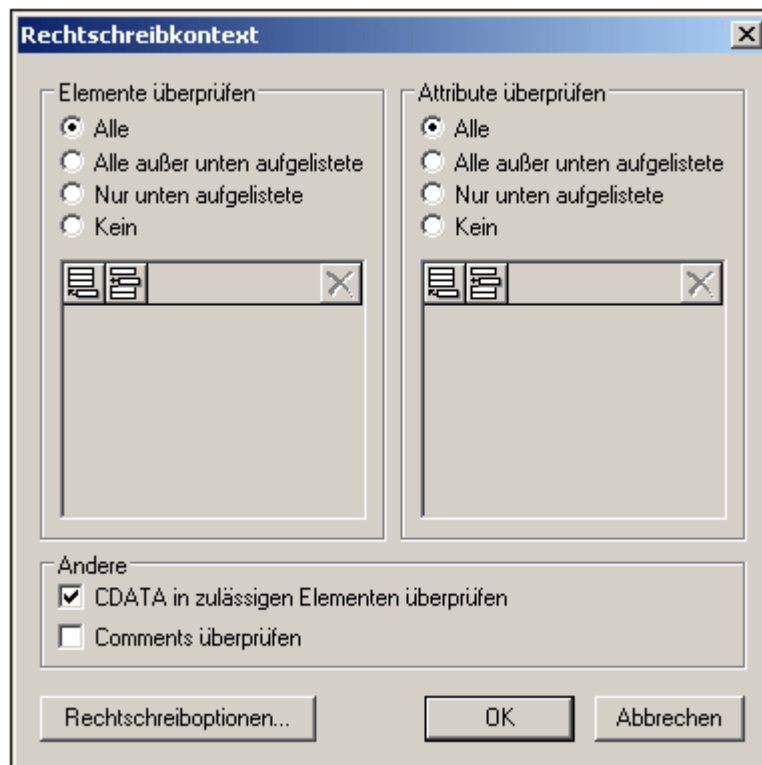
Die verschiedenen Einstellungen, die in diesen beiden Dialogfeldern zur Verfügung stehen, werden in den Unterabschnitten dieses Abschnitts beschrieben:

- [Rechtschreibkontext](#)
- [Rechtschreibung Optionen](#)
- [Hinzufügen von Wörterbüchern für die Rechtschreibprüfung](#)
- [Arbeiten mit dem Benutzerwörterbuch](#)

Dialogfeld "Rechtschreibkontext"

Das Dialogfeld "Rechtschreibkontext" (*Abbildung unten*) wird geöffnet, wenn Sie in der Text- oder der Grid-Ansicht den Befehl **Rechtschreibung Optionen** aufrufen. Im Dialogfeld "Rechtschreibkontext" können Sie wählen, an welchen Bereichen die Rechtschreibprüfung durchgeführt werden soll. 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.

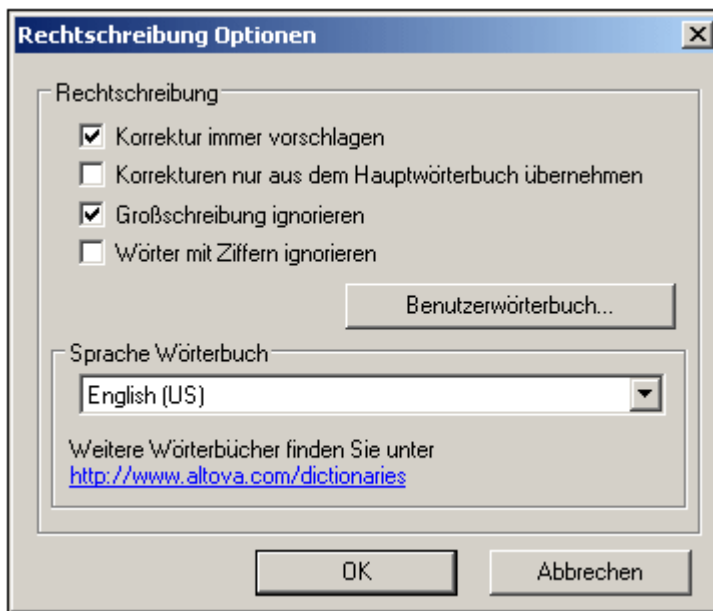


Sie können eine Liste von Elementen und/oder Attributen zusammenstellen, deren Rechtschreibung überprüft werden soll bzw. die bei der Rechtschreibprüfung nicht berücksichtigt werden sollen. Alternativ dazu können Sie die Rechtschreibprüfung an allen Elementen und/oder Attributen oder an keinem Element oder Attribut durchführen.

Wenn Sie am unteren Rand des Dialogfelds auf die Schaltfläche **Rechtschreiboptionen** klicken, wird das Dialogfeld "Rechtschreibung Optionen" geöffnet.

Rechtschreibung Optionen

Über das Dialogfeld **Rechtschreibung Optionen** 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 keine 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.

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](http://www.altova.com/dictionaries) 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 im Ordner `Lexicons` installiert:

Auf Windows 7: C:\ProgramData\Altova\SpellChecker\Lexicons

Auf Windows XP: C:\Documents and Settings\All Users\Application Data\Altova\SharedBetweenVersions\SpellChecker\Lexicons

Im Ordner `Lexicons` werden unterschiedliche Sprachwörterbücher in jeweils anderen Ordnern gespeichert: `<Sprachename>\<Wörterbuchdateien>`. So werden z.B. Dateien für die beiden englischsprachigen Wörterbücher (`English (British)` und `English (US)`) auf einem

Windows 7-Rechner folgendermaßen gespeichert:

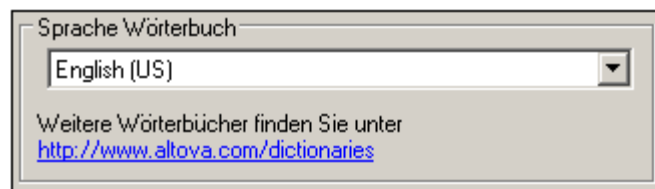
```
C:\ProgramData\Altova\SpellChecker\Lexicons\English (British)\en_GB.aff  
C:\ProgramData\Altova\SpellChecker\Lexicons\English (British)\en_GB.dic  
C:\ProgramData\Altova\SpellChecker\Lexicons\English (US)\en_US.dic  
C:\ProgramData\Altova\SpellChecker\Lexicons\English (US)\en_US.dic
```

Die Sprachwörterbücher werden im Dialogfeld "Rechtschreibung Optionen" 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:

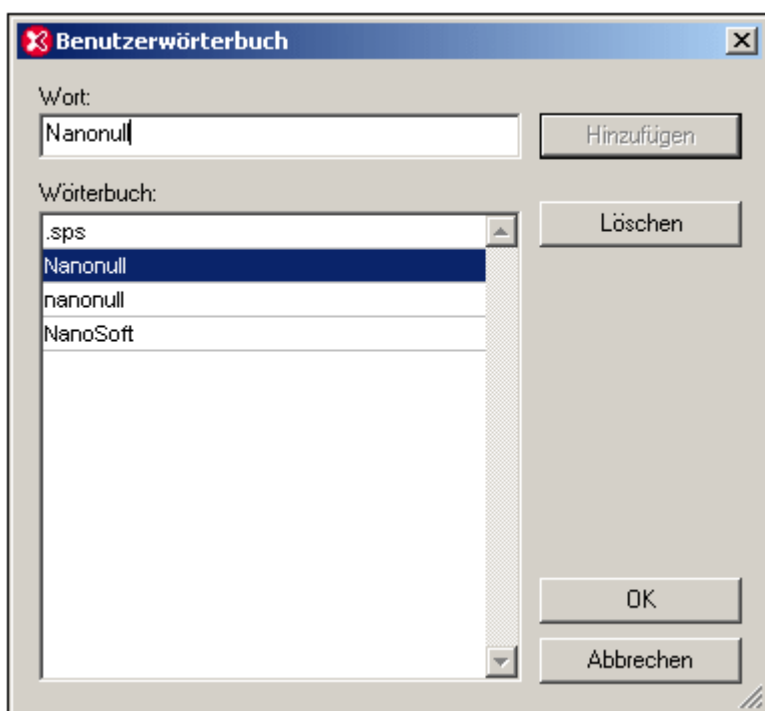
- Durch Hinzufügen von Hunspell-Wörterbüchern in einem neuen Unterordner des Ordners `Lexicons`. Hunspell-Wörterbücher können z.B. von <http://wiki.services.openoffice.org/wiki/Dictionaryes> oder <http://extensions.services.openoffice.org/en/dictionaries> heruntergeladen werden. (Beachten Sie bitte, 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.



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 "Rechtschreibung Optionen" aufgerufen (siehe *Abbildung im obersten Bereich dieses Abschnitts*).



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 [Rechtschreibung](#) 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:

Auf Windows 7: C:\Users\<user>\Documents\Altova\SpellChecker\Lexicons\user.dic

Auf Windows XP: C:\Documents and Settings\<user>\My Documents\Altova\SpellChecker\Lexicons\user.dic

18.13.3 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 [Skripterstellung](#).

Anmerkung: Zur Ausführung des Skript-Editors muss die .NET Framework Version 2.0 oder höher auf Ihrem Rechner installiert sein.

18.13.4 Makros

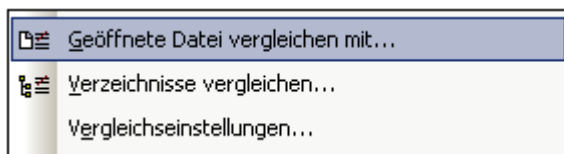
Wenn Sie die Maus über den Befehl **Makros** positionieren, wird ein Untermenü mit den Makros angezeigt, die im Skripting-Projekt, das in XMLSpy gerade aktiv ist, definiert sind (*Abbildung unten*).



Durch Klicken auf ein Makro im Untermenü (*Siehe Abbildung oben*) wird das Makro ausgeführt.

18.13.5 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](#)
- [Verzeichnisse vergleichen](#)
- [Vergleichseinstellungen](#)

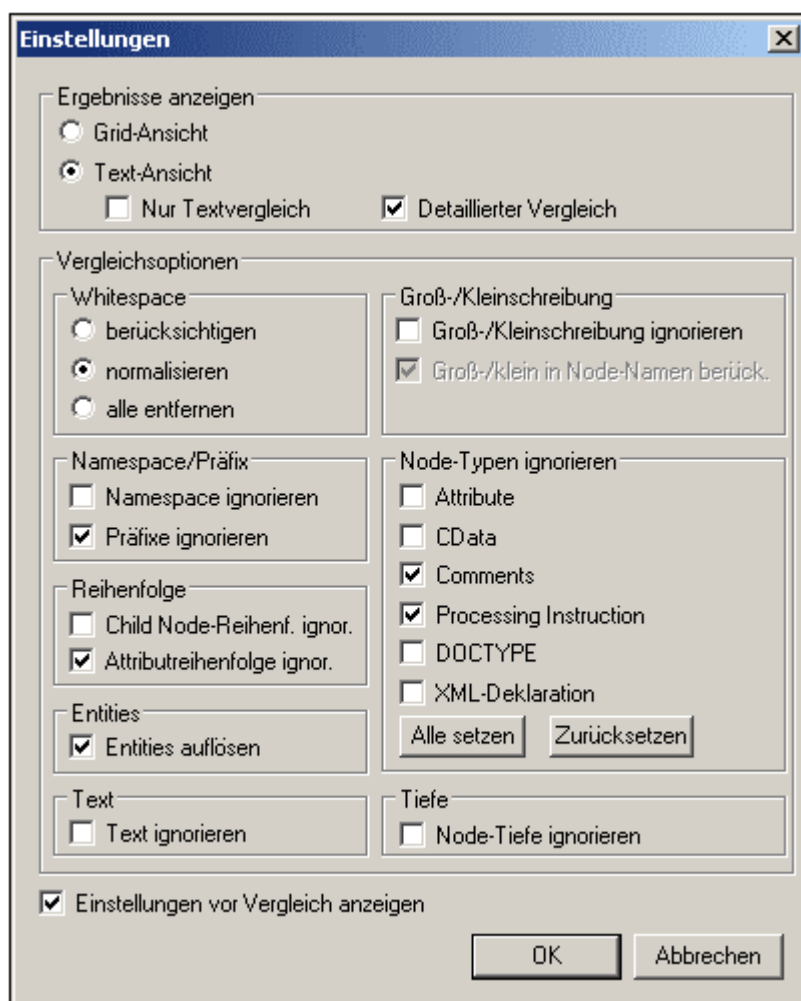
Diese Befehle werden in den folgenden Unterabschnitten näher beschrieben.

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 grün markiert sind. Dateien können als XML-Dokumente (unter Berücksichtigung der Struktur und Semantik der Tags) oder als Textdokumente verglichen werden.

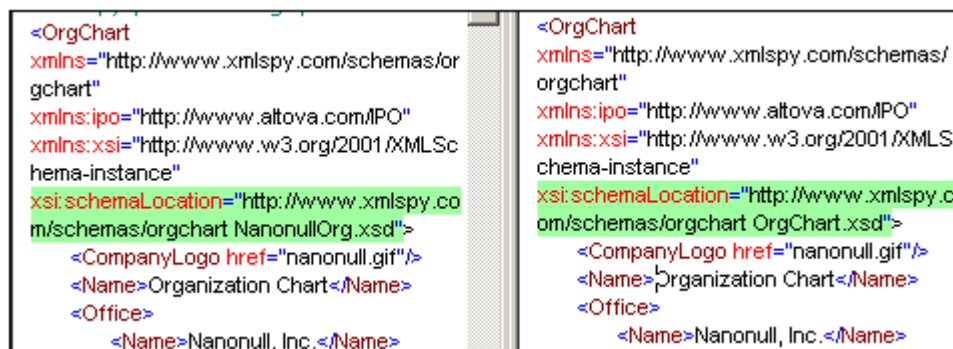
So vergleichen Sie eine geöffnete Datei mit einer anderen Datei:

1. Wenn die Datei im Hauptfenster geöffnet ist (im Hauptfenster kann immer nur jeweils eine Datei aktiv sein), klicken Sie auf **Extras | Geöffnete Datei vergleichen mit....**
2. Suchen Sie die Datei, mit der Sie die geöffnete Datei vergleichen möchten, und wählen Sie sie aus. Sie können eine Datei auch über eine globale Ressource, eine URL (klicken Sie dazu auf die Schaltfläche [Durchsuchen](#)) oder eines der offenen Fenster in XMLSpy auswählen (Klicken Sie dazu auf die Schaltfläche **Fenster**).
3. Bestätigen Sie mit OK. Die zweite Datei wird geöffnet und das Dialogfeld "Einstellungen" wird angezeigt:



Diese Einstellungen werden unter [Vergleichseinstellungen](#) beschrieben. Wenn Sie nicht möchten, dass dieses Dialogfeld jedes Mal angezeigt wird, wenn Sie einen Dateivergleich starten, deaktivieren Sie das Kontrollkästchen "Einstellungen vor Vergleich anzeigen" am unteren Rand des Dialogfelds.

4. Wählen Sie die gewünschten Einstellungen aus und klicken Sie auf OK. Die beiden Dateien werden nun im Hauptfenster angezeigt. Zeilen, die Unterschiede aufweisen, erscheinen grün markiert. Der jeweils markierte Unterschied wird in beiden Dateien in einem dunkleren Grün angezeigt. (siehe auch den Abschnitt "Markierungsfarben" unten!)



Außerdem wird das Steuerungsfenster "Dateien vergleichen" angezeigt:

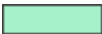
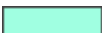


Um den nächsten Unterschied auszuwählen, klicken Sie auf die Schaltfläche **"Weiter"**. Mit Hilfe der Schaltflächen **Erste**, **Letzte** und **Vorherig** können Sie von einem Unterschied zu anderen navigieren und einen Unterschied markieren. Im Bereich "Unterschiede zusammenführen" können Sie die markierten unterschiedlichen Stellen von einer Datei in die andere kopieren. Um das Zusammenführen von Dateien zu aktivieren, müssen die folgenden Vergleichsoptionen eingestellt sein: Die Option "Detaillierter Vergleich" muss aktiviert sein und die Option "Node-Tiefe ignorieren" muss deaktiviert sein. Wenn Sie einen Zusammenführvorgang rückgängig machen wollen, stoppen Sie den Vergleich, wählen Sie die Datei, in der die Änderung rückgängig gemacht werden soll, und klicken Sie auf den Befehl **Bearbeiten | Rückgängig** oder drücken Sie **Strg + Z**.

5. Nach Abschluss der Überprüfung und/oder des Zusammenführens von Unterschieden, klicken Sie auf "Fertig".

Markierungsfarben

Die Unterschiede und die zusammengeführten Unterschiede in den beiden Dateien werden mit den folgenden Farben markiert.:

	<i>Unterschied</i>
	<i>Aktueller Unterschied</i>
	<i>Zusammengeführter Unterschied</i>
	<i>Aktueller zusammengeführter Unterschied</i>

Bitte beachten Sie:

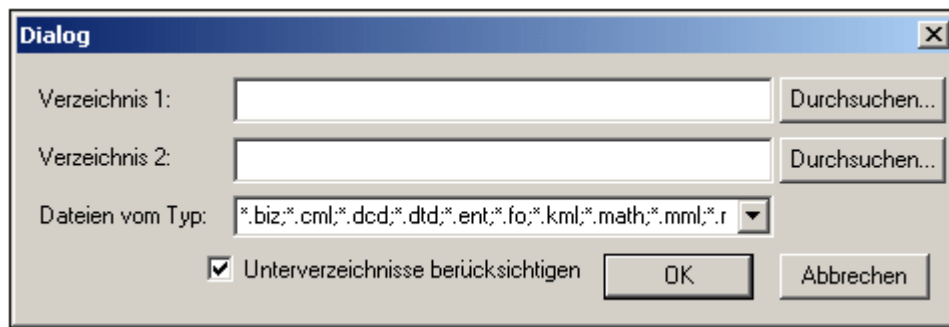
- Während des Vergleichsvorgangs können keine Bearbeitungen vorgenommen werden und auch die Ansicht kann nicht gewechselt werden. Bei Versuchen, die Ansicht einer der Dateien zu wechseln oder eine Datei zu bearbeiten, wird eine Meldung angezeigt mit der Warnung, dass dadurch der Vergleich beendet wird.
- Sie können während eines Vergleichs die Vergleichseinstellungen ändern (durch Auswahl von **Extras | Vergleichseinstellungen...**), doch werden die Änderungen erst beim nächsten Vergleichsvorgang wirksam; die neuen Einstellungen haben keinen Einfluss auf die aktuelle Sitzung.

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.

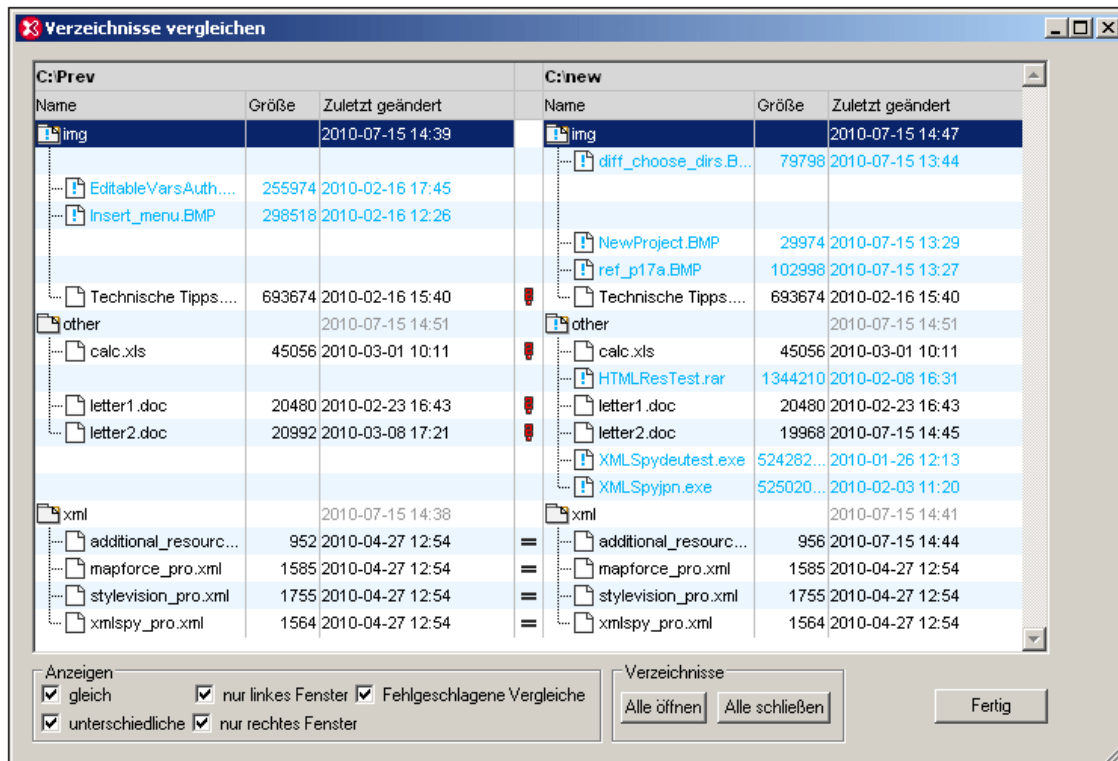
So vergleichen Sie zwei Verzeichnisse:

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](#); und (iii) alle Dateitypen.
4. Klicken Sie auf OK. Das Dialogfeld "Einstellungen" wird angezeigt (Beschreibung siehe [Vergleichseinstellungen](#)).
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:



Verzeichnissymbole

Alle Verzeichnisnamen werden schwarz angezeigt.



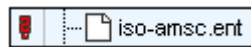
Das Verzeichnis ist reduziert und sein Inhalt wird nicht angezeigt.

-  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.

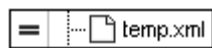
Bitte beachten Sie: Wenn ein Verzeichnis sowohl eine Datei enthält, die im verglichenen Verzeichnis nicht existiert, als auch eine Datei, die sich von der gleichnamigen Datei im verglichenen Verzeichnis unterscheidet, wird für das Verzeichnis das Symbol  angezeigt.

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 jünger 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.

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](#)").

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.

The screenshot shows a Windows-style dialog box titled "Einstellungen" (Settings). It contains several sections of options:

- Ergebnisse anzeigen** (Show results):
 - ☐ Grid-Ansicht
 - ☒ Text-Ansicht
 - ☐ Nur Textvergleich
 - ☒ Detaillierter Vergleich
- Vergleichsoptionen** (Comparison options):
 - Whitespace**:
 - ☐ berücksichtigen
 - ☒ normalisieren
 - ☐ alle entfernen
 - Groß-/Kleinschreibung** (Case):
 - ☐ Groß-/Kleinschreibung ignorieren
 - ☒ Groß-/klein in Node-Namen berück.
 - Namespace/Präfix**:
 - ☐ Namespace ignorieren
 - ☒ Präfixe ignorieren
 - Node-Typen ignorieren** (Ignore node types):
 - ☐ Attribute
 - ☐ CDATA
 - ☒ Comments
 - ☒ Processing Instruction
 - ☐ DOCTYPE
 - ☐ XML-Deklaration
 - Reihenfolge** (Order):
 - ☐ Child Node-Reihenf. ignor.
 - ☒ Attributreihenfolge ignor.
 - Entities**:
 - ☒ Entities auflösen
 - Text**:
 - ☐ Text ignorieren
 - Tiefe** (Depth):
 - ☐ Node-Tiefe ignorieren
- Buttons: "Alle setzen" (Set all) and "Zurücksetzen" (Reset).
- ☒ Einstellungen vor Vergleich anzeigen (Show settings before comparison).
- Buttons: "OK" and "Abbrechen" (Cancel).

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.

Bitte beachten Sie: 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.

Bitte beachten Sie: Um die Zusammenführung von Dateien zu aktivieren, muss die Option "Detaillierter Vergleich" aktiviert sein.

Whitespaces

Bei Leerzeichen handelt es sich um Leerzeichen, Tabulatorzeichen, Wagenrücklauf und Zeilenvorschub. Die Optionen in diesem Bereich dienen zum Vergleichen von Dateien, bei denen die Leerzeichen identisch sind, zum Vergleich von Dateien bei denen die Leerzeichen vereinheitlicht wurden (d.h. alle aufeinander folgenden Leerzeichen werden als ein Leerzeichen betrachtet) und zum Vergleich von Dateien, bei denen alle Leerzeichen 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.

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. Beachten Sie jedoch, dass jeder Child-Element-Node an seinem Namen, seinen Attributen und seiner Position identifiziert wird. 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

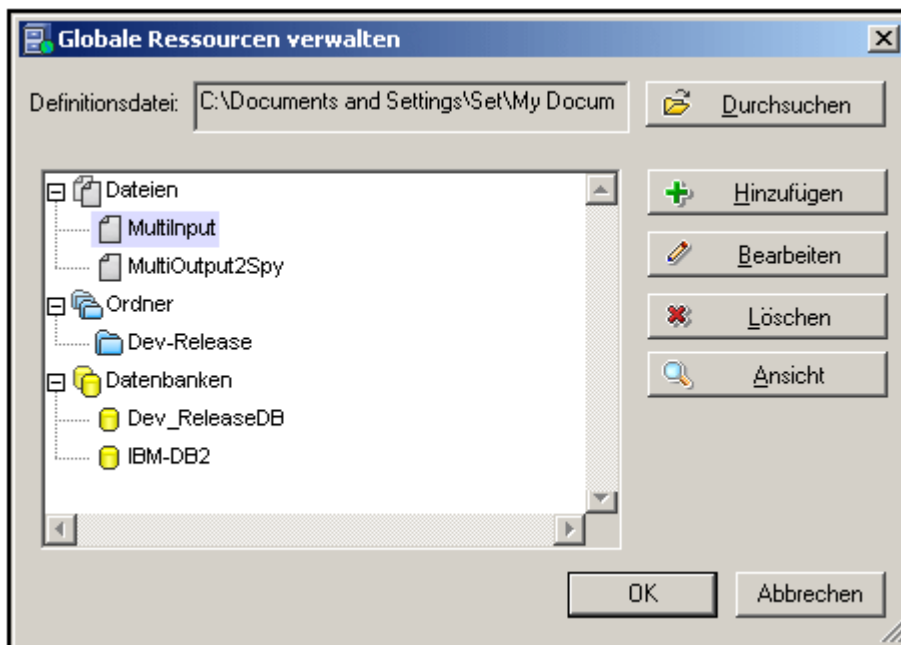
Wenn Sie diese Option aktivieren, wird das Dialogfeld "Einstellungen" jedes Mal angezeigt, wenn Sie auf den Befehl **"Geöffnete Datei vergleichen mit"** klicken. Auf diese Art können Sie die Einstellungen vor jedem Vergleich überprüfen und ändern.

Ist dieses Kontrollkästchen deaktiviert, wird der Vergleich direkt gestartet, wenn Sie den entsprechenden Befehl wählen.

18.13.6 Globale Ressourcen

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.

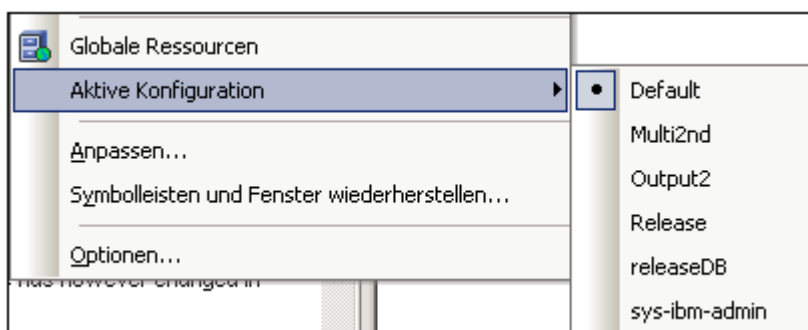


Eine Beschreibung, wie man globale Ressourcen definiert, finden Sie im Abschnitt [Definieren globaler Ressourcen](#).

Anmerkung: Das Dialogfeld "Globale Ressourcen" kann auch über die [Symbolleiste "Globale Ressourcen"](#) aufgerufen werden (**Ansicht | Anpassen | Symbolleisten | Globale Ressourcen**)

18.13.7 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](#) 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.

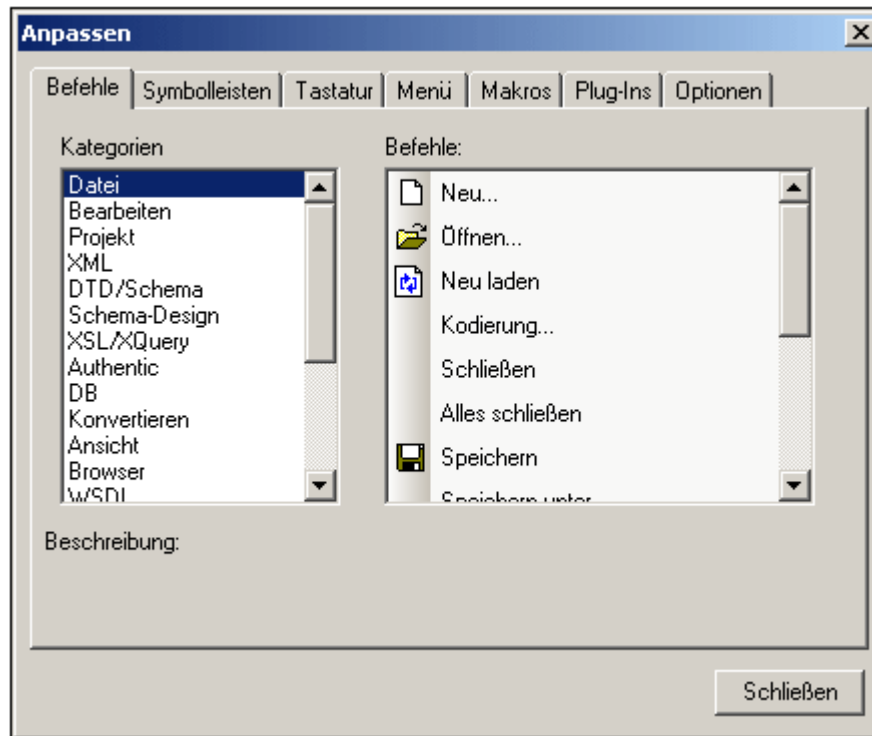
Anmerkung: Die aktive Konfiguration kann auch über die [Symbolleiste "Globale Ressourcen"](#) ausgewählt werden (**Ansicht | Anpassen | Symbolleisten | Globale Ressourcen**).

18.13.8 Anpassen

Mit diesem Befehl können Sie XMLSpy an Ihre eigenen Bedürfnisse anpassen.

Befehle

Auf dem Register Befehle können Sie Ihre Menüs oder Symbolleisten anpassen.



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-Listefeld. Die verfügbaren Befehle werden im entsprechenden Listefeld angezeigt.
 3. Markieren Sie einen Befehl aus dem Listefeld, 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.
- An der Spitze des Mauszeigers wird beim Ziehen eines Befehls eine kleine Schaltfläche angezeigt. Das Zeichen unterhalb des Zeigers zeigt an, dass der Befehl nicht an diese Position gestellt werden kann.
 - Das Zeichen verschwindet, sobald Sie eine Position erreichen, an die der Befehl gestellt werden kann (innerhalb eines Menüs oder einer Symbolleiste).
 - Wenn Sie den Mauszeiger über ein Menü halten, wird dieses geöffnet, so dass Sie den Befehl an einer beliebigen Stelle im Menü einfügen können.
 - Befehle können in Menüs oder Symbolleisten eingefügt werden. Wenn Sie eine eigene Symbolleiste erzeugt haben, können Sie nun die gewünschten Befehle einfügen.

Bitte beachten Sie:

Sie können mit dieser Methode auch die Befehle der **Kontextmenüs** bearbeiten (rechtsklicken Sie auf eine beliebige Stelle im Fenster, und das Kontextmenü wird geöffnet). Klicken Sie auf das Register Menü und wählen Sie aus dem Dropdown-Listefeld das gewünschte Kontextmenü aus.

So löschen Sie einen Befehl oder ein Menü:

1. Öffnen Sie dieses Dialogfeld über die Menübefehle **Extras | Anpassen**.
2. Klicken Sie auf den Menüeintrag oder das Symbol, welches Sie löschen möchten, und ziehen Sie es mit der Maus an eine Stelle, an der die kleine Schaltfläche am Zeiger angezeigt wird.
3. Lassen Sie die Maustaste los, und der Befehl wird entfernt.

Symbolleisten

Auf dem Register Symbolleisten können Sie spezifische Symbolleisten aktivieren oder deaktivieren. Weiters können Sie eigene Symbolleisten erstellen.

XMLSpy Symbolleisten enthalten Symbole für die meistverwendeten Befehle. Zu jedem Symbol gehört ein kurzer Tooltip, der angezeigt wird, wenn Sie mit dem Cursor auf das Symbol gehen. In der Statusleiste wird eine detailliertere Beschreibung zum jeweiligen Symbol angezeigt.

Sie können die Symbolleisten aus Ihrer Standardposition an eine andere beliebige Stelle am Bildschirm verschieben, sowie diese an den rechten oder linken Rand des Hauptfensters andocken.

- Symbolleisteneinstellungen, die in der Grid-, Schema-, und Text-Ansicht definiert wurden, stehen in den jeweiligen Ansichten zur Verfügung. Die Symbolleisten der Browser-Ansicht sind unabhängig von der jeweiligen Ansicht immer verfügbar.



So aktivieren oder deaktivieren Sie eine Symbolleiste:

1. Klicken Sie auf das Kontrollkästchen um die jeweilige Symbolleiste zu aktivieren oder deaktivieren.

Änderungen in allen Ansichten anwenden

- Aktivieren Sie dieses Kontrollkästchen am unteren Rand des Dialogfelds, um die Änderungen auf alle Ansichten (und nicht nur auf die aktuelle Ansicht) anzuwenden. Wenn Sie auf **Schließen** klicken, werden alle Änderungen, die **nach** Aktivierung des Kontrollkästchens vorgenommen wurden, auf alle Ansichten angewendet.

So erstellen Sie eine neue Symbolleiste:

1. Klicken Sie auf die Schaltfläche **Neu...** und geben Sie der neuen Symbolleiste einen Namen (Dialogfeld Symbolleistenname).
2. Fügen Sie Befehle in die neue Symbolleiste ein (Register [Befehle](#) im Dialogfeld **Anpassen**).

So setzen Sie die Menüleiste zurück

- Klicken Sie auf den Menüleisteneintrag und
- Klicken Sie auf die Schaltfläche **Zurücksetzen**, um die Menübefehle auf Werkseinstellung zurückzusetzen.

So setzen Sie alle Symbolleisten und Menübefehle zurück

- Klicken Sie auf die Schaltfläche **Alles Zurücksetzen**, um die Befehle der Symbolleisten auf Werkseinstellung zurückzusetzen. Sie erhalten eine Meldung, dass nun alle Symbolleisten und Menüs zurückgesetzt werden.
- Klicken Sie auf Ja um das Zurücksetzen zu bestätigen.

So ändern Sie den Namen einer Symbolleiste:

- Klicken Sie auf die Schaltfläche **Umbenennen...** um den Namen der Symbolleiste zu ändern.

So löschen Sie eine Symbolleiste:

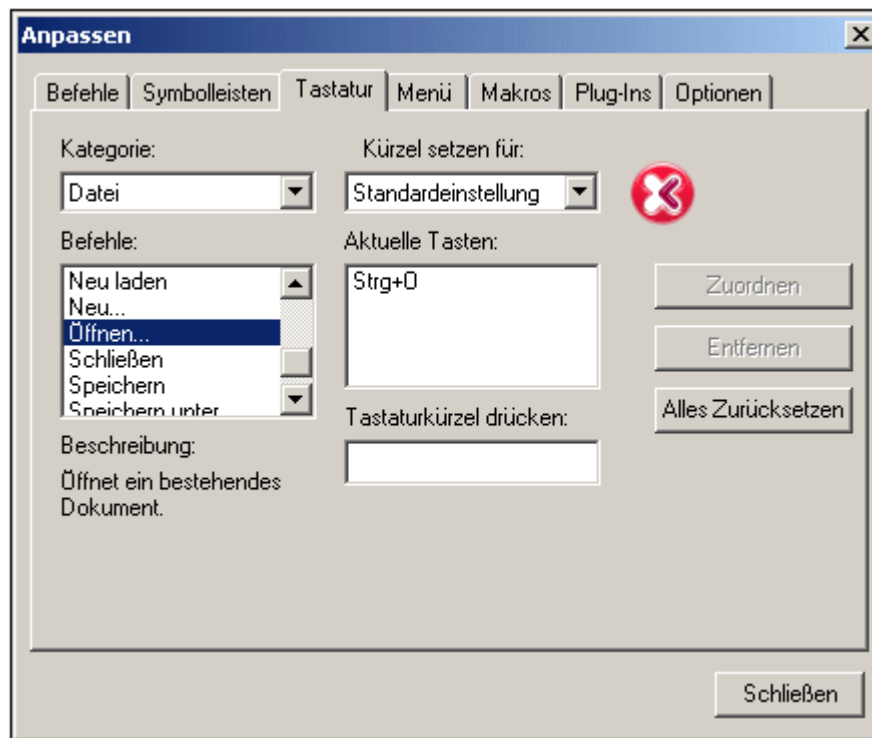
- Klicken Sie auf die Schaltfläche **Löschen** um die im Listenfeld markierte Symbolleiste zu löschen.
- Sie erhalten eine Meldung, in der Sie gefragt werden, ob die markierte Symbolleiste tatsächlich gelöscht werden soll. Bestätigen Sie den Löschvorgang mit Ja.
- oder, rechtsklicken Sie auf die Menü- oder Symbolleiste und wählen Sie Löschen aus dem Popup-Menü.

Textlabels zeigen:

Wenn Sie diese Option aktivieren, werden Hilfetexte zu den Symbolen angezeigt.

Tastatur

Auf dem Register Tastatur können Sie Tastaturkürzel für jeden Befehl in XMLSpy definieren (oder löschen).



So weisen Sie einem Befehl ein neues Kürzel zu:

1. Wählen Sie die Befehlskategorie in der Auswahlliste **Kategorie**. Beachten Sie: Wenn ein Makro als verknüpfter Befehl ausgewählt wurde, können über die Auswahlliste auch Makros ausgewählt werden und es kann für das Makro ein Tastenkürzel definiert werden.
2. Wählen Sie den **Befehl** aus dem Listefeld, welchem Sie ein neues Kürzel zuweisen möchten.
3. Klicken Sie in das "Tastaturkürzel eingeben:" Textfeld und geben Sie das Kürzel ein. Die Kürzel werden sofort im Textfeld angezeigt. Wenn das Kürzel schon einem anderen Befehl zugeordnet war, wird diese Funktion im Textfeld angezeigt.
4. Klicken Sie auf die Schaltfläche **Zuweisen**.
5. Das neue Kürzel wird in die Liste der aktuellen Kürzel aufgenommen.
(um das Textfeld zu **löschen** drücken Sie die Strg-, ALT- oder Umschalttaste).

So heben Sie eine Zuweisung auf (oder löschen ein Kürzel):

1. Klicken Sie auf den Kürzel in der Liste der aktuellen Kürzel, der gelöscht werden soll und
2. Klicken Sie auf die Schaltfläche **Entfernen**, die jetzt aktiv ist.
3. Klicken Sie auf die Schaltfläche Schließen um die Änderungen zu speichern.

Kürzel setzen für:

Derzeit noch keine Funktion.

Zugewiesene Tastatur-Kürzel:**Kürzel nach Tasten**

F1	Hilfe-Menü
F3	Weitersuchen
F5	Aktualisieren
F7	Wohlgeformtheit prüfen
F8	Validieren
F9	Breakpoint einfügen/löschen
Umschalt+F9	Tracepoint einfügen/löschen
Strg+F9	Breakpoint aktivieren/deaktivieren
Umschalt+Strg+F9	Tracepoint aktivieren / deaktivieren
F10	XSL Transformation
Strg+F10	XSL:FO Transformation
F11	Einsteigen
Strg+F11	Überspringen
Umschalt + F11	Aussteigen
Alt+F11	Debugger starten
Num +	Erweitern
Num -	Reduzieren
Num *	Alles erweitern
Strg+Num-	Unselektierte reduzieren
Strg + G	Gehe zu Zeichen/Zeile
Strg+TAB	Wechselt zwischen offenen Dokumenten
Strg+F6	Offene Fenster durchblättern
Pfeiltasten oben/unten	Laufleiste auf/ab
Esc.	Dialogfeld schließen/Editieren abbrechen
Eingabetaste/Leerzeichentaste	Bestätigen
Alt + F4	Schließt XMLSpy
Strg + F4	Schließt aktives Fenster
Alt + F, 1	Öffnet letzte Datei
Strg + Doppelklick Element (Schema-Ansicht)	Elementdefinition anzeigen
Strg + N	Datei Neu
Strg + O	Datei Öffnen
Strg + S	Datei Speichern
Strg + P	Datei Drucken
Strg + A	Alles auswählen
Umschalt + Del	Ausschneiden (oder Strg + X)
Strg + C	Kopieren
Strg + V	Einfügen
Strg + Z	Rückgängig
Strg + Y	Wiederherstellen
Del	Löschen (Objekt in der Schema-Ansicht/Grid Ansicht)

Strg + F	Suchen
F3	Weitersuchen
Strg + H	Ersetzen
Strg + I	Attribut anhängen
Strg + E	In der Grid-Ansicht: Element anhängen. In der Textansicht: Gehe zum Star/End-Tag, wenn sich der Cursor auf dem Gegenstück des Tags befindet
Strg + T	Text anhängen
Strg + D	CDATA anhängen
Strg + M	Comment anhängen
Strg + Umschalt + I	Attribut einfügen
Strg + Umschalt + E	Element einfügen
Strg + Umschalt + T	Text content einfügen
Strg + Umschalt + D	CDATA einfügen
Strg + Umschalt + M	Comment einfügen
Strg + ALT + I	Child Attribute hinzu
Strg + ALT + E	Child Element hinzu
Strg + ALT + T	Child Text hinzu
Strg + ALT + D	Child CDATA hinzu
Strg + ALT + M	Child Comment hinzu
Tastaturkürzel für die Textansicht	
Strg + "+"	Vergrößern
Strg + "-"	Verkleinern
Strg + 0	Zoom zurücksetzen
Strg + Mausrad nach vorne	Vergrößern
Strg + Mausrad nach hinten	Verkleinern

Kürzel nach Funktion

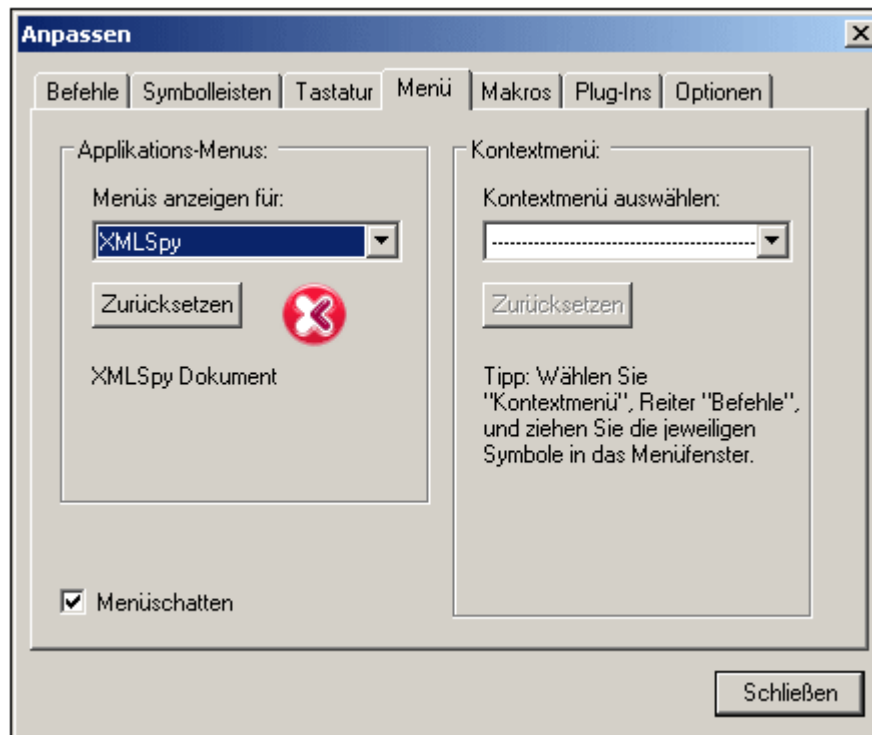
Editieren abbrechen	Esc.
Child Attribute hinzu	Strg + ALT + I
Child CDATA hinzu	Strg + ALT + D
Child Comment hinzu	Strg + ALT + M
Child Element hinzu	Strg + ALT + E
Child Text hinzu	Strg + ALT + T
Attribut anhängen	Strg + I
CDATA anhängen	Strg + D
Comment anhängen	Strg + M
Element anhängen	Strg + E (Grid-Ansicht)
Text anhängen	Strg + T
Wohlgeformtheit prüfen	F7
Aktives Fenster schließen	Strg + F4
Schließt XMLSpy	Alt + F4
Reduzieren	Num -
Unselektierte reduzieren	Strg + Num-
Bestätigen	Return / Space bar
Kopieren	Strg + C
Ausschneiden	Umschalt + Del (or Strg + X)
Offenen Fenster durchblättern	Strg + TAB und Strg + F6
Löschen	Del
Breakpoint	Strg + F9
aktivieren/deaktivieren	
Tracepoint	Umschalt + Strg + F9
aktivieren/deaktivieren	
Erweitern	Num +
Alles erweitern	Num *
Datei Neu	Strg + N
Datei Öffnen	Strg + O
Datei Drucken	Strg + P
Datei Speichern	Strg + S
Suchen	Strg + F
Weitersuchen	F3
Gehe zu Zeichen/Zeile	Strg + G
Hilfemenü	F1
Den anderen Tag des	Strg + E (Textansicht)
Tag-Paars markieren, wenn	
sich der Cursor innerhalb eines	
Element-Start- oder End-Tags	
befindet	
Attribut einfügen	Strg + Umschalt + I
CDATA einfügen	Strg + Umschalt + D
Comment einfügen	Strg + Umschalt + M
Element einfügen	Strg + Umschalt + E
Breakpoint einfügen/löschen	F9
Tracepoint einfügen/löschen	Umschalt+F9
Text content einfügen	Strg + Umschalt + T
Laufleiste auf/ab	Pfeiltasten oben/unten
Letzte Datei öffnen	Alt + F, 1
Einfügen	Strg + V
Wiederherstellen	Strg + Y
Aktualisieren	F5
Ersetzen	Strg + H
Alles markieren	Strg + A

Start Debugger	Alt + F11
Einsteigen	F11
Aussteigen	Umschalt + F11
Überspringen	Strg + F11
Definition anzeigen	Strg + Doppelklick Element
Rückgängig	Strg + Z
Validieren	F8
XSL Transformation	F10
XSL:FO Transformation	Strg + F10

In der Applikation sehen Sie im Dialogfeld "Tastaturhilfe" ([Hilfe | Tastaturzuordnungen](#)) eine Liste der Befehle mit Ihren Tastaturkürzeln und Beschreibungen.

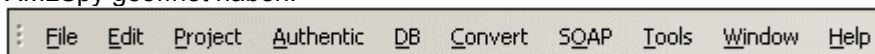
Menü

Auf dem Register Menü können Sie die Menüleisten und die Rechtsklick-Kontextmenüs anpassen.



Sie können sowohl die standardmäßigen Menüleisten, als auch die XMLSpy Menüleisten anpassen.

Das **Standardmenü** (Abbildung unten) wird angezeigt, wenn Sie kein XML-Dokument in XMLSpy geöffnet haben.



Das XMLSpy-Menü ist die Menüleiste, die dargestellt wird, wenn mindestens ein XML-Dokument geöffnet ist.

So passen Sie ein Menü an:

1. Wählen Sie die Menüleiste aus dem Dropdown-Listefeld "Menüs anzeigen für:", welche Sie anpassen wollen.

2. Klicken Sie auf das Register **Befehle** und ziehen Sie die gewünschten Befehle in das ausgewählte Menü.

So löschen Sie Befehle aus einem Menü:

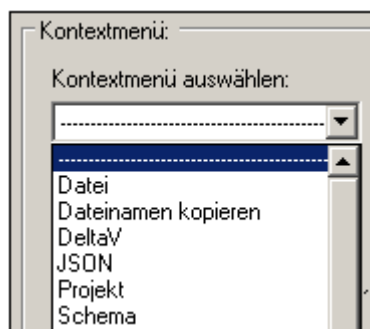
1. Rechtsklicken Sie bei geöffnetem Dialogfeld "Anpassen" auf den Befehl, oder das Symbol, welches für den Befehl steht, und
2. Wählen Sie die Option Löschen aus dem Popup Menü,
oder
1. Wählen Sie **Extras | Anpassen** um das Dialogfeld Anpassen zu öffnen und
2. Ziehen Sie den Befehl aus dem Menü bis am Mauszeiger die kleine Schaltfläche angezeigt wird.

So setzen Sie eine der Menüleisten zurück:

1. Wählen Sie entweder den Eintrag Standard oder XMLSpy aus dem Dropdown-Listefeld und
2. Klicken Sie auf die Schaltfläche **Zurücksetzen** unter dem Menünamen.
Mit einer Meldung werden Sie gefragt, ob Sie die Menüleiste tatsächlich zurücksetzen möchten.

So passen Sie ein Kontextmenü an (Rechtsklick-Menü):

1. Wählen Sie das entsprechende Kontextmenü aus dem Dropdown-Listefeld.
2. Klicken Sie auf das Register **Befehle** und ziehen Sie die Befehle in das offene Kontextmenü.

**So löschen Sie Befehle aus einem Kontextmenü:**

1. Rechtsklicken Sie bei geöffnetem Dialogfeld "Anpassen" auf den Befehl, oder das Symbol, welches für den Befehl steht, und
2. Wählen Sie die Option Löschen aus dem Popup Menü,

oder
1. Wählen Sie **Extras | Anpassen** um das Dialogfeld Anpassen zu öffnen und
2. Ziehen Sie den Befehl aus dem Menü heraus und lassen Sie die Maustaste los, sobald am Mauszeiger die kleine Schaltfläche angezeigt wird.

So setzen Sie eines der Kontextmenüs zurück:

1. Wählen Sie das Kontextmenü aus dem Dropdown-Listefeld und
2. Klicken Sie auf die Schaltfläche **Zurücksetzen** unter dem Menünamen.
Mit einer Meldung werden Sie gefragt, ob Sie das Kontextmenü tatsächlich zurücksetzen möchten.

So schließen Sie ein Kontextmenüfenster:

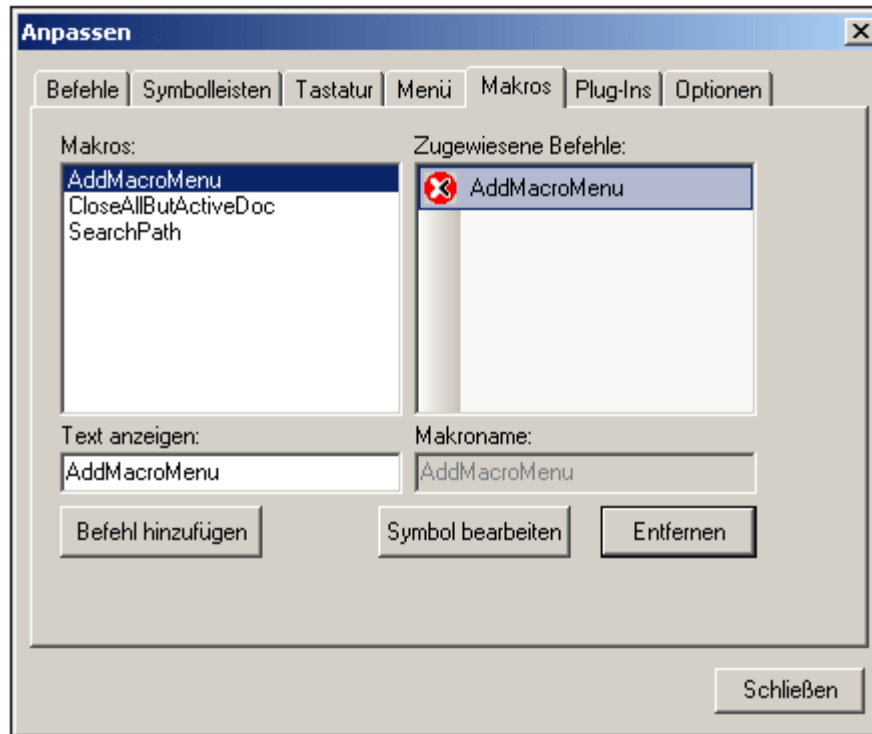
1. Klicken Sie auf das Symbol **Schließen** rechts oben in der Titelleiste oder
2. Klicken Sie auf die Schaltfläche "Schließen" im Dialogfeld "Anpassen".

Menüschatten

- Klicken Sie auf das Kontrollkästchen Menüschatten, wenn Menüs mit Schatten dargestellt werden sollen.

Makros

Auf dem Register Makros können Sie Makros, die Sie zuvor in der XML Skript-Umgebung erzeugt wurden, in eine Symbolleiste oder ein Menü einfügen.



So fügen Sie das Makrosymbol in eine Symbolleiste oder ein Menü ein:

1. Starten Sie die Skriptumgebung über **Extras | Skripting-Umgebung**.
2. Doppelklicken Sie auf den Eintrag XMLSpyMacros im Ordner Module des Projekts **XMLSpyGlobalScripts**.
Zuvor definierte Makros werden im rechten Fenster angezeigt.
3. Wechseln Sie zurück in XMLSpy und wählen Sie **Extras | Anpassen** und klicken Sie auf das Register Makros.
Die in der Skriptumgebung definierten Makros werden nun im Makros Listenfeld links angezeigt.
4. Klicken Sie auf den Namen des Makros und dann die Schaltfläche "Befehl hinzufügen".
Der Makroname wird im Listenfeld "Zugewiesene Befehle" angezeigt.
5. Klicken Sie im Listenfeld "Zugewiesene Befehle" auf den Namen des Makros und ziehen Sie ihn in die Symbolleiste oder das Menü.

So bearbeiten Sie ein Makrosymbol:

- Klicken Sie auf die Schaltfläche **Symbol bearbeiten**.

So löschen Sie ein Makro aus der Liste der zugewiesenen Befehle:

- Klicken Sie auf die Schaltfläche **Entfernen**.

Anmerkung: Wenn ein Makro als "zugewiesener Befehl" ausgewählt wurde, können Sie dafür [ein Tastaturkürzel definieren](#). Wählen Sie auf dem Register "Tastatur" des Dialogfelds "Anpassen" in der Auswahlliste "Kategorie" die Kategorie "Makros" aus, klicken Sie auf das

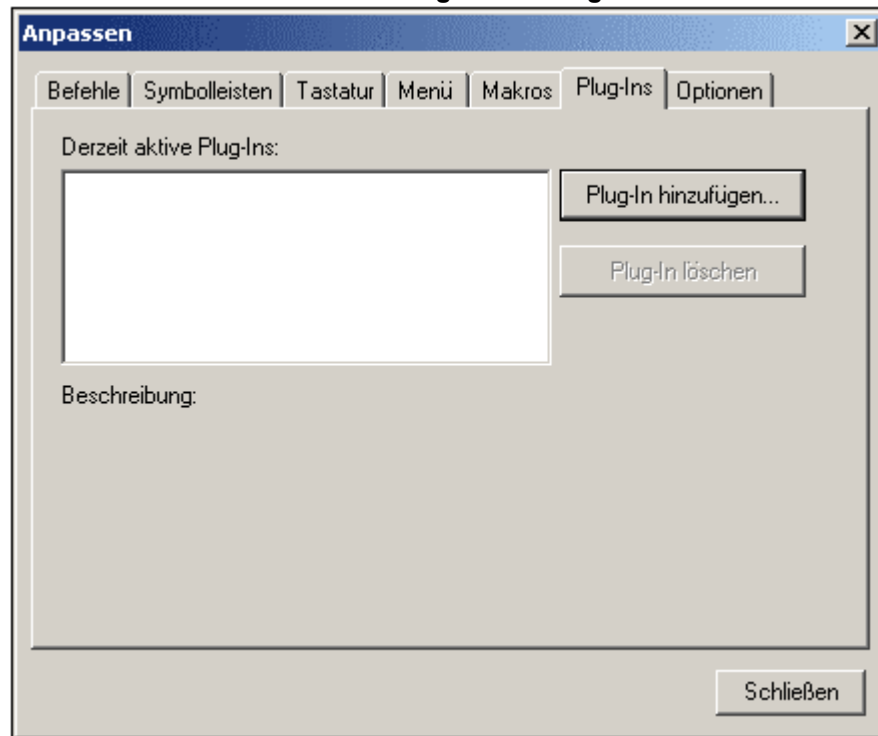
gewünschte Makro und definieren Sie das Tastenkürzel. Sie sollten ein Makro als "zugewiesenen Befehl" definieren, damit für Makros Tastenkürzel definiert werden können.

Plug-Ins

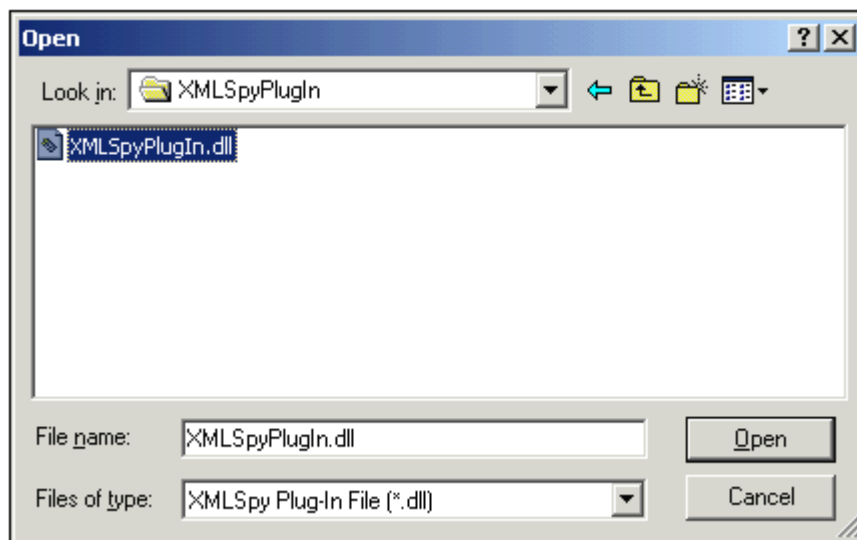
Auf dem Register Plug-Ins können Sie Plug-Ins in eine Symbolleiste oder ein Menü einfügen.

So fügen Sie das Symbol für ein Plug-In in eine Symbolleiste oder Menü ein:

1. Klicken Sie auf die Schaltfläche **"Plug-In hinzufügen"**.



2. Wählen Sie den Ordner aus und klicken Sie auf die Plug-In Datei um diese zu markieren (in diesem Beispiel: XMLSpyPlugIn.dll).



3. Klicken Sie auf die Schaltfläche **Öffnen** um das Plug-In zu installieren.

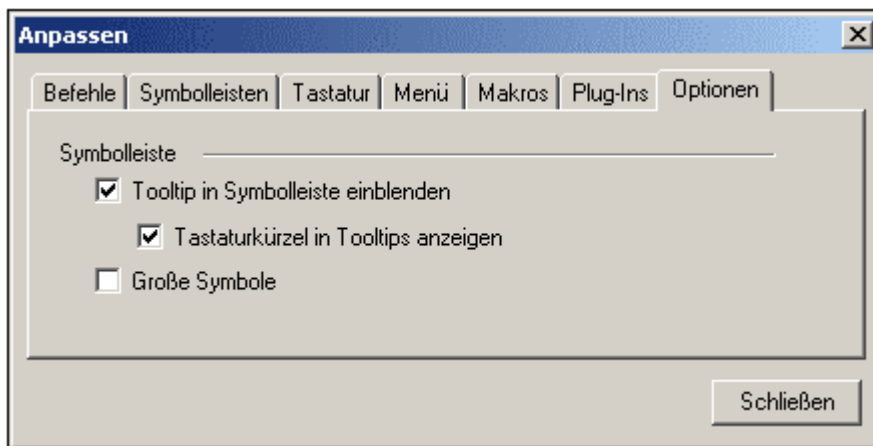
Der Name des Plug-Ins wird in der Liste "Derzeit aktive Plug-Ins" angezeigt und das Symbol des/der Plug-Ins wird in einer neuen Symbolleiste angezeigt.

So entfernen Sie ein Plug-In:

1. Klicken Sie auf den Namen des Plug-Ins in der Liste "Derzeit aktive Plug-Ins" und klicken Sie auf die Schaltfläche "Plug-In löschen".

Optionen

Auf dem Register Optionen können Sie verschiedene, allgemeine Einstellungen vornehmen.



Symbolleiste

Wenn das Kontrollkästchen **Tooltip in Symbolleiste einblenden** 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, ein Tastaturkürzel.

Das Kontrollkästchen "Tastaturkürzel in Tooltips anzeigen" ermöglicht es Ihnen, auszuwählen, ob die Tastaturkürzel (wo vorhanden) angezeigt werden sollen.

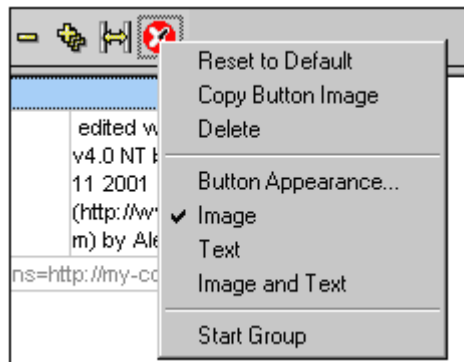
Mit dem Kontrollkästchen Große Symbole können Sie die Größe der angezeigten Symbole beeinflussen.

Kontextmenü 'Anpassen'

Über das Kontextmenü "Anpassen" können Sie Symbole und Menüobjekte Ihren Ansprüchen entsprechend anpassen.

So öffnen Sie das Kontextmenü "Anpassen":

1. Zuerst öffnen Sie das Menü Anpassen über den Menübefehl **Extras | Anpassen**.
2. Dann setzen Sie den Mauszeiger auf das Symbol (oder Menü) und **rechtsklicken Sie**, um das Kontextmenü zu öffnen.



Zurücksetzen

Derzeit keine Funktion.

Schaltflächensymbol kopieren

Das Symbol wird in die Zwischenablage kopiert.

Löschen

Mit dieser Option wird das Symbol oder Menü gelöscht, auf welches Sie zuvor rechtsgeklickt haben. Wenn Sie ein Menü löschen, werden alle enthaltenen Menüoptionen gelöscht!

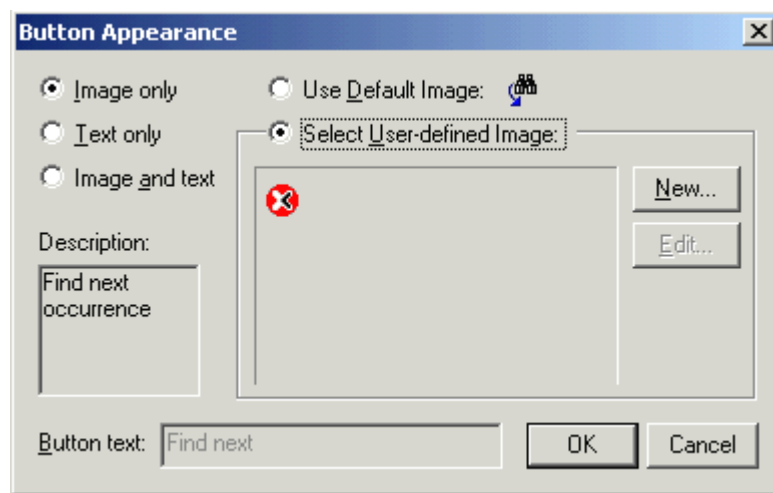
So stellen Sie ein gelöscht Menü wieder her:

- Wählen Sie das Register Menü
- Wählen Sie das Menü, welches wiederhergestellt werden soll (XMLSpy oder Standardmenü).
- Klicken Sie auf die Schaltfläche Zurücksetzen unter dem Menü im Dropdown-Listenfeld.

Schaltflächendarstellung

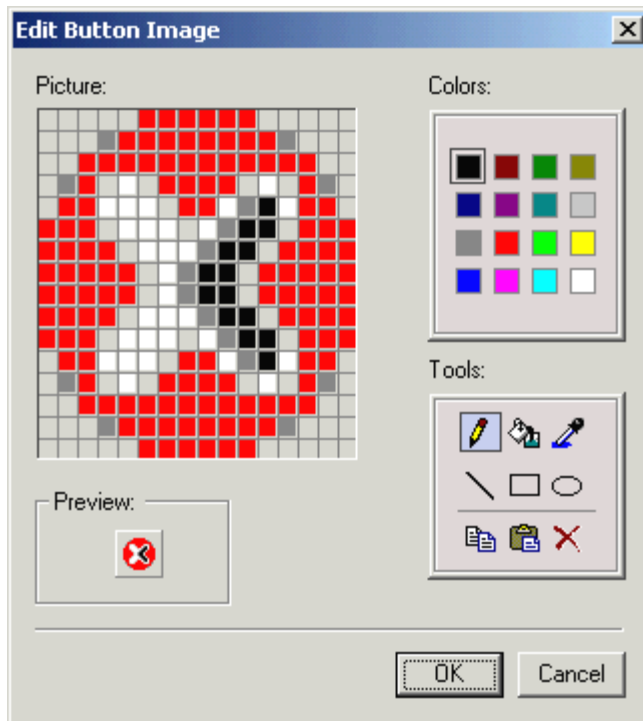
Mit dieser Option können Sie Schaltflächensymbole und Text bearbeiten. Derzeit kann nur das Symbol für Makros bearbeitet werden (Standardsymbol XMLSpy).

Mit den Optionsfeldern "Nur Symbol", "Nur Text" oder "Symbol und Text" können Sie definieren, was Sie editieren möchten. Klicken Sie auf das Optionsfeld **Benutzerdefiniertes Symbol** und klicken Sie auf eines der verfügbaren Symbole.



Klicken Sie auf die Schaltfläche "Bearbeiten" um das Dialogfeld **Schaltflächensymbol editieren** zu öffnen. Der Schaltflächentext kann bearbeitet werden, wenn Sie entweder die

Option "Nur Text" oder "Symbol und Text" ausgewählt haben.



Symbol

Mit dieser Option ändern Sie die Textbeschreibung einer Funktion in das grafische Symbol für diese Funktion.

Text

Mit dieser Option ändern Sie die grafische Anzeige des Symbols in seine Textbeschreibung.

Symbol und Text

Mit dieser Option wird die gleichzeitige Darstellung des Symbols und des Textes aktiviert.

Gruppe starten

Mit dieser Option wird links neben dem Symbol, auf welches Sie rechtsgeklickt haben, ein vertikaler Trennungsstrich angezeigt.

18.13.9 Symbolleisten und Fenster zurücksetzen

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 deren Größe angepasst haben und die Originaleinstellungen wiederherstellen möchten.

18.13.10 Optionen

Mit dem Befehl "Optionen" können Sie globale Programmeinstellungen in einem mit Registern versehenen Dialogfeld vornehmen. Diese Einstellungen werden in der Registry gespeichert und auf alle aktuellen und zukünftigen Dokumentfenster angewendet. Die Schaltfläche "Übernehmen" zeigt die Änderungen in den aktuell geöffneten Dokumenten an und speichert die Einstellungen. Die Änderungen sehen Sie sofort in den Fenstern im Hintergrund.

In diesem Abschnitt werden die einzelnen Register des Dialogfelds "Optionen" beschrieben.

Datei

Das Register "**Datei**" definiert die Art, wie XMLSpy Ihre XML-Dokumente öffnet und speichert. Weitere Einstellungen hierzu finden Sie auf dem Register [Kodierung](#).

Öffnen/Neue Datei in Grid-Ansicht

Sie können festlegen, ob eine existierende oder eine neue Datei entweder in der Grid- oder der Text-Ansicht geöffnet werden soll. Zusätzlich können Sie beim Auswählen der Grid-Ansicht auswählen, ob alle Zeilen ausgeklappt werden sollen.

Auto-laden 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. XMLSpy wird Sie fragen, ob die Datei neu geladen werden soll, sobald eine Änderung in einem offenen Dokument entdeckt wurde.

Validierung

Wenn Sie Dokument Type Definitions (DTDs) oder Schemas zur Definition der Struktur Ihrer XML-Dokumente verwenden, können Sie das Dokument beim Öffnen oder Speichern automatisch auf Gültigkeit überprüfen lassen. Sie haben beim Speichern und Öffnen die Option, Dateien nur zu validieren, wenn die Dateigröße eine von Ihnen definierte Größe nicht übersteigt. Wenn das Dokument ungültig ist, wird eine Fehlermeldung angezeigt und die Operation wird ohne weitere Benachrichtigung fortgesetzt. Mit XMLSpy können Sie die Dateien auch in den Speicher cachieren, sodass unnötiges Neuladen entfällt (z.B. wenn das referenzierte Schema über eine URL aufgerufen wird). Falls in Ihrer schemaLocation-Deklaration eine URL verwendet wird, deaktivieren Sie die Option "DTD/Schema Dateien im Cachespeicher", um am Schema vorgenommene Änderungen sofort anzuzeigen und nicht die im Cache gespeicherte Version.

des Schemas zu verwenden.

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 | Schema-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.

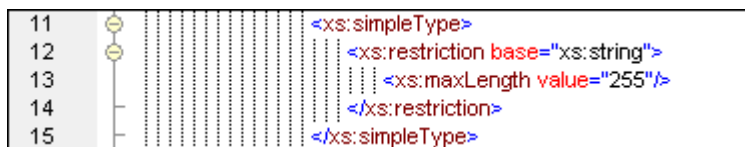
Zeilenumbrüche

Wenn Sie eine Datei öffnen, werden die Zeichen für Zeilenumbrüche beibehalten, wenn Sie die Option **Alte beibehalten** gewählt haben. Alternativ dazu können Zeilenumbrüche in **CR&LF** für PCs, **CR** für MacOS, oder **LF** für Unix kodiert werden.

Keine Ausgabe-Formatierung für

In der Textansicht können Sie Child-Elemente aus Gründen der besseren Übersichtlichkeit einrücken (siehe **Datei speichern**). Sie können jedoch den definierten Einzug für einzelne Elemente auch außer Kraft setzen, indem Sie den Namen des Elements in das Feld **Keine Ausgabe-Formatierung für** eingeben. Alle in diesem Feld definierten Elemente werden ohne Whitespaces zwischen aufeinanderfolgenden untergeordneten Elementen formatiert. (siehe *Abbildung*).

Hierarchische Einrückung für alle Elemente:

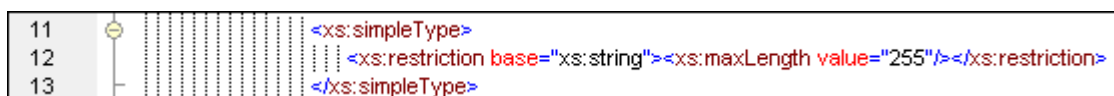


```

11 <xs:simpleType>
12   <xs:restriction base="xs:string">
13     <xs:maxLength value="255"/>
14   </xs:restriction>
15 </xs:simpleType>

```

Für das Element `xs:restriction` wurde die Einstellung **Keine Ausgabeformatierung** definiert.



```

11 <xs:simpleType>
12   <xs:restriction base="xs:string"><xs:maxLength value="255"/></xs:restriction>
13 </xs:simpleType>

```

Klicken Sie anschließend auf **OK**, um die Einstellung zu bestätigen und das Dialogfeld "Optionen" zu schließen.

Dateiarten

Auf dem Register "Dateiarten" können Sie für XMLSpy verschiedene Einstellungen für die einzelnen Dateiarten festlegen.

Wählen Sie eine Dateiart im Listenfeld "Dateiarten" aus, um folgende Funktionen für die ausgewählte Dateiart anzupassen:

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.

XML Konformität

XMLSpy bietet für verschiedene Dateiarten spezifische Editierfunktionen und andere Funktionen, die Sie durch Einstellung der Konformität in dieser Option festlegen können. In XMLSpy können Sie für eine Dateiart festlegen, dass sie mit XML-, XQuery, ZIP, JSON und anderer (Text)-Syntax konform sein muss. Des Weiteren lässt sich für XML-Dateiarten die Konformität mit XML, DTD and XML Entities festlegen. Für viele Dateiarten ist bereits eine standardmäßige Konformitätseinstellung festgelegt, die am besten beibehalten werden sollte, es sei denn Sie fügen eine neue Dateiart hinzu oder möchten für eine bestimmte Dateiart die Konformität explizit ändern.

Standard Ansicht

In dieser Gruppe definieren Sie, welche Standardansicht für welche Dateiart verwendet werden soll.

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.

Leere Elemente im Kurzformat "<E/>" speichern

Manche XML-Tools haben Probleme, die Kurzform `<Element/>` für leere Elemente zu verstehen (die in der XML 1.0 Spezifikation definiert wurde). Sie können XMLSpy anweisen, diese

Elemente in der Langform zu speichern $\langle \text{Element} \rangle \langle \text{Element} \rangle$, die gleichermaßen gültig ist.

Schaltfläche "Dateierweiterung hinzufügen"

Eine neue Dateierweiterung wird in die Liste der Dateiartern aufgenommen. Mit den Optionen in diesem Dialogfeld definieren Sie die Einstellungen für die neue Dateiartern.

Schaltfläche "Markierte Dateierweiterung löschen"

Löscht die Dateiartern und die dazugehörigen Einstellungen.

Klicken Sie anschließend auf **OK**, um die Einstellung zu bestätigen und das Dialogfeld "Optionen" zu schließen.

Bearbeiten

Auf dem Register "**Bearbeiten**" können Sie Bearbeitungseinstellungen für XMLSpy definieren.

Editier-Hilfen

- ☒ Eingabehilfen anzeigen
 - ☒ Beim Öffnen Eingabehilfen laden
 - Sortieren nach: ☒ Attribut ☐ Element
 - Pflichtfeld zuerst: ☒ Attribut ☐ Element
- ☒ Pflicht-Child-Elem. an neue Elem. autom. anhängen
 - ☐ Anzahl der Elemente minimieren
 - ☒ Nur obligatorische Elemente erzeugen
 - ☐ Alle Elemente erzeugen
- ☐ Nicht obligatorische Attribute erzeugen
- ☐ Elementinhalt von nillable Elementen als nicht-obligatorisch behandeln
- Bei Elementen mit abstract type nach
 - ☒ Möglichkeit bei xsi:type non-abstract type verwenden

Textansicht

- ☒ Autokomplettierung in Textansicht
 - Autokomplettierung und Eingabehilfen deaktivieren, wenn Datei größer ist als: MB
- Tabellen-Ansicht**
 - ☒ Autotabellen-Erkennung bei wiederh. Elem.
 - ☐ Tabelle für alle wiederh. Elem. erzeugen
 - ☐ Einzel-Subelemente als Tabelle zeigen
 - Keine Auto-Tabellen bei folgenden Elementen:
- In die Zwischenablage der Grid-Ansicht kopieren als:
 - ☒ XML-Text
 - ☐ Strukturierten Text (TAB-begrenzt)

Editier-Hilfen

XMLSpy bietet Ihnen intelligente Eingabehilfen zum Bearbeiten von Dokumenten. Verschiedene Bearbeitungseinstellungen für diese Eingabehilfen lassen sich hier anpassen. Die Einstellungen variieren je nach Dateityp. 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.

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.

In die Zwischenablage der Grid-Ansicht kopieren als

Sie können das Format, in dem Daten über die Zwischenablage in Fremdanwendungen exportiert werden sollen, auswählen. Wenn Sie XML-Text wählen, wird der Inhalt der Zwischenablage genau wie die resultierende XML-Datei formatiert und mit Tags versehen.

Im strukturierten Textmodus wird versucht, den Inhalt der Zwischenablage als Tabelle (Verwendung in Spreadsheet- oder Datenbankapplikationen) zu formatieren. Diese Option betrifft nicht das interne Zwischenablagenformat, das XMLSpy zum Kopieren und Einfügen verwendet.

Tabellen-Ansicht

Weiters können Sie einstellen, wie XMLSpy sich wiederholende Elemente in der [Tabellenansicht](#) darstellen soll.

Klicken Sie anschließend auf **OK** um die Einstellungen zu bestätigen und das Dialogfeld zu schließen.

Ansicht

Auf dem Register "Ansicht" können Sie die Darstellung der XML-Dokumente in XMLSpy anpassen.

Grid-Ansicht

Die XML-Elemente können in der Grid-Ansicht zu einer einzigen Zeile, in der der Name des Elements angezeigt wird, reduziert werden. In dieser Zeile können auch die Attribute des Elements angezeigt werden. Wenn die Option *Attributvorschau einblenden* aktiviert ist, werden die Attribute bei reduzierten Elementen grau angezeigt. Andernfalls werden die dazugehörigen Attribute bei reduzierten Elementen nicht angezeigt. Sie können beim Bearbeiten der Dateien

automatisch die [optimale Breite](#) anwenden. Zusätzlich dazu kann eine Beschränkung für die maximale optimale Breite und Höhe einer Zelle definiert werden. Wenn der Inhalt nicht der gesamte Inhalt einer Zelle angezeigt werden kann, so wird dies durch ein Auslassungszeichen angezeigt.

Pretty-Print

Bei Auswahl des Befehls **Bearbeiten | Pretty-Print** in der Textansicht oder wenn Sie aus einer anderen Ansicht in die Textansicht wechseln, wird das Dokument mit Pretty-Print formatiert und zwar mit oder ohne Einrückungen, je nachdem, ob die Option "Einrückungen verwenden" in diesem Dialogfeld aktiviert ist oder nicht. Die Anzahl der Einrückungen kann im Bereich "Tabulatoren" im Dialogfeld [Einstellungen für Textansicht](#) vorgenommen werden.

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.

Authentic-Ansicht

XML-Dateien basierend auf einem **StyleVision Power Stylesheet** (*.sps), werden bei Aktivieren dieser Option automatisch in der Authentic-Ansicht geöffnet.

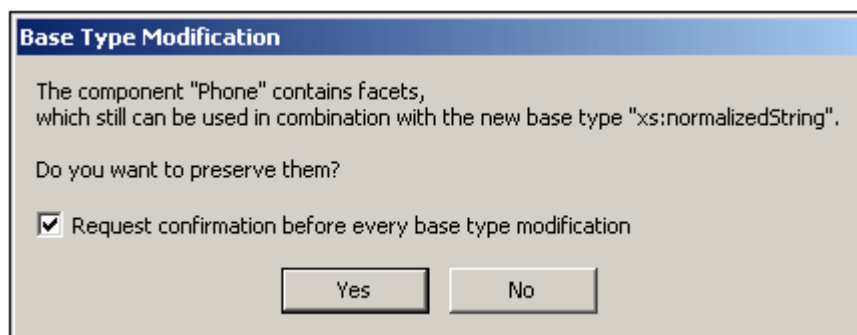
Browser-Ansicht

Sie können die Browser-Ansicht in einem separaten Fenster öffnen, sodass Sie die Ansicht, in der Sie arbeiten, und die Browser-Ansicht nebeneinander sehen.

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:

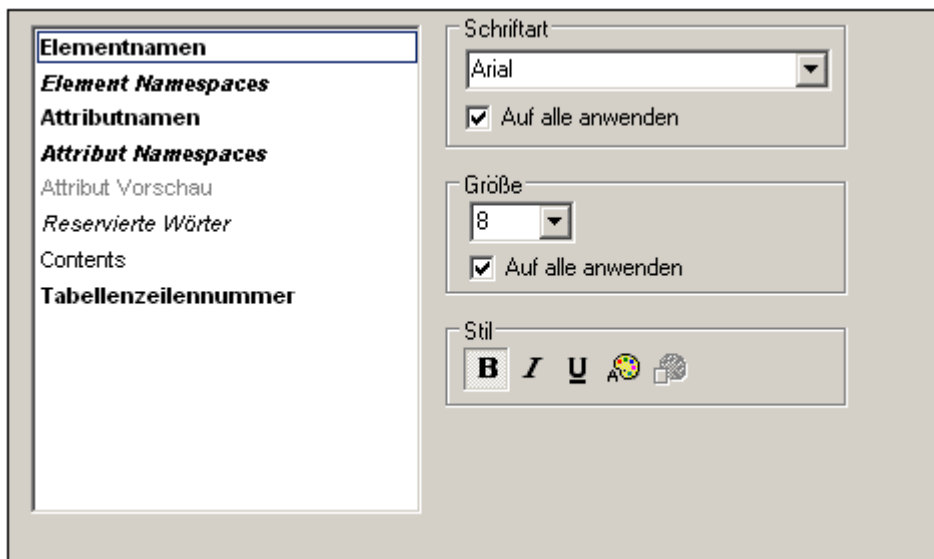
- *Inhalt beibehalten*: Wenn die zum Definieren des "derived Type" verwendeten Definitionen mit dem neuen Basistyp verwendet werden können, werden die Definitionen automatisch beibehalten, wenn diese Option aktiviert ist.
- *Vor jeder Änderung... Bestätigung anfordern*: Nachdem Sie den Basistyp geändert 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.



Klicken Sie anschließend auf **OK** um die Einstellungen zu bestätigen und das Dialogfeld zu schließen.

Grid-Schriftarten

Auf dem Register **Grid-Schriftarten** können Sie das Aussehen von Texten in der [Grid-Ansicht](#) anpassen.



Schriftart

Wählen Sie Schriftart und Schriftgröße, die in der Grid Ansicht verwendet werden sollen. Dieselben Einstellungen werden auch für das Drucken verwendet. Deswegen können nur **TrueType** Schriftarten ausgewählt werden.

Größe

Wählen Sie hier die gewünschte Größe aus. Wenn Sie für alle Elemente die selbe Schriftgröße verwenden möchten, aktivieren Sie das Kontrollkästchen "Auf alle anwenden".

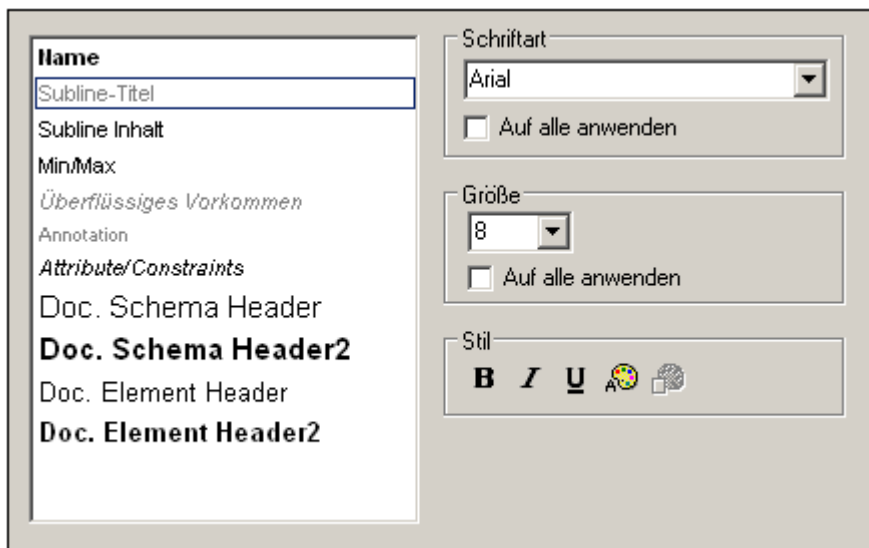
Stil

Stil und Farbe können im über die Auswahl Schaltflächen eingestellt werden. Die Einstellungen werden sofort als Vorschau in der in Liste auf der linken Seite angezeigt. So sehen Sie sofort, wie Ihr Dokument aussehen wird.

Klicken Sie anschließend auf **OK** um die Einstellungen zu bestätigen und das Dialogfeld zu schließen.

Schemaschriftarten

Auf dem Register **Schemaschriftarten** können Sie das Aussehen von Texten in der [Schema-Ansicht](#) anpassen.



Schriftart

Wählen Sie Schriftart und Schriftgröße, die in der Schema-Ansicht verwendet werden sollen. Dieselbe Schriftgröße und Schriftart wird zum Drucken und Erstellen von [Schema Dokumentation](#) verwendet. Deswegen können nur TrueType Schriftarten ausgewählt werden. Komponenten mit dem Präfix "Doc." werden in der Schema-Dokumentation verwendet.

Größe

Wählen Sie hier die gewünschte Größe aus. Wenn Sie für alle Elemente die selbe Schriftgröße verwenden möchten, aktivieren Sie das Kontrollkästchen "Auf alle anwenden".

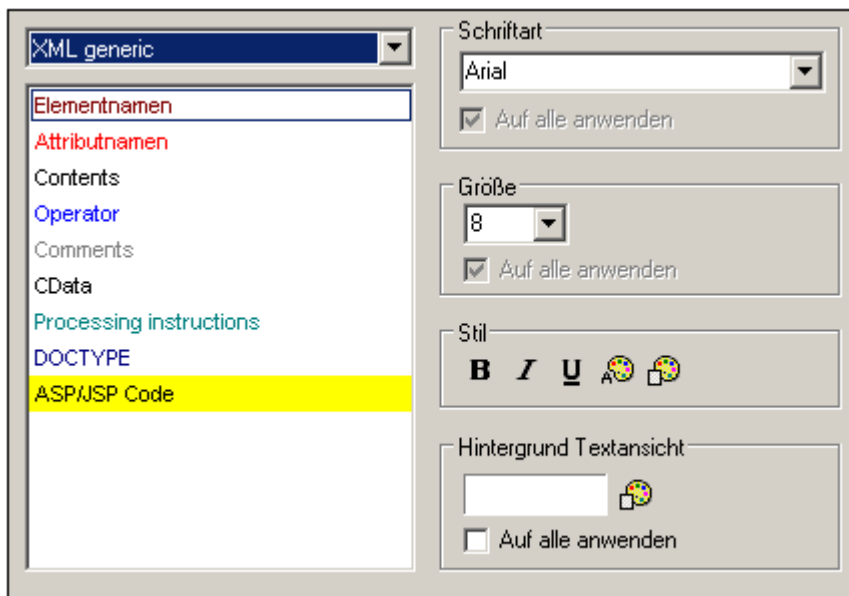
Stile

Stil und Farbe können über die Auswahl Schaltflächen eingestellt werden. Die Einstellungen werden sofort als Vorschau in der Liste auf der linken Seite angezeigt. So sehen Sie sofort, wie Ihr Dokument aussehen wird.

Klicken Sie anschließend auf **OK** um die Einstellungen zu bestätigen und das Dialogfeld zu schließen.

Textschriftarten

Auf dem Register **Textschriftarten** können Sie das Aussehen von Texten in der Text-Ansicht anpassen. Sie können das Aussehen von Textelementen je nach der Art des Textelements anpassen. So können z.B. die Namen von Elementen und Attributen in unterschiedlichen Farben erscheinen.



Die Textelementtypen werden in drei Gruppen eingeteilt:

- XML generic
- XQuery
- CSS
- JSON

So passen Sie Textschriftarten an:

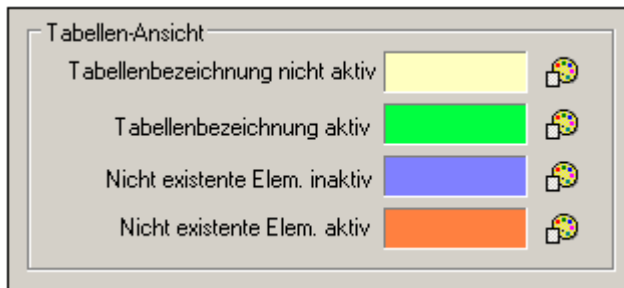
1. Wählen Sie in der Auswahlliste links oben die Art des Dokuments aus, für das Sie Textschriftarten anpassen möchten. Dabei werden die Textelementtypen für diesen Dokumenttyp im Kästchen unterhalb der Auswahlliste angezeigt. *(In der Abbildung oben wurde als Dokumenttyp XML generic ausgewählt.)*
2. 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. Sie können die Schriftart, Größe und den Stil, die Schriftfarbe und Texthintergrundfarbe auswählen. Außerdem können Sie auch eine Hintergrundfarbe für die gesamte Textansicht auswählen.

Anmerkung: Beachten Sie, dass für alle Textelementtypen dieselbe Schriftart, derselbe Stil und dieselbe Größe verwendet werden. Sie können für einzelne Textarten nur Textfarbe und Hintergrundfarbe ändern. Dies aktiviert die Syntaxfärbung.

Klicken Sie anschließend auf **OK** um die Einstellungen zu bestätigen und das Dialogfeld zu schließen.

Farben

Auf dem Register **Farben** können Sie die in der Tabellen- oder Grid-Ansicht verwendeten Hintergrundfarben anpassen. In der Abbildung unten wurden die Standardfarben durch Klicken auf die Palettensymbole neben den einzelnen Einträgen und Auswahl einer Farbe geändert.



Farben der Tabellenüberschrift

Die Optionen "Tabellenbezeichnung nicht aktiv" und "Tabellenbezeichnung aktiv" beziehen sich auf die Spalten- und Zeilenüberschriften. Im unten gezeigten Screenshot sind die Tabellenbezeichnungen deaktiviert. Die Farbe entspricht den Einstellungen im obigen Dialogfeld.

Name		Organization Chart			
Office (2)					
	Name	Desc	Location	Address	Address_EU
1	Nanonull, Inc.	Desc	US	Address	
2	Nanonull Europe, AG	Desc	EU		Address_EU

Die im Feld "Tabellenbezeichnungen aktiv" definierte Farbe wird aktiviert, wenn alle Überschriften ausgewählt sind (*Screenshot unten*) - nicht aber, wenn nur einzelne Überschriften ausgewählt werden. Sie sehen dies im nachfolgenden Screenshot, in dem die im obigen Dialogfeld definierten Farben verwendet werden. Die Auswahl aller Tabellenüberschriften erfolgt durch Klicken auf die Zelle, in der sich beide Überschriftszeilen schneiden oder durch Auswahl des Elements, das als Tabelle dargestellt wird, bzw. durch Auswahl eines seiner übergeordneten Elemente.

Name		Organization Chart			
Office (2)					
	Name	Desc	Location	Address	Address_EU
1	Nanonull, Inc.	Desc	US	Address	
2	Nanonull Europe, AG	Desc	EU		Address_EU

Nicht-existente Elemente

Wenn ein Element oder ein Attribut im XML-Dokument nicht vorhanden ist, kann es unterschiedliche Hintergrundfarben zugewiesen erhalten, wenn es ausgewählt und nicht ausgewählt ist. Ein Beispiel dafür sehen Sie im nachfolgenden Screenshot, in welchem die erste Zeile ausgewählt ist.

Name		Organization Chart			
Office (2)					
	Name	Desc	Location	Address	Address_EU
1	Nanonull, Inc.	Desc	US	Address	
2	Nanonull Europe, AG	Desc	EU		Address_EU

Bitte beachten Sie: Zusätzlich zu den hier definierten Farben, verwendet XMLSpy die normalen Einstellungen für Menüfarben und hervorgehobenen Text, die in der Systemsteuerung Ihrer Windowsinstallation unter Anzeigeeinstellungen festgelegt sind.

Klicken Sie anschließend auf **OK** um die Einstellungen zu bestätigen und das Dialogfeld zu schließen.

Kodierung

Auf dem Register **Kodierung** definieren Sie die Einstellungen für die Dateikodierung.

Standardkodierung für neue XML-Dateien

Unicode UTF-8

☒ Little-endian Bytefolge

☐ Big-endian Bytefolge

Dateien mit unbekannter Kodierung öffnen als:

Unicode UTF-8

Nicht-XML-Dateien öffnen in

Codepage 1252 (Western)

BOM

☒ BOM immer erstellen, wenn nicht UTF-8

☐ Festgestellten BOM beim Speichern beibehalten

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](#) 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.

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](#) 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*).

Klicken Sie anschließend auf **OK** um die Einstellungen zu bestätigen und das Dialogfeld zu schließen.

XSL

Auf dem Register **XSL** (*Abbildung unten*) werden die Optionen für [XSLT-Transformationen](#) und über die Applikation ausgeführte [XSL-FO-Transformationen](#) definiert.

☒ Integrierter XSLT-Prozessor (wird immer für das XSLT Debuggen verwendet!)
☐ Microsoft XML Parser (MSXML): ☐ v3.0 ☐ v4.0 ☐ v6.0 ☒ Version automatisch wählen
☐ Externes XSL-Transformationsprogramm:

Geben Sie die Befehlszeilen-Parameter für das externe XSL-Transformationsprogramm in folgender Form ein:
 Program.exe %1 %2 %3

%1 wird durch den XML Eingabe-Dateinamen ersetzt, %2 durch den Ausgabe-Dateinamen und %3 (optional) durch den Namen der XSL Stylesheet-Datei. Sie können auch weitere notwendige Parameter anhängen.

☒ Ausgabe des externen Programms nach der Transformation anzeigen
☒ Ausgabefehler des externen Programms nach Transformation anzeigen

Standard-Dateierweiterung der Ausgabedatei: ☒ Ausgabefenster wiederverwenden

☒ Dateierweiterung aus <xsl:output method=""> Attribut verwenden, falls verfügbar

Pfad zum XSL-FO Transformationsprozessor eingeben (bei Verwendung von FOP bitte Pfad zu fop.bat eingeben):

☒ Oben ausgewählten XSLT-Prozessor zur Durchführung des XSLT-Teils und anschließend XSL-FO-Prozessor für FO-Teil verwenden
☐ XSL-FO-Prozessor für XSLT- und FO-Teil der Transformation verwenden

XSLT-Transformationen

XMLSpy enthält den Altova XSLT 1.0-Prozessor und den Altova XSLT 2.0-Prozessor, die 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 verwendet. Dies gilt sowohl für XSLT-Transformationen als auch für das XSLT Debugging mit dem XSLT/XQuery-Debugger von XMLSpy.

Zur Transformation von XML-Dokumenten mittels XSLT können Sie die folgenden Prozessoren verwenden:

- den integrierten Altova XSLT-Prozessor (einschließlich Altova XSLT 1.0- und Altova XSLT 2.0-Prozessor).
- den MSXML 3.0-, 4.0- oder 6.0-Parser (welcher bereits vorinstalliert ist). Wenn Sie wissen, welche Version des MSXML Parsers auf Ihrem Rechner läuft, könnten Sie ihn auswählen. Andernfalls wird die Version automatisch von der Applikation ausgewählt. (Die Option *Version automatisch auswählen* ist standardmäßig aktiv.) In diesem Fall versucht die Applikation, die neueste verfügbare Version auszuwählen.
- einen externen XSLT-Prozessor Ihrer Wahl. Sie müssen den Befehlszeilenstring für den externen XSLT-Prozessor definieren. Dazu stehen die folgenden Variablen zur Verfügung:
 - %1 = zu verarbeitendes XML-Dokument
 - %2 = zu generierende Ausgabedatei
 - %3 = zu verwendendes XSLT Stylesheet (wenn das XML-Dokument keine Referenz auf ein Stylesheet enthält)

Der Befehl zur Ausführung einer einfachen Transformation mit dem Saxon (XSLT 1.0) Prozessor lautet z.B.:

```
saxon.exe -o output.xml input.xml stylesheet.xslt
parameter-name=parameter-value
```

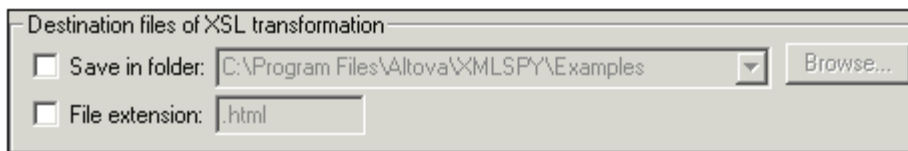
Um diesen Befehl von der Applikation aus zu starten, aktivieren Sie das Optionsfeld "Externes XSL-Transformationsprogramm" und geben Sie im Textfeld die folgende Zeile ein:

```
c:\saxon\saxon.exe -o %2 %1 %3 parameter-name=parameter-value
```

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"](#) definiert sind, werden nur an die internen Altova-XSLT-Prozessoren übergeben und nicht an andere XSLT-Prozessoren, die als XSLT-Standardprozessor konfiguriert wurden.

- Bei Auswahl der Option *Ausgabefenster wiederverwenden* wird das Ergebnisdokument bei nachfolgenden Transformationen im selben Ausgabefenster angezeigt. Wenn die XML-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](#) ebenfalls deaktiviert ist.



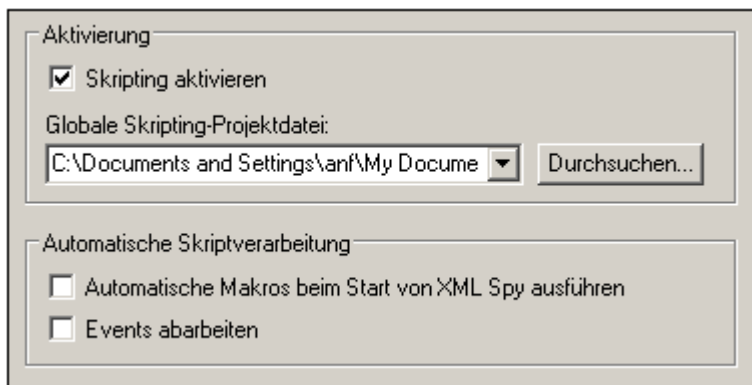
XSL-FO-Transformationen

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](#) 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](#)) oder vom XSLT-Prozessor, der möglicherweise bereits in den FO-Prozessor integriert ist, den Sie als Standard-FO-Prozessor für die Applikation definiert haben. Klicken Sie zur Auswahl zwischen den beiden Optionen auf das entsprechende Optionsfeld.

Klicken Sie anschließend auf **OK** um die Einstellungen zu bestätigen und das Dialogfeld zu schließen.

Skripting-Umgebung

Auf dem Register **Skripting-Umgebung** (*Abbildung unten*) können Sie die [Skripting-Umgebung](#) beim Start der Applikation aktivieren. Aktivieren Sie dazu das Kontrollkästchen *Skripting aktivieren*. Anschließend können Sie die globale Skripting-Projektdatei definieren. (*siehe Abbildung unten*).

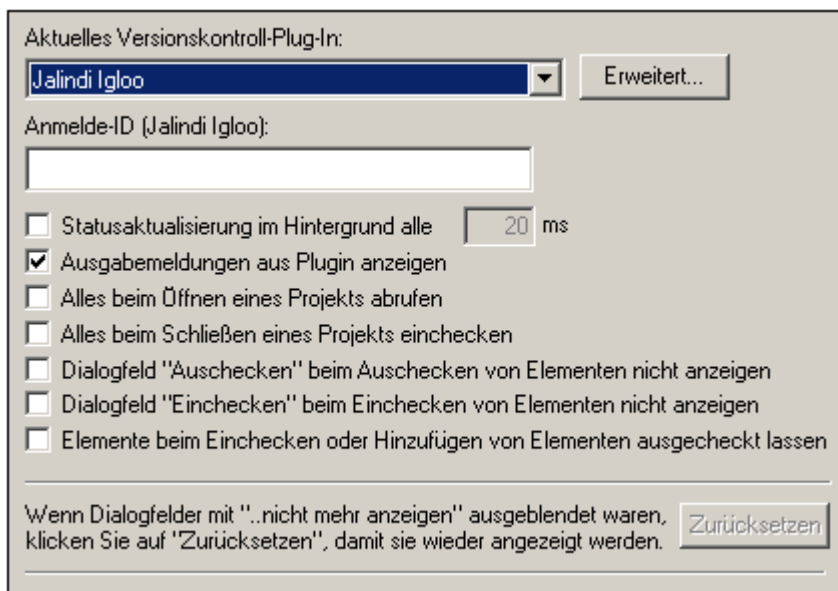


Um ein globales Skripting-Projekt für XMLSpy zu definieren, aktivieren Sie das Kontrollkästchen *Skripting aktivieren* und suchen Sie die gewünschte Altova Skripting-Projektdatei (.asprj-Datei). Außerdem können Sie folgende Einstellungen definieren: (i) ob automatische Makros im Skripting-Projekt beim 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.

Versionskontrolle

Auf dem Register **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

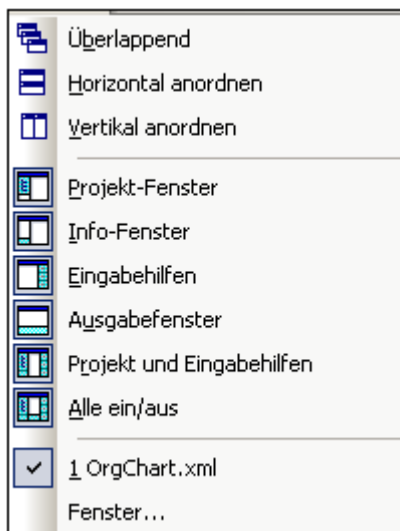
Es steht eine Reihe von Benutzereinstellungen zur Verfügung, darunter die folgenden:

- 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.

Klicken Sie anschließend auf **OK** um die Einstellungen zu bestätigen und das Dialogfeld zu schließen.

18.14 Menü "Fenster"

Mit Hilfe dieses Menüs können Sie die Dokumentfenster für Ihre XMLSpy Sitzung anordnen. Das Menü "Fenster" enthält Standardbefehle, die aus den meisten Windows-Applikationen bekannt sind.



Sie können die offenen Dokumente hintereinander oder nebeneinander anzeigen. Weiters können Sie Dokumentsymbole neu anordnen (nachdem Sie diese minimiert haben) und die verschiedenen Eingabehilfenfenster ein- oder ausblenden oder direkt über das Menü in ein beliebiges offenes Dokument-Fenster wechseln.

18.14.1 Überlappend

Mit diesem Befehl arrangieren Sie alle offenen Dokumentfenster, sodass sie übereinander und überlappend angezeigt werden.

18.14.2 Horizontal anordnen

Mit diesem Befehl arrangieren Sie alle offenen Dokumente **horizontal nebeneinander**, sodass alle Dokumentfenster gleichzeitig sichtbar sind.

18.14.3 Vertikal anordnen

Mit diesem Befehl arrangieren Sie alle offenen Dokumente **vertikal nebeneinander**, sodass alle Dokumentfenster gleichzeitig sichtbar sind.

18.14.4 Projekt-Fenster

Mit diesem Befehl schalten Sie das [Projekt-Fenster](#) ein oder aus.

Dieses Fenster ist abdockbar, d.h. wenn Sie auf die Titelleiste klicken, können Sie das Fenster an jede beliebige Stelle verschieben. Wenn Sie mit der rechten Maustaste auf die Titelleiste klicken, können Sie das Fenster andocken oder ausblenden.

18.14.5 Info-Fenster

Mit diesem Befehl schalten Sie das [Info-Fenster](#) ein oder aus.

Dieses Fenster ist andockbar, d.h. wenn Sie auf die Titelleiste klicken, können Sie das Fenster an jede beliebige Stelle verschieben. Mit Rechtsklick docken Sie das Fenster wieder an, oder verbergen es.

18.14.6 Eingabehilfen

Mit diesem Befehl blenden Sie alle drei Eingabehilfe-Fenster ein oder aus.

Dieses Fenster ist andockbar, d.h. wenn Sie auf die Titelleiste klicken können Sie das Fenster an jede beliebige Stelle verschieben. Mit Rechtsklick docken Sie das Fenster wieder an, oder verbergen es.

18.14.7 Ausgabefenster

Bei den Ausgabefenstern handelt es sich um eine Gruppe von drei mit Registern versehenen Ausgabebereichen. Diese sind das Fenster "Meldungen" (in dem Meldungen wie z.B. Validierungsergebnisse angezeigt werden), das Fenster "In Dateien suchen" und das Fenster "XPath" (in dem sie Ergebnisse der XPath-Auswertung angezeigt werden). In der Anfangseinstellung werden sie unterhalb des Hauptfensters geöffnet. Mit dem Befehl "Ausgabefenster" können Sie die Ausgabefenster ein- oder ausblenden.

Das Fenster für die Ausgabebereich kann an- oder abgedockt werden. Wenn Sie die Titelleiste ziehen, können Sie es verschieben und als freischwebendes Fenster anordnen. Um das Fenster anzudocken oder auszublenden, klicken Sie mit der rechten Maustaste auf die Titelleiste.

Eine ausführliche Beschreibung der Ausgabefenster finden Sie im Abschnitt "Text-Ansicht" unter [Ausgabefenster](#).

18.14.8 Projekt- und Eingabehilfen

Mit diesem Befehl können Sie das Projektfenster und die Eingabehilfen gleichzeitig ein- oder ausblenden.

18.14.9 Alle ein/aus



Mit diesem Befehl werden alle andockbaren Fenster zusammen ein- oder ausgeschaltet:

- [Projekt-Fenster](#)
- [Info-Fenster](#)
- alle drei Eingabehilfen-Fenster
- die [Ausgabefenster](#)

Diese Funktion ist besonders hilfreich, wenn Sie schnell alle anderen Fenster ausblenden möchten, um Ihr Dokument in maximaler Größe anzeigen zu können.

18.14.10 Liste aller offenen Fenster

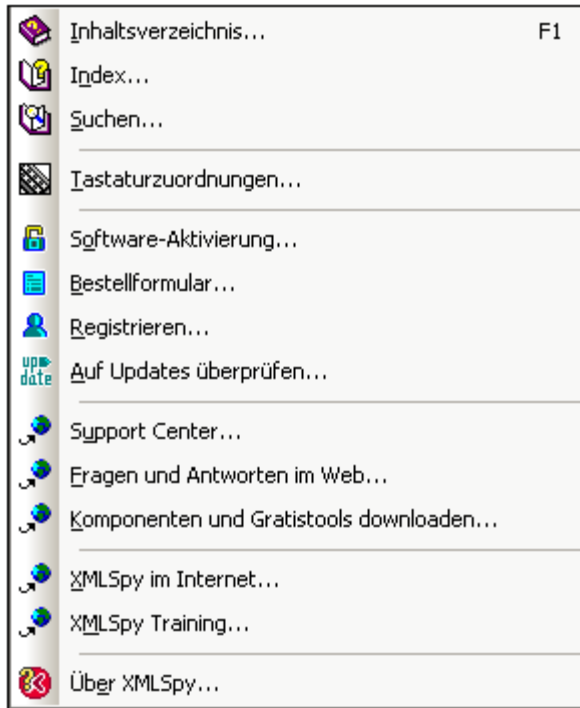
In dieser Liste werden alle derzeit offenen Fenster angezeigt. So können Sie schnell von einem Dokument zum anderen wechseln.



Hierzu können Sie auch die Tastaturkürzel Strg-TAB oder Strg F6 verwenden.

18.15 Menü "Hilfe"

Im Hilfemenü finden Sie alle Befehle, die Sie benötigen, um weitere Informationen zu spezifischen Themen von XMLSpy zu erhalten, sowie Links zu Informationen und Supportseiten auf unserem Webserver.



Im Hilfemenü finden Sie weiters den [Registrierungsdialog](#), über den Sie Ihren Keycode eingeben können, sobald Sie das Produkt erworben haben.

18.15.1 Inhaltsverzeichnis

Mit diesem Befehl erhalten Sie eine **hierarchische Darstellung** aller Kapitel und Themen, die in der Onlinehilfe enthalten sind. Verwenden Sie diesen Befehl, um direkt von XMLSpy in das Inhaltsverzeichnis zu gelangen.

Sobald das Hilfefenster geöffnet wurde, können Sie zwischen dem Inhaltsverzeichnis, dem [Index](#), und der [Suchfunktion](#) wählen. Auf dem Register "Favoriten" können Sie Lesezeichen zu Kapiteln innerhalb der Hilfe setzen.

18.15.2 Index

Mit diesem Befehl gelangen Sie in das **Stichwortverzeichnis** der Onlinehilfe. Sie können auch das Register Index in der Onlinehilfe verwenden.

Der Index bietet eine Liste von Stichwörtern, mit denen Sie durch Doppelklick zu den jeweiligen Abschnitten gelangen, die das Stichwort behandeln. Wenn es mehrere Abschnitte oder Kapitel gibt, in denen das Stichwort vorkommt, werden Sie aufgefordert, aus einer Liste das gewünschte Kapitel auszuwählen.

18.15.3 Suche

Mit dem Suchbefehl können Sie eine **Volltextsuche** in der gesamten Onlinehilfe starten.

- Geben Sie den Suchbegriff im Suchfeld ein und drücken Sie die Eingabetaste. Die Onlinehilfe zeigt eine Liste von Kapiteln an, in denen der Suchbegriff vorkommt.
- Doppelklicken Sie auf einen Eintrag in der Liste, um das entsprechende Kapitel aufzurufen.

18.15.4 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.

Tastaturhilfe		
  Kategorie: Datei Kürzelanzeige für: Standardeinstellung 		
Befehl	Tasten	Beschreibung
DateiAlleInaktivenSchließen		Alle offenen Dokumente mit Ausnahme des aktiven schlie...
DateiAllesSchließen		Alle offenen Dokumente schließen
DateiAllesSpeichern		Alle offenen Dokumente speichern
DateiAlsMailSenden...		Dokument per E-Mail senden
DateiBeenden		Verläßt die Anwendung; fragt, ob Dokumente gespeichert...
DateiDrucken...	Strg+P	Aktives Dokument drucken
DateiDruckereinrichtung...		Ändert den Drucker und die Druckoptionen.
DateiDruckvorschau		Druckvorschau
DateiKodierung...		Setzt oder ändert die Zeichensatz-Kodierung der aktuelle...
DateiNeu...	Strg+N	Erstellt ein neues Dokument.
DateiNeuLaden		Offene Datei neu laden
DateiSchließen		Schließt das aktive Dokument.
DateiSpeichern	Strg+S	Speichert das aktive Dokument.
DateiSpeichernUnter...		Speichert das aktive Dokument unter neuem Namen.
DateiÖffnen...	Strg+O	Öffnet ein bestehendes Dokument.

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"](#) Ihre eigenen Tastaturkürzel für verschiedene Menübefehle definieren.

18.15.5 Aktivierung, Bestellformular, Registrieren, Updates

Software-Aktivierung

Nachdem Sie Ihre Altova-Software heruntergeladen haben, können Sie sie entweder mit Hilfe eines kostenlosen Evaluierungs-Keycode oder eines käuflich erworbenen permanenten Lizenzkeycode aktivieren.

- **Kostenloser Evaluierungs-Keycode.** Wenn Sie die Software zum ersten Mal starten, wird das Dialogfeld "Software-Aktivierung" angezeigt. Es enthält eine Schaltfläche, über die Sie einen kostenlosen Evaluierungs-Keycode anfordern können. Geben Sie in das Dialogfeld, das daraufhin angezeigt wird, Ihren Namen, den Namen Ihrer Firma und Ihre E-Mail-Adresse ein und klicken Sie auf "Jetzt anfordern!". Der Evaluierungs-Keycode wird an die von Ihnen angegebene E-Mail-Adresse gesendet und sollte in wenigen Minuten eintreffen. Geben Sie den Schlüssel nun im Dialogfeld "Software-Aktivierung" in das Feld Key-Code ein und klicken Sie auf OK, um Ihr Altova-Produkt zu aktivieren. Die Software wird für einen Zeitraum von 30 Tagen aktiviert.
- **Permanenter Lizenz Keycode.** Das Dialogfeld "Software-Aktivierung" enthält eine Schaltfläche, über die Sie einen permanenten Lizenz-Keycode erwerben können. Wenn Sie auf diese Schaltfläche klicken, gelangen Sie zum Altova Online Shop, wo Sie einen permanenten Lizenz-Keycode für Ihr Produkt erwerben können. Bei Ihrem permanenten Lizenz-Keycode handelt es sich entweder um eine Einzelbenutzerlizenz oder um eine Mehrfachlizenz. Beide Arten werden per E-Mail zugesandt. Eine *Einzelbenutzerlizenz* enthält Ihre Lizenzdaten, Ihren Namen, die Firma, E-Mail-Adresse und Keycode. Eine *Mehrfachlizenz* enthält Ihre Lizenzdaten, ihren Firmennamen und den Keycode. Beachten Sie, dass gemäß Ihrer Lizenzvereinbarung auf den Computern Ihres Unternehmens nur so viele Versionen der Altova-Software installiert werden dürfen, wie lizenziert wurden (Mehrplatzlizenz).

Anmerkung: Stellen Sie bitte sicher, dass die von Ihnen in das Registrierungsdialogfeld eingegebenen Daten genau mit denen in Ihrer Lizenz-E-Mail übereinstimmen. Geben Sie außerdem bei Mehrplatzlizenzen den Namen des jeweiligen Benutzers in das Feld "Name " ein.

Das Dialogfeld "Software-Aktivierung" kann jederzeit über den Befehl **Hilfe | Software-Aktivierung** aufgerufen werden.

Bestellformular

Sobald Sie eine lizenzierte Version der Software bestellen möchten, klicken Sie im Dialogfeld "Software-Aktivierung" (*siehe oben*) auf die Schaltfläche **Permanenter Key-Code erwerben...** um zum sicheren Online-Shop von Altova weitergeleitet zu werden.

Registrierung

Wenn Sie Ihre Altova-Software nach der Aktivierung das erste Mal starten, erscheint ein Dialogfeld, in dem Sie gefragt werden, ob Sie Ihr Produkt registrieren möchten. Es gibt in diesem Dialogfeld drei Schaltflächen:

- **OK:** Leitet Sie zum Registrierungsformular weiter
- **Später erinnern:** Ruft ein Dialogfeld auf, in dem Sie auswählen können, wann Sie das nächste Mal erinnert werden möchten.
- **Abbrechen:** Schließt das Dialogfeld und unterdrückt die Anzeige in Zukunft. Wenn Sie das Produkt zu einem späteren Zeitpunkt registrieren möchten, verwenden Sie den Befehl **Hilfe | Registrieren**

Auf Updates überprüfen

Überprüft, ob am Altova Server eine neuere Version Ihres Produkts vorhanden ist und zeigt eine entsprechende Meldung an.

18.15.6 Support Center, Fragen und Antworten, Komponenten downloaden**Support Center**

Falls Sie Fragen zu unseren Produkten haben, kontaktieren Sie über diesen Befehl unser Support-Team. Hier finden Sie Links zu unserer Liste von Fragen und Antworten, Supportformulare und die Email-Adressen, um unser Support-Team zu kontaktieren.

Fragen und Antworten im Web

Um Ihnen den bestmöglichen Support bieten zu können, haben wir eine Liste von Fragen und Antworten im Internet (FAQ) zusammengestellt. Diese Liste wird ständig von unseren Support-Mitarbeitern aktualisiert.

Sollten Sie Probleme mit einem unserer Produkte haben, raten wir Ihnen, zuerst die Fragen und Antworten im Internet zu konsultieren. Oft erhalten Sie hierdurch schnelle Hilfe.

Leider bieten wir derzeit noch keinen telefonischen Support. Ihre E-Mails an das Support-Team werden aber in der Regel innerhalb eines Werktages beantwortet.

Wenn Sie uns Anregungen oder Vorschläge zu XMLSpy unterbreiten möchten, verwenden Sie bitte unseren Fragebogen.

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. U.a. finden Sie hier den Altova Validator, XSLT- und XQuery-Prozessoren.

18.15.7 Internet**XMLSpy im Internet**

Mit diesem Befehl gelangen Sie zum Altova Webserver auf <http://www.altova.com>. Hier erhalten Sie alle Informationen, Produkt-Updates und weitere Angebote von Altova.

Training

Mit diesem Befehl gelangen Sie zur Online-Training-Seite der Altova Website. Hier können Sie sich für Online-Kurse zu unseren Altova-Produkten anmelden.

18.15.8 Über

Mit diesem Befehl wird der XMLSpy Splashscreen und das Dialogfeld mit urheberrechtlichen Informationen und dem XMLSpy-Logo angezeigt. Wenn Sie die 64-Bit-Version von XMLSpy verwenden, wird in der Statusleiste das Suffix (x64) hinter dem Applikationsnamen angezeigt. Für die 32-Bit-Version gibt es kein Suffix.

Bitte beachten Sie:

In diesem Dialogfeld wird die Versionsnummer angezeigt. Der Build sowie die komplette Versionsnummer der Version, die Sie verwenden, werden in der Statusleiste angezeigt.

18.16 Befehlszeile

Bestimmte XMLSpy-Aktionen können über die Befehlszeile ausgeführt werden. Diese Befehle sind im Folgenden aufgelistet:

Datei öffnen

Befehl: `xmlspy.exe file.xml`

Aktion: Öffnet die Datei `file.xml`, in XMLSpy

Anmerkung: Wenn einer XML-Datei bereits eine SPS-Datei zugewiesen wurde, so wird die XML-Datei in der Authentic-Ansicht geöffnet. Andernfalls wird die Datei in der Textansicht geöffnet. Wenn keine SPS-Datei zugewiesen wurde, kann diese mit Hilfe des Flag `/sps` (siehe unten) zugewiesen werden.

Mehrere Dateien öffnen

Befehl: `xmlspy.exe file1.xml file2.xml`

Aktion: Öffnet die Dateien `file1.xml` und `file2.xml` in XMLSpy

Einer XML-Datei eine SPS-Datei für die Bearbeitung in der Authentic-Ansicht zuweisen

Befehl: `xmlspy.exe myxml.xml /sps mysp.sps`

Aktion: Öffnet die Datei `myxml.xml` in der Authentic-Ansicht mit `mysp.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

Befehl: `xmlspy.exe mysp.sps`

Aktion: 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

Befehl: `xmlspy.exe /raw mysp.sps`

Aktion: Öffnet die Datei `mysp.sps` als XML-Dokument in der Textansicht. Das Flag `/raw` gibt an, dass die nachfolgende SPS-Datei als XML-Datei zu bearbeiten ist.

Altova XMLSpy 2012

Referenz für Programmierer

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](#)).

Ausführungsumgebungen

Die Applikations-API kann von den folgenden Umgebungen aus aufgerufen werden:

- [Skript-Editor](#)
- [IDE Plug-ins](#)
- [Externe Programme](#)
- [ActiveX Integration](#)

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](#) 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](#).

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](#).

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 Applikations-API bedienen. Eine Beschreibung

der Applikations-API finden Sie im Abschnitt [Applikations-API](#).

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 [Programmierersprachen](#).

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](#) 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#](#) 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](#): 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](#) 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 die gleichen, wie im [Abschnitt "Schnittstellen" des Abschnitts Applikations-API](#) 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](#).

Informationen über die Referenz für Programmierer

Die XMLSpy Referenz für Programmierer besteht aus den folgenden Abschnitten:

- [Skript-Editor](#): eine Benutzerreferenz für die Skripting-Umgebung in XMLSpy
- [IDE Plug-ins](#): eine Anleitung zum Erstellen von Plug-ins für XMLSpy
- [Applikations-API](#): eine Referenz für die Applikations-API
- [ActiveX Integration](#): Anleitung und Referenz zur Integration der Benutzeroberfläche und Funktionalitäten von XMLSpy mit Hilfe eines ActiveX Control.

1 Skripterstellung

Der Skript-Editor von XMLSpy verwendet die Formular-Editor-Komponenten von Microsoft .NET Framework und gestattet somit Zugriff auf Microsoft .NET Framework. Das bedeutet, dass JScripts und VBScripts nicht nur mit der XMLSpy API - einer COM API und der API von XMLSpy - funktionieren, sondern auch Zugriff auf Klassen von Microsoft .NET Framework haben und diese verwenden können.

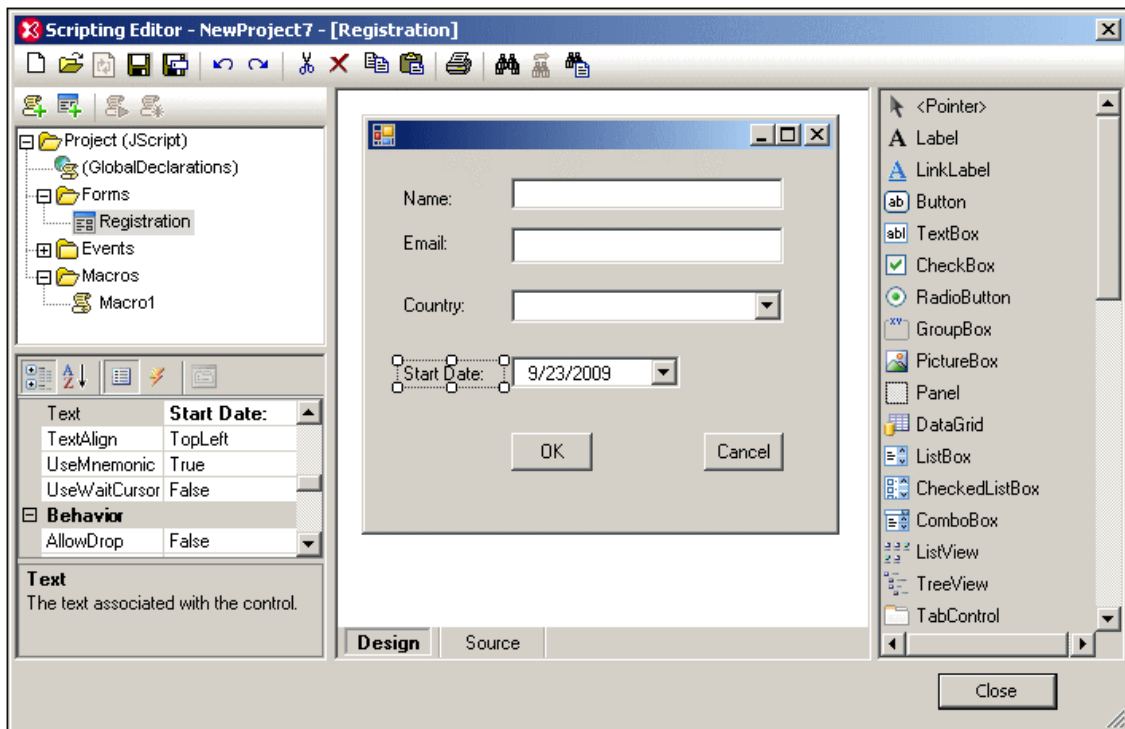
Sie können daher innerhalb von XMLSpy Ihre eigenen Makros und Formulare erstellen und verwenden und die Funktionalitäten Ihrer Installation von XMLSpy auf diese Art ergänzen und ändern.

Anmerkung: Visual Basic wird in der Skripting-Umgebung als Sprache **nicht unterstützt**. Unterstützt werden nur VBScript und JavaScript. Stellen Sie sicher, dass Sie in der Skripting-Umgebung die Syntax von VBScript und nicht die Syntax von Visual Basic verwenden.

Anmerkung: Voraussetzung für die Verwendung des Skript-Editors ist die vorherige Installation von Microsoft **.NET Framework 2.0 oder höher**.

Der Skript-Editor

Der Skript-Editor (*Abbildung unten*) wird in einem separaten Fenster geöffnet und über den Menübefehl **Extras | Skript-Editor** der XMLSpy Benutzeroberfläche aufgerufen. In der Skripting-Umgebung können die Programmiersprachen **JScript** und **VBScript** verwendet werden. Durch Rechtsklick auf den Projekteintrag im Projektfenster und Auswahl des Befehls **Skriptsprache** und anschließend der gewünschten Skriptsprache können Sie die Skriptsprache ändern.



Was können Sie mit dem Skript-Editor tun?

Sie können im Skript-Editor Formulare, EventHandler und Makros erstellen, um ein Skripting-Projekt anzulegen. Anschließend kann ein Skripting-Projekt als das globale Skripting-Projekt für XMLSpy definiert werden, sodass Skripts aus dem Skripting-Projekt in der Applikation verwendet werden können. Außerdem können verschiedenen XMLSpy-Projekten verschiedene Skripting-Projekte zugewiesen werden, sodass Sie für unterschiedliche Projekte unterschiedliche XMLSpy-Skripts verwendet können.

In jedem Skripting-Projekt kann die gewünschte .NET Runtime Version definiert werden. Eine Applikation kann mehrere Skripting-Projekte mit unterschiedlichen .NET Runtime Versionen gleichzeitig ausführen, doch muss die entsprechende .NET-Version installiert sein. So lassen sich z.B. Skripting-Projekte mit .NET 4.0 nur auf Rechnern ausführen, auf denen .NET 4.0 installiert ist.

Dokumentation zum Skript-Editor

Die Dokumentation zur Beschreibung der Skripting-Umgebung (dieser Abschnitt) ist in die folgenden Unterabschnitte gegliedert:

- [eine Übersicht](#), in der der Skript-Editor und Skripting-Projekt überblicksmäßig beschrieben werden.
- [eine Liste der Schritte, die zur Erstellung eines Skripting-Projekts erforderlich sind](#).
- [eine Erläuterung zu globalen Deklarationen](#) sowie ein Beispiel dazu.
- [eine Anleitung zur Erstellung von Formularen](#).
- [eine Erörterung von XMLSpy-spezifischen Event Handlern](#).
- [eine Anleitung zur Verwendung von Makros](#) im Skript-Editor und in XMLSpy.

1.1 Übersicht

Der Skript-Editor bietet eine Benutzeroberfläche über die Sie (i) Formulare grafisch erstellen können, wobei Sie Komponenten im Formular Formularskripts zuweisen; (ii) Event Handler und (iii) Makros erstellen können.

Diese Formulare, Event Handler und Makros sind in Skripting-Projekte gegliedert, die anschließend XMLSpy-Applikationsprojekten zugewiesen werden und in der Applikation verwendet werden können.

Variablen und Funktionen können in einem globalen Deklarations-Skript definiert werden, das immer vor der Ausführung von Makros oder Event Handler Skripts ausgeführt wird.

Dieser Abschnitt enthält eine Übersicht über den Skript-Editor und Skripting-Projekte. Er ist in die folgenden Abschnitte gegliedert:

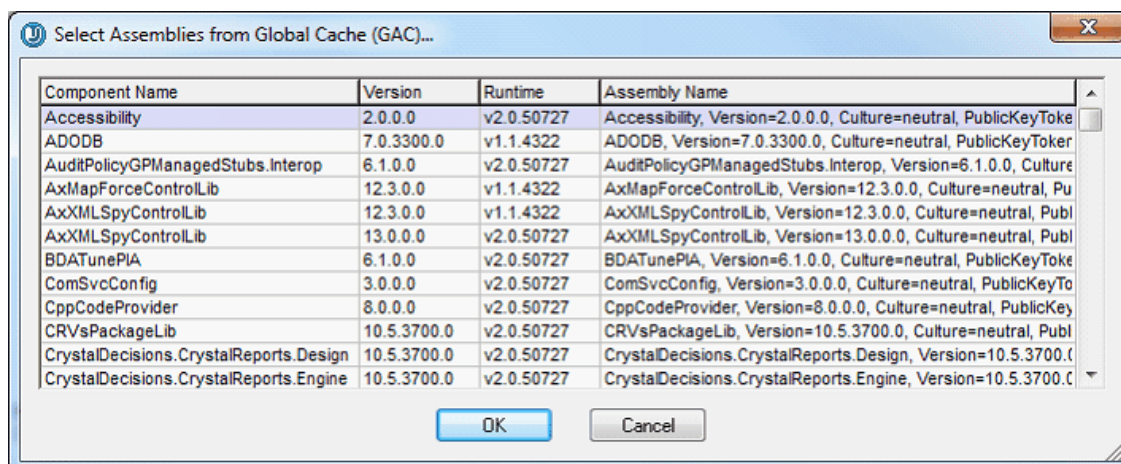
- [Skripting-Projekte in XMLSpy](#): Hier wird beschrieben, wie die von Ihnen erstellten Skripting-Projekte im Skript-Editor XMLSpy verwendet werden.
- [Die Benutzeroberfläche des Skript-Editors](#): Hier werden die einzelnen Bestandteile der Skript-Editor-Benutzeroberfläche und ihre Verwendung im Detail beschrieben.
- [Komponenten eines Skripting-Projekts](#): Hier werden die verschiedenen Komponenten beschrieben, aus denen ein Skripting-Projekt besteht.

Nähere Informationen zur Erstellung der verschiedenen Komponenten ([Globale Deklarationen](#), [Formulare](#), [Event Handler](#) und [Makros](#)) finden Sie in den entsprechenden Abschnitten.

.NET Assemblies

Jedes Skripting-Projekt kann - zusätzlich zu den Standardreferenzen - Referenzen auf .NET Assemblies enthalten. .NET Assemblies können für das gesamte Skripting-Projekt oder für einzelne Makros hinzugefügt werden (mit Hilfe des neuen Befehls `CLR.LoadAssembly` im Quellcode; siehe [Vordefinierte Befehle](#)). Assemblies können z.B. aus dem Global Assembly Cache hinzugefügt werden.

Um eine Assembly hinzuzufügen, klicken Sie mit der rechten Maustaste auf das Projekt oder Makro und wählen Sie im Kontextmenü den Befehl **.NET Assembly hinzufügen | Assembly aus Global Cache (GAC)**.



Dies funktioniert auf dieselbe Art wie bei Visual Studio und bietet nicht nur Zugriff auf das komplette Microsoft .NET Framework, sondern auch auf jede benutzerdefinierte Assembly.

1.1.1 Skripting-Projekte in XMLSpy

Alle im Skript-Editor erstellte Skripts und Skripting-Informationen werden in **Altova Skripting-Projekten** (. **asprj** Dateien) gespeichert.

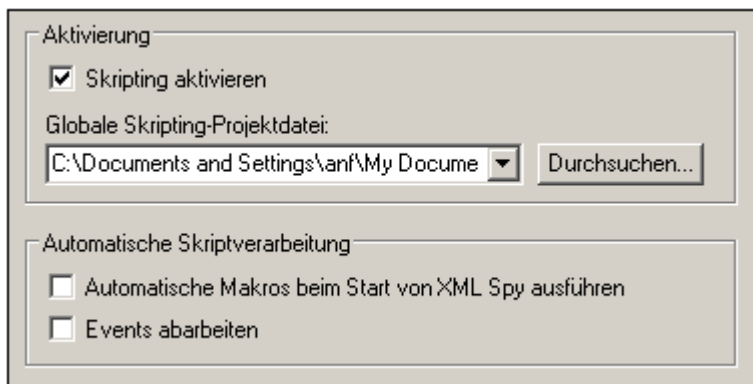
Sie können beliebig viele Altova Skripting-Projekte erstellen. Nachdem ein Skripting-Projekt erstellt wurde, kann es auf die folgenden Arten verwendet werden:

- Es kann als globales Skripting-Projekt für XMLSpy definiert werden. Skripts im globalen Skripting-Projekt können anschließend von der Applikation aus aufgerufen werden und Makros des globalen Skripting-Projekts können für alle XMLSpy Projekte verwendet werden.
- Es kann einem XMLSpy-Projekt (als ein Applikationsprojekt) zugewiesen werden. Wenn ein XMLSpy Projekt in XMLSpy geöffnet ist, können Skripts aus dem damit verknüpften Skripting-Projekt aufgerufen werden.

Ihr XMLSpy-Paket enthält ein Skripting-Beispielprojekt namens `SampleScripts.asprj`. Diese Datei befindet sich im Ordner: `C:\Documents and Settings\<username>\My Documents\Altova\XMLSpy2012\Examples\` und enthält globale Deklarationen für einige Standardaufgaben.

Definieren des globalen Skripting-Projekts einer Applikation

Das globale Skripting-Projekt einer Applikation wird im XMLSpy Dialogfeld "Optionen" auf dem Register "Skripting-Umgebung" (*Abbildung unten Extras | Optionen*) definiert.



Um ein globales Skripting-Projekt für XMLSpy zu definieren, aktivieren Sie das Kontrollkästchen *Skripting aktivieren* und navigieren Sie anschließend zum gewünschten Altova Skripting-Projekt (. *asprj* -Datei). Außerdem können Sie festlegen (i) ob automatische Makros im Skripting-Projekt automatisch ausgeführt werden sollen, wenn XMLSpy gestartet wird, und (ii) ob Applikations-Event Handler Skripts im Projekt automatisch ausgeführt werden sollen oder nicht; aktivieren bzw. deaktivieren Sie das entsprechende Kontrollkästchen.

Anmerkung: Die Ausführung von geschachtelten Skripts ist möglich, d.h. Makros können andere Makros aufrufen und Events werden während der Makro- oder Event-Ausführung empfangen.

Zuweisen eines Skripting-Projekts zu einem XMLSpy-Projekt

Um einem XMLSpy Projekt ein Skripting-Projekt zuzuweisen, gehen Sie folgendermaßen vor:

1. Öffnen Sie auf der XMLSpy Benutzeroberfläche das gewünschte Applikationsprojekt.
2. Wählen Sie den Menübefehl **Projekt | Skript-Einstellungen**.
3. Aktivieren Sie das Kontrollkästchen *Projektskripts aktivieren* und wählen Sie das

gewünschte Skripting-Projekt (. asprj Datei) aus. Wenn Sie möchten, dass automatische Makros ausgeführt werden, wenn das XMLSpy-Projekt geladen wird, aktivieren Sie das Kontrollkästchen *Automatische Makros ausführen*.

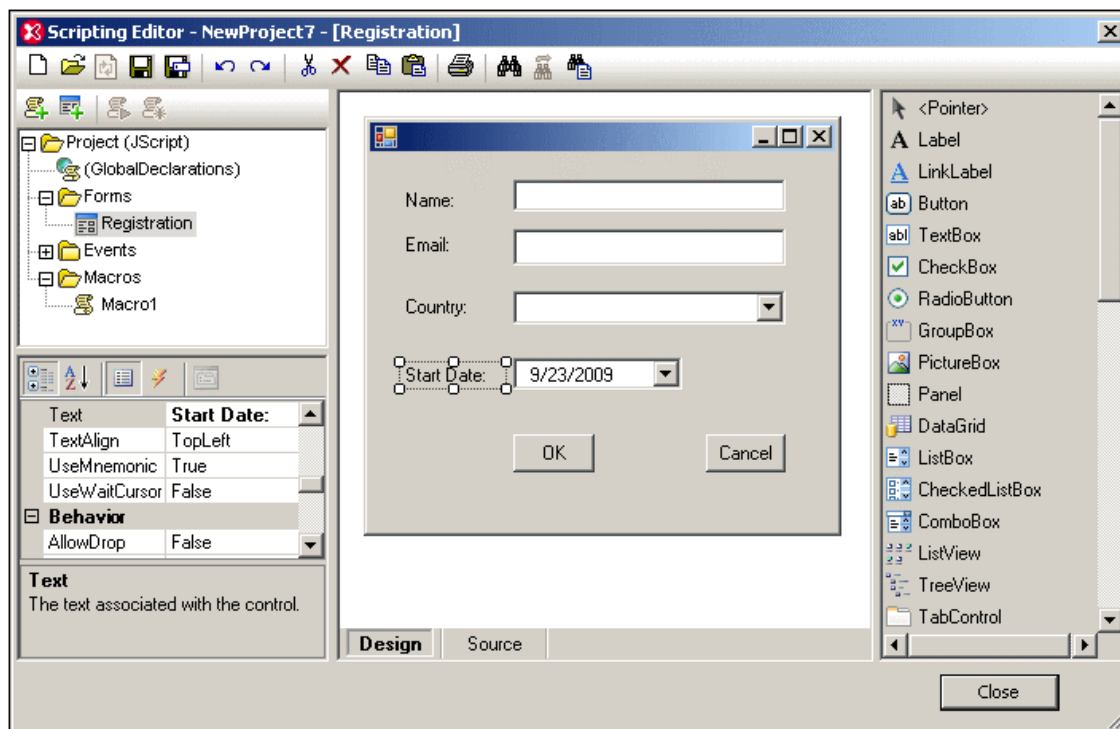
4. Klicken Sie zum Fertigstellen auf **OK**.

Anmerkung: Um das Skripting-Projekt eines XMLSpy-Projekts zu deaktivieren, also die Zuweisung aufzuheben, deaktivieren Sie das Kontrollkästchen *Projektskripts aktivieren*.

1.1.2 Die Benutzeroberfläche des Skript-Editors

Unten sehen die Benutzeroberfläche des Skript-Editors. Er besteht aus den folgenden Bestandteilen:

- einer [Symbolleiste](#)
- einem [Skripting-Projekt-Strukturfenster](#) (linke obere Seite)
- einem [Eigenschafts- und Events-Fenster](#) (links unten)
- einem [Hauptfenster](#) mit den Registern "Design" und "Quelle"
- einer [Formularobjektpalette](#) (rechte Seite)



Skript-Editor-Symbolleiste

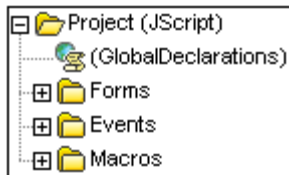
Die Symbolleiste des Skript-Editors enthält die folgenden Schaltflächen:

- Schaltflächen für Standarddateibefehle wie **Neu**, **Öffnen**, **Speichern** und **Drucken**. Mit Hilfe dieser Befehle werden neue Skripting-Projekte erstellen, bestehende geöffnet und Skripting-Projekte gespeichert und gedruckt.
- Schaltflächen für Standardbearbeitungsbefehle wie **Kopieren**, **Einfügen**, **Rückgängig**, **Wiederholen**, **Suchen** und **Ersetzen**. Beachten Sie, dass die Befehle **Suchen** und **Ersetzen** auf den Code auf dem Register "Quelle" des Skript-Editors angewendet

werden.

Skripting-Projektstruktur

In der Skripting-Projektstruktur (*Abbildung unten*) sehen Sie die verschiedenen Komponenten des Skripting-Projekts, die in vier Hauptzweige gegliedert sind: (i) Globale Deklarationen (GlobalDeclarations), (ii) Formulare (Forms), (iii) Events und (iv) Makros (Macros).



Über das Skripting-Projekt haben Sie Zugriff auf die einzelnen Komponenten des Skripting-Projekts. Um z.B. ein bestimmtes Formular anzuzeigen und zu bearbeiten, erweitern Sie den Ordner "Forms" in der Struktur (*siehe Abbildung oben*), klicken Sie mit der rechten Maustaste auf das gewünschte Formular und wählen Sie im Kontextmenü den Befehl **Open**.

Eine schnellere Methode, um ein Formular, Event oder das Skript für globale Deklarationen zu öffnen, ist, auf das entsprechende Symbol bzw. den Text doppelzuklicken. Um ein Formular oder Makro aus dem Skripting-Projekt zu löschen, klicken Sie mit der rechten Maustaste auf die Komponente und wählen Sie im Kontextmenü den Befehl **Delete**.

Das Fenster "Skripting-Projektstruktur" enthält eine Symbolleiste mit Schaltflächen (*Abbildung unten*).

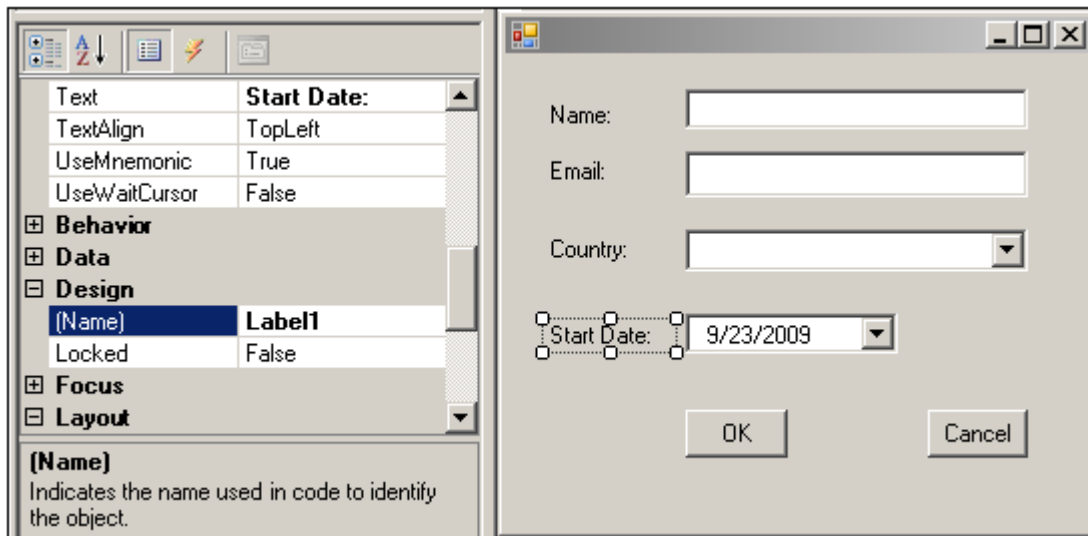


Die Schaltflächen dienen von links nach rechts beschrieben zum (i) [Erstellen eines neuen Makros](#), (ii) [Erstellen eines neuen Formulars](#), (iii) [Ausführen eines Makros](#) und (iv) [Debuggen eines Makros](#). Diese Befehle stehen auch über das Kontextmenü zur Verfügung, das angezeigt wird, wenn Sie mit der rechten Maustaste auf eine Komponente in der Skripting-Projektstruktur klicken.

Eigenschaften und Events

Im Eigenschafts- und Eventfenster (*Abbildung unten*) werden die folgenden Informationen angezeigt:

- Formulareigenschaften, wenn das Formular ausgewählt ist
- Objekteigenschaften, wenn ein Objekt in einem Formular ausgewählt ist. (In der Abbildung unten sehen Sie auf der linken Seite die Eigenschaften des Objekts, das im Formular auf der rechten Seite ausgewählt ist)
- Formular-Events, wenn ein Formular ausgewählt ist
- Objekt-Events, wenn ein Objekt in einem Formular ausgewählt ist



Um zwischen den Eigenschaften und Events der ausgewählten Komponente hin- und herzuwechseln, klicken Sie auf die Schaltfläche **Properties** (die dritte von links in der Symbolleiste "Properties und Events", siehe Abbildung oben) bzw. auf die Schaltfläche **Events** (vierte von links).

Die erste und die zweite Schaltfläche (von links) der Symbolleiste sind die Schaltflächen **Categorized** und **Alphabetical**. Sie dienen zum Anzeigen der Eigenschaften oder Events, entweder nach Kategorie oder in aufsteigender alphabetischer Reihenfolge geordnet.

Wird eine Eigenschaft oder ein Event ausgewählt, wird am unteren Rand des Eigenschafts- und Event-Fensters eine kurze Beschreibung angezeigt.

Hauptfenster

Im Hauptfenster wird immer nur eine Komponente angezeigt. Es hat ein oder zwei Register, je nachdem, was gerade darin angezeigt wird. Wenn ein globales Deklarationskript, ein Event oder ein Makro angezeigt wird, so sehen Sie ein einziges Register, das Register "Source", auf dem der Quellcode der ausgewählten Komponente angezeigt wird.

Das Register "Source" unterstützt:

- Syntaxfärbung
- Klappleiste
- Setzen/Löschen von Lesezeichen mit **Strg+F2**
- Autokomplettierungseingabehilfe mit Parameterinfo
- Goto Brace, Goto Brace Extend
- Zoom In / Zoom Out
- vollständige Methoden/Eigenschaftssignatur. Wird neben der Autokomplettierungseingabehilfe angezeigt
- Markierung von geschwungenen Klammern bei der Codeeingabe

```
if ( x == y.GetName( a, b, c() ) )
```
- Popup-Anzeige bei Mauspositionierung; bei Positionierung der Maus über eine bekannte Methode oder Eigenschaft wird ihre Signatur (und Dokumentation, falls vorhanden) angezeigt

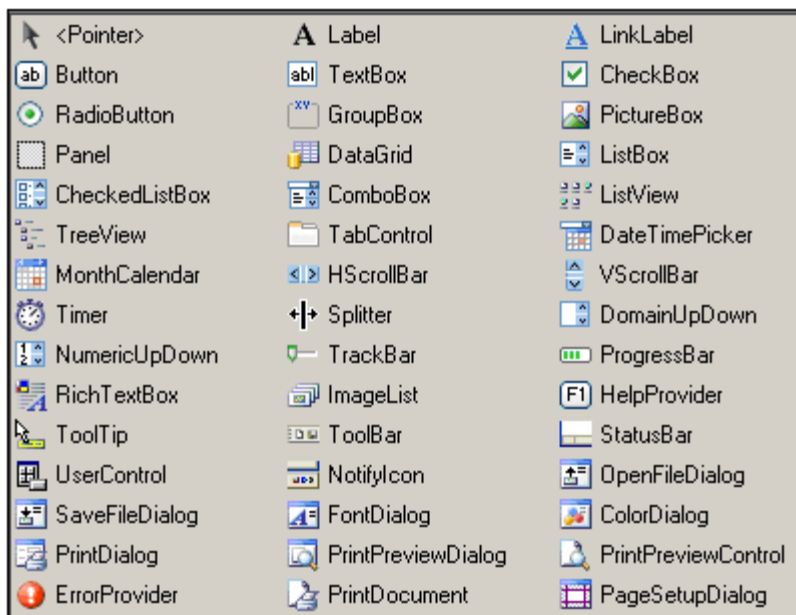
Wenn ein **Formular** angezeigt wird, sehen Sie im Hauptfenster zwei Register: ein Register "Design", in dem das Layout des Formulars angezeigt und aktiviert wird und ein Register "Source", in dem der Quellcode für das Formular angezeigt wird. Sowohl der Inhalt des Registers "Design" als auch der des Registers "Source" kann bearbeitet werden.

Anmerkung: Da JScript und VB Script Sprachen typenlose Sprachen sind, werden die Eingabehilfen und die Autokomplettierung nur im Fall von "vollständig qualifizierten Konstrukten" und "vordefinierten" Namen unterstützt.

Wenn Namen mit `objDocument`, `objProject`, `objXMLDat`, oder `objAuthenticRange` beginnen, werden Mitglieder der entsprechenden Schnittstelle angezeigt. Auch Autokomplettierungseingabehilfen und Parameterinformationen werden bei der Bearbeitung angezeigt, können aber auch jederzeit durch Drücken von Strg+Leerzeichen angezeigt werden.

Formularobjektpalette

Die Formularobjektpalette enthält alle Objekte, die beim Erstellen von Formularen zur Verfügung stehen, und sieht in etwa so aus, wie in der Abbildung unten. Registrierte ActiveX Controls können durch Rechtsklick in das Fenster und Auswahl des Befehls **Add ActiveX Control** zur Formularobjektpalette hinzugefügt werden.



Um ein Objekt aus der Formularobjektpalette einzufügen, klicken Sie in der Palette auf das gewünschte Objekt und anschließend an die Stelle im Formular, an der das Objekt eingefügt werden soll. Daraufhin wird das Objekt an dieser Stelle eingefügt. In vielen Fällen müssen Sie über das Eigenschafts- und Events-Fenster mehr Eigenschaften des Objekts definieren. Sie können das Objekt an eine andere Stelle verschieben und seine Größe ändern. Außerdem können über das Kontextmenü des ausgewählten Formularobjekts eine Reihe von Bearbeitungsbefehlen wie Zentrieren und Stapeln von Objekten aufgerufen werden.

Einige Formularobjekte wie z.B. `Timer` werden nicht zum Formular hinzugefügt, sondern als Bereichskomponenten in einem Bereich am unteren Rand des Hauptfensters erstellt. Sie können das Objekt im Bereich auswählen und die Eigenschaften und Event Handler für das Objekt über das Eigenschafts- und Event-Fenster definieren. Ein Beispiel dafür, wie eine Bereichskomponente behandelt wird, finden Sie unter [Verwendung von Formularen und](#)

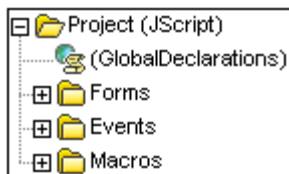
[Befehlen.](#)

1.1.3 Komponenten eines Skripting-Projekts

Ein Altova Skripting-Projekt besteht aus den folgenden vier Hauptbestandteilen:

- *Globalen Deklarationen*, einer Komponente, die die Definitionen von Variablen und Funktionen enthält, die allen Formularen, Makros und Event Handler Skripts im Skripting-Projekt zur Verfügung stehen und von ihnen verwendet werden können.
- *Formularen*, einer Komponente, die alle im Skripting-Projekt definierten Formulare enthält.
- *Events*, einer Komponente, die Event Handler Skripts für alle applikationsbasierten - im Gegensatz zu formularbasierten - Events enthält.
- *Makros*, eine Komponente, die alle im Skripting-Projekt definierten Makros enthält.

Diese Komponenten werden in der Skripting-Projektstruktur des Skript-Editors angezeigt und können hier aufgerufen werden (*Abbildung unten*).



Im Folgenden sehen Sie eine kurze Beschreibung der einzelnen Komponenten.

Globale Deklarationen

Die Komponente "Global Declarations" (Globale Deklarationen) ist ein Skript, das Variablen und Funktionen enthält, die von Formularen, Event Handlern und Makros verwendet werden können. Die Funktionen nutzen zum Aufrufen der XMLSpy Funktionalität die XMLSpy API. Wenn Sie im Modul "Global Declarations" eine Variable oder Funktion erstellen, können Sie diese von allen Formularen, Event Handlern und Makros im Skripting-Projekt aus aufrufen.

Um eine Variable oder Funktion hinzuzufügen, öffnen Sie die Komponente "Global Declarations" (durch Rechtsklick in die Skripting-Projektstruktur und Auswahl von **Öffnen**) und bearbeiten Sie das Global Declarations-Skript im Hauptfenster. Fügen Sie in diesem Skript die gewünschte Variable oder Funktion hinzu.

Formulare

Sie können im Skript-Editor grafisch mit Hilfe einer Palette von Formularobjekten wie Texteingabefeldern und Schaltflächen ein Formular erstellen. So können Sie z.B. ein Formular erstellen, um die Eingabe eines Elementnamens zu akzeptieren und dann alle Instanzen dieses Elements aus dem aktiven XML-Dokument entfernen.

Bei einem solchen Formular kann ein Funktionsskript mit einem Textfeld verknüpft werden, damit dieses eine Eingabevariable akzeptiert, und ein Event Handler kann mit einer Schaltfläche verknüpft werden, um die Ausführung der Löschfunktion zu starten, die in der XMLSpy API zur Verfügung steht. Ein Formular wird entweder durch Aufruf dieses Formulars innerhalb einer Funktion (im globalen Deklarationsskript) oder direkt über ein Makro aufgerufen. Nähere Informationen zum Erstellen und Bearbeiten von Formularen finden Sie im Abschnitt [Formulare](#).

Behandlung von Events

Event Handler Skripts können mit den verschiedensten verfügbaren Events verknüpft werden.

Sie können sowohl Events steuern, die innerhalb von Formularen ([Formular-Events](#)) vorkommen als auch Events, die innerhalb der allgemeinen Applikationsschnittstelle ([Applikationsevents](#)) vorkommen. Das mit einem Event verknüpfte Skript wird sofort beim Auslösen des Event ausgeführt.

Die meisten Events haben Parameter, die ausführliche Informationen zum Event enthalten. Der Rückgabewert aus dem Skript weist die Applikation normalerweise an, wie sie mit der Verarbeitung fortfahren soll (so gestattet die Applikation z.B. die Bearbeitung nicht).

Ein Event Handler wird ausgeführt, wenn das entsprechende Event im Formular oder in XMLSpy vorkommt. Nähere Informationen zum Erstellen von Event Handlern finden Sie unter [Event Handler](#).

Makros

Makros dienen zum Implementieren komplexer oder sich wiederholender Aufgaben. Makros verwenden weder Parameter noch Rückgabewerte.

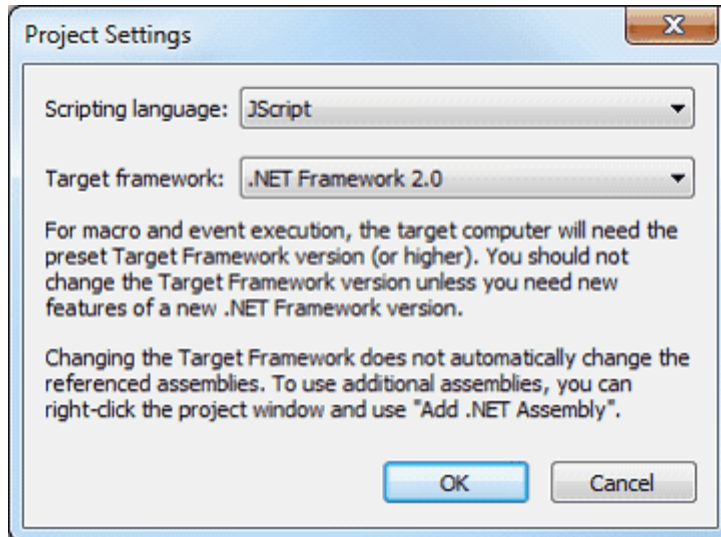
In einem Makro können alle in den globalen Deklarationen definierten Variablen und Funktion aufgerufen werden und es könnten Formulare für die Benutzereingabe angezeigt werden.

Ein einfaches Beispiel zur Erstellung eines Makros finden Sie unter [Schreiben eines Makros](#). Eine Beschreibung zu den Aufrufmethoden eines Makros finden Sie unter [Ausführen von Makros](#). Ein Makro wird auf der XMLSpy Benutzeroberfläche durch Auswahl von **Extras | Makros | [Makroname]** ausgeführt.

1.2 Erstellen eines Skripting-Projekts

Der allgemeine Ablauf bei der Erstellung eines Skripting-Projekts ist der folgende:

1. Öffnen Sie den Skript-Editor durch Auswahl des Befehls **Extras | Skript-Editor**.
2. Öffnen Sie im Skript-Editor ein neues Skripting-Projekt. Klicken Sie dazu in der Symbolleiste des Skript-Editors auf die Schaltfläche **Neu**. Daraufhin wird das Dialogfeld "Projekteinstellungen" angezeigt (*Abbildung unten*).



Wählen Sie in der ersten Auswahlliste entweder JScript oder VBScript und in der zweiten Auswahlliste das .NET Framework aus und klicken Sie auf **OK**. Daraufhin wird das neue Skripting-Projekt erstellt.

3. Klicken Sie in der Symbolleiste des Skript-Editors auf die Schaltfläche **Speichern**, um das Skripting-Projekt als `.aspxj`-Datei zu speichern.
4. Ein Skripting-Projekt besteht aus mehreren Komponenten, die zusammenarbeiten. Dabei handelt es sich normalerweise um eine Kombination aus globalen Deklarationen, Formularen, Events, und Makros. Sie können in jeder beliebigen Reihenfolge erstellt werden, doch sollten Sie genau verstehen, wie diese zusammenarbeiten. [Weiter unten](#) wird beschrieben, wie die einzelnen Komponententypen aufgerufen und ausgeführt werden. Eine Beschreibung zur Erstellung der einzelnen Komponententypen finden Sie in den entsprechenden Abschnitten zum jeweiligen Komponententyp.
5. Nachdem Sie mit der Erstellung aller gewünschten Komponenten fertig sind, speichern Sie das Skripting-Projekt (durch Klicken auf die Skript-Editor-Symbolleistenschaltfläche **Speichern**).
6. Schließen Sie den Skript-Editor.

Bitte beachten Sie:

Wenn Sie mit der rechten Maustaste auf den Projektordner klicken und "Skripting-Sprache..." wählen, können Sie die Skripting-Sprache jederzeit wechseln.

Wie Formulare, Event Handler und Makros aufgerufen und ausgeführt werden

Formulare, Event Handler und Makros werden alle im Skript-Editor erstellt. Die Art des Aufrufs und der Ausführung ist jedoch bei jeder einzelnen Komponente anders und hat einen Einfluss darauf, wie Sie Ihre Skripting-Projekte erstellen.

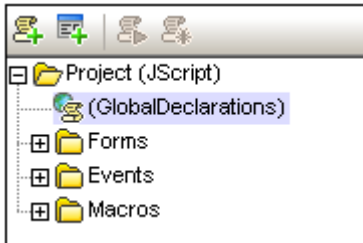
- Ein Formular wird entweder von einer Funktion im globalen Deklarationsskript aus oder direkt von einem Makro aufgerufen.
- Ein Event Handler wird ausgeführt, wenn das entsprechende Event in XMLSpy

vorkommt. Wenn ein Event Handler für ein einzelnes Event sowohl im globalen Skripting-Projekt als auch im XMLSpy-projektspezifischen Skripting-Projekt definiert ist, so wird zuerst der Event Handler für das projektspezifische Skripting-Projekt ausgeführt und unmittelbar danach der für das globale Skripting-Projekt.

- Ein Makro wird von der XMLSpy Benutzeroberfläche aus ausgeführt. Klicken Sie dazu auf **Extras | Makros | [Makroname]**. In einem Makro können alle in den globalen Deklarationen deklarierten Variablen und Funktionen aufgerufen werden und Formulare können darin für die Benutzereingabe angezeigt werden.

1.3 Globale Deklarationen

Die Komponente "Global Declarations" ist standardmäßig in jedem Skripting-Projekt vorhanden (siehe *Abbildung unten*) und muss daher nicht erstellt werden. Um Variablen und Funktionen zum Global Declarations-Skript eines Skripting-Projekts hinzuzufügen, müssen Sie das Global Declarations-Skript öffnen und das Codefragment zu diesem Skript hinzufügen. Nähere Informationen hierzu finden Sie unter [Komponenten eines Skripting-Projekts](#) und [Erstellen eines Skripting-Projekts](#).



Klicken Sie zum Öffnen des Global Declarations-Skripts mit der rechten Maustaste in der Skripting-Projektstruktur auf den Eintrag *Global Declarations* (*Abbildung oben*) und wählen Sie den Befehl **Öffnen**. Daraufhin wird das Global Declarations-Skript im Hauptfenster geöffnet.

Anmerkung: Bei jeder Ausführung eines Makros bzw. bei jedem Aufruf eines Event Handlers werden die globalen Deklarationen neu initialisiert.

Unten sehen Sie eine Beispielfunktion. Denken Sie daran: Wenn Sie eine Variable oder Funktion im Global Declarations-Skript erstellen, so kann diese Variable oder Funktion von allen Formularen, Event Handlern und Makros aus aufgerufen werden.

Beispielfunktion

Eine Funktion namens `RemoveAllNamespaces` hätte den folgenden Code:

```
function RemoveAllNamespaces(objXMLData)
{
    if(objXMLData == null)
        return;

    if(objXMLData.HasChildren) {
        var objChild;

        // spyXMLDataElement := 4
        objChild = objXMLData.GetFirstChild(4);

        while(objChild) {
            RemoveAllNamespaces(objChild);

            try{
                var nPos, txtName;
                txtName = objChild.Name;

                if((nPos = txtName.indexOf(":")) >= 0) {
                    objChild.Name = txtName.substring(nPos+1);
                }
            }

            objChild = objXMLData.GetNextChild();
        }
        catch(Err) {
            objChild = null;
        }
    }
}
```



```
    }  
  }  
}
```

Anmerkung:

- Es besteht die Möglichkeit lokale Variablen und Hilfsfunktionen innerhalb von Makros und Event Handlern zu definieren. Beispiel:

```
//return value: true allows editing  
//return value: false disallows editing  
var txtLocal;  
function Helper()  
{  
  txtMessage = txtLocal;  
  Application.ShowForm( "MsgBox" );  
}  
function On_BeforeStartEditing( objXMLData )  
{  
  txtLocal = "On_BeforeStartEditing()";  
  Helper();  
}
```

- Rekursive Funktionen werden unterstützt.

1.4 Formulare

Die Erstellung und Bearbeitung von Formularen im Skript-Editor erfolgt folgendermaßen:

1. [Anlegen eines neuen Formulars](#). Das neue Formular wird erstellt und erhält einen Namen. Es sind Eigenschaften dafür definiert.
2. [Erstellung des Formulars](#). Das Formular wird entworfen, indem Formularobjekte zum Formular hinzugefügt und den verschiedenen Formularobjekten Werte zugewiesen werden.
3. [Skripterstellung für Formular-Events](#). Den Formular-Events werden Skripts zugewiesen.

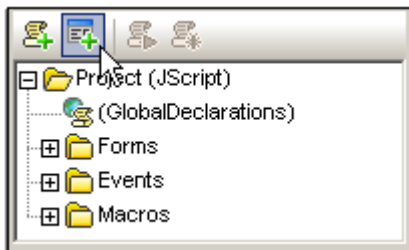
1.4.1 Erstellung eines neuen Formulars

Die Erstellung eines neuen Formulars im Skript-Editor erfolgt in den folgenden Schritten:

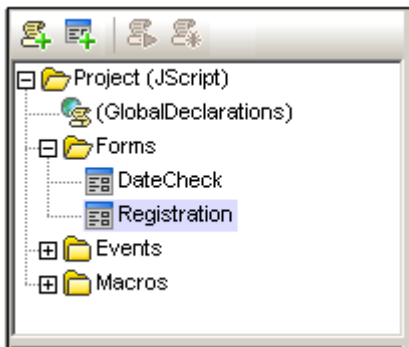
1. [Erstellung eines neuen Formulars und Benennung des Formulars](#)
2. [Definition der Formulareigenschaften](#)

Erstellung eines neuen Formulars und Benennung des Formulars

Um ein neues Formular zu einem Skripting-Projekt hinzuzufügen, klicken Sie in der Symbolleiste des Projektübersichtsfensters auf die Schaltfläche **Formular hinzufügen** (in der Abbildung unten markiert). Geben Sie den Namen des Formulars ein.



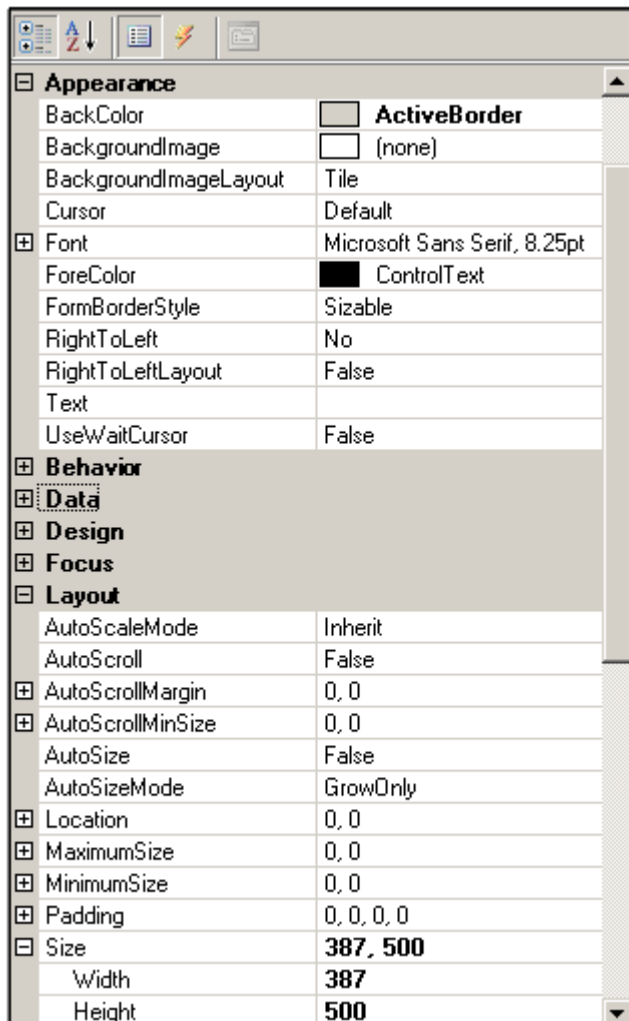
Ein neues Formular wird zum Projekt hinzugefügt und im Hauptfenster angezeigt. Im Skripting-Projektstrukturfenster wird dafür unterhalb der Überschrift "Form" ein Eintrag erstellt. Drücken Sie die Taste F2, um das Formular umzubenennen oder klicken Sie mit der rechten Maustaste auf den Formularnamen und wählen Sie im Kontextmenü den Befehl "Umbenennen" aus. In der Abbildung unten haben wir dem neuen Formular den Namen *Registration* gegeben.



Formulareigenschaften

Die Eigenschaften des Formulars wie z.B. seine Größe, Hintergrundfarbe und die Schriftarteneigenschaften können im Eigenschaftsfenster definiert werden. In der Abbildung unten

sehen Sie, dass die Eigenschaftswerte für "size" und "background-color" in den Kategorien *Layout* und *Appearance* fett markiert erscheinen.



Testen eines Formulars

Sie können ein Formular im Skript-Editor testen, indem Sie mit der rechten Maustaste in das Projektübersichtsfenster klicken und den Befehl **Formular testen** auswählen.

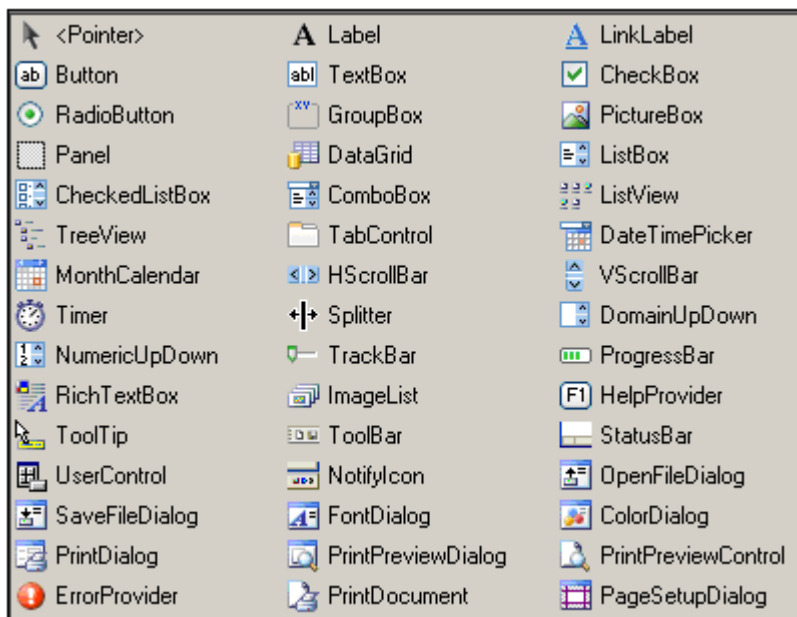
1.4.2 Formulardesign und Formularobjekte

Gehen Sie beim Entwerfen eines Formulars folgendermaßen vor:

- Platzieren Sie ein Objekt aus der [Formularobjektpalette](#) in das Formulardesign.
- Weisen Sie dem Objekt Werte für die [Eigenschaften einzelner Formularobjekte zu](#).
- [Weisen Sie Skripts für formularbasierte Events zu](#).

Die Formularobjektpalette

Die Formularobjektpalette enthält alle Objekte, die beim Erstellen von Formularen zur Verfügung stehen, und sieht in etwa so aus, wie in der Abbildung unten. Registrierte ActiveX Controls können durch Rechtsklick in das Fenster und Auswahl des Befehls **Add ActiveX Control** zur Formularobjektpalette hinzugefügt werden.



Um ein Objekt aus der Formularobjektpalette einzufügen, klicken Sie in der Palette auf das gewünschte Objekt und anschließend an die Stelle im Formular, an der das Objekt eingefügt werden soll. Daraufhin wird das Objekt an dieser Stelle eingefügt. In vielen Fällen müssen Sie über das Eigenschafts- und Events-Fenster mehr Eigenschaften des Objekts definieren. Sie können das Objekt an eine andere Stelle verschieben und seine Größe ändern. Außerdem können über das Kontextmenü des ausgewählten Formularobjekts eine Reihe von Bearbeitungsbefehlen wie Zentrieren und Stapeln von Objekten aufgerufen werden.

Einige Formularobjekte wie z.B. `Timer` werden nicht zum Formular hinzugefügt, sondern als Bereichskomponenten in einem Bereich am unteren Rand des Hauptfensters erstellt. Sie können das Objekt im Bereich auswählen und die Eigenschaften und Event Handler für das Objekt über das Eigenschafts- und Event-Fenster definieren. Ein Beispiel dafür, wie eine Bereichskomponente behandelt wird, finden Sie unter [Verwendung von Formularen und Befehlen](#).

Im Folgenden sind einige der am häufigsten verwendeten Objekte beschrieben:



Beschriftung: Fügt Textfelder wie Überschriften oder Feldbeschreibungen hinzu.



Schaltfläche: Fügt eine Schaltfläche hinzu. Auch Bitmaps können diesen Schaltflächen als Hintergrundbild zugewiesen werden.



Kontrollkästchen: Fügt ein Kontrollkästchen hinzu, über das Auswahlmöglichkeiten vom Typ Ja/Nein erfolgen können.



Auswahlliste: Fügt eine Auswahlliste hinzu, über die der Benutzer eine Option aus einem Dropdown-Menü auswählen kann.



Listenfeld: Fügt ein Listenfeld hinzu, das eine Liste von Einträgen zur Auswahl anzeigt.



Textfeld: Dient zur Eingabe einer einzigen Textzeile.



Mehrzeiliges Bearbeitungsfeld: Dient zur Eingabe mehrzeiliger Texte.

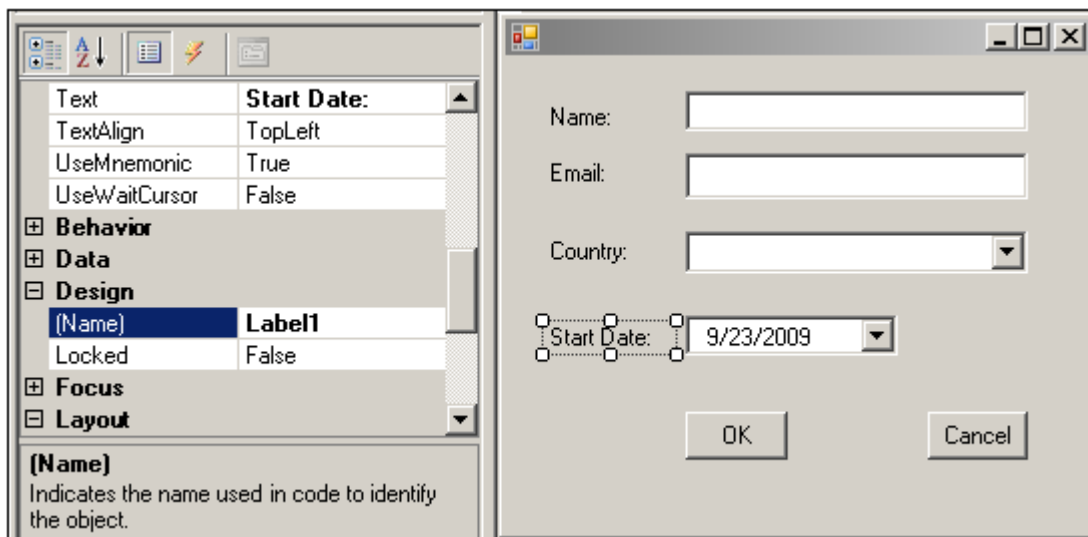
Erstellen von Objekten und Definieren ihrer Eigenschaften

Um im Formular ein Objekt zu erstellen, wählen Sie das gewünschte Objekt in der Formularobjektpalette zuerst aus und klicken Sie anschließend an die Stelle im Formular, an der es eingefügt werden soll. Nachdem das Objekt eingefügt wurde, können Sie seine Größe anpassen und es an eine andere Stelle im Formular ziehen.

Wenn ein Objekt im Design ausgewählt ist, können Sie seine Eigenschaften im Eigenschafts- und Event-Fenster definieren. Klicken Sie in der Symbolleiste des Eigenschafts- und Event-Fensters auf das Eigenschaftssymbol, um eine Liste der Eigenschaften des Objekts anzuzeigen.

So wurde z.B. in der Abbildung unten das Beschriftungsobjekt mit dem Text *Start Date* im Design ausgewählt. Im Eigenschafts- und Event-Fenster wird der Name des Objekts (welcher der Name ist, der verwendet wird, um das Objekt im Code zu identifizieren - in der Abbildung unten *Label1*) in der Eigenschaftskategorie *Design* angegeben; in diesem Fall ist der Name des Objekts *Label1*.

Der Beschriftungstext (also der Text, der im Formular angezeigt wird) muss als Wert der Eigenschaft *Text* in der Eigenschaftskategorie *Appearance* eingegeben werden.



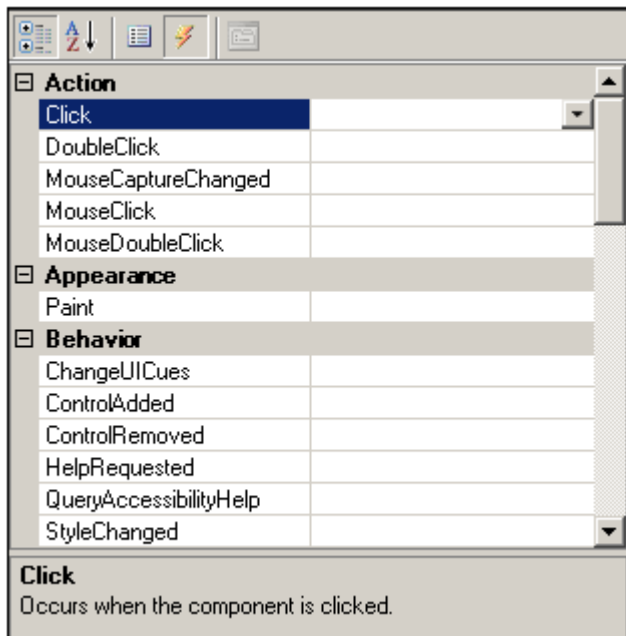
Um andere Objekteigenschaften zuzuweisen, geben Sie dafür Werte in das Eigenschafts- und Event-Fenster ein.

Testen eines Formulars

Sie können ein Formular im Skript-Editor testen, indem Sie mit der rechten Maustaste in das Projektübersichtsfenster klicken und den Befehl **Formular testen** auswählen.

1.4.3 Formular-Events

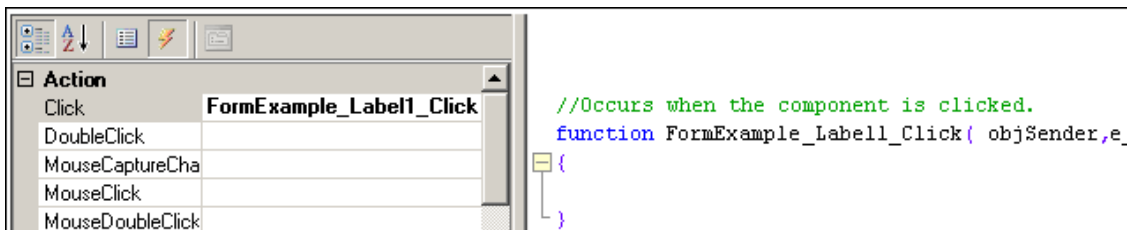
Wenn ein Objekt im Design ausgewählt ist, werden beim Klicken auf die Schaltfläche "Events" in der Symbolleiste des Eigenschafts- und Events-Fensters (*vierte Schaltfläche von links*) alle für dieses Objekt verfügbaren Events angezeigt (*siehe Abbildung unten*). Diese können entweder nach Kategorie (*Abbildung unten*) oder alphabetisch angezeigt werden.



Sie können für jedes Event den Namen eines bestehenden Event Handlers oder einer Funktion eingeben. Alternativ dazu können Sie:

- durch Doppelklick auf ein Event (i) auf dem Register *Source* des Hauptfensters ein leeres Funktionsskript erstellen und (ii) eine Verknüpfung der neu erstellten Funktion mit dem ausgewählten Event herstellen.
- durch Doppelklick auf eine Schaltfläche im Register "Design" den Handler Stub direkt im Codefenster generieren.

Der Screenshot unten wurde aufgenommen, nachdem auf das *Click* Event doppelgeklickt wurde. Beachten Sie, dass im Hauptfenster eine leere Event Handler-Funktion namens `FormExample_Label1_Click` erstellt wurde und dass diese Funktion im Eigenschafts- und Event-Fenster mit dem *Click* Event verknüpft wurde.



Geben Sie den gewünschten Skripting-Code ein und speichern Sie das Projekt.

Schreiben des erforderlichen Skripts

Nachdem das visuelle Design des Formulars abgeschlossen ist, werden Formularobjekte normalerweise mit passenden Skripts verknüpft. Im Beispiel unten wird ein Skript gezeigt, das Farben hinzufügt, wenn man auf eine Schaltfläche klickt. Das Script wird als Event Handler für das *Click* Event der Schaltfläche `Button1` eingefügt (das Event steht im Eigenschafts- und Event-Fenster zur Verfügung, wenn die Schaltfläche im Design ausgewählt wird):

```
function FormExample_Button1_Click( objSender, e_EventArgs )
{
    // Sets the ForeColor (red) of the button.
    objSender.ForeColor = CLR.Static( "System.Drawing.Color" ).Red;
}
```

```
        // Sets the BackColor (blue) of the button.  
        objSender.BackColor = CLR.Static( "System.Drawing.Color" ).Blue;  
        // Sets the form BackColor (green).  
        objSender.FindForm().BackColor = CLR.Static( "System.Drawing.Color"  
    ).Green;  
    }
```

1.5 Events

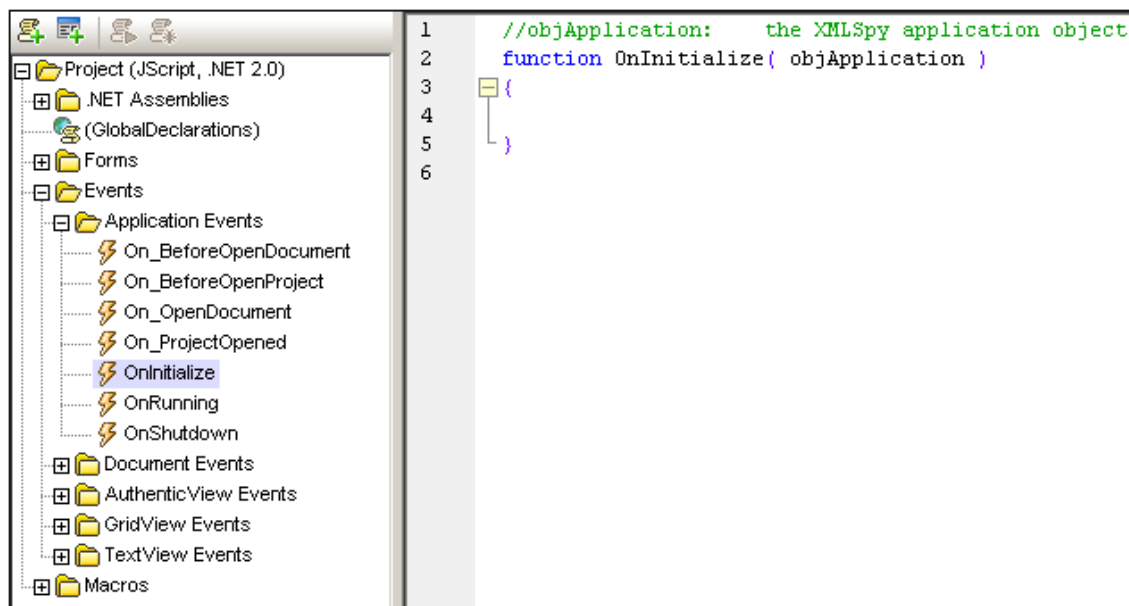
Der Ordner "Events" des Skripting-Projekts (*siehe Abbildung unten*) enthält Ordner für die folgenden Event-Arten:

- Application Events
- Document Events
- Authentic View Events
- Grid View Events
- Text View Events

Beachten Sie, dass diese Events im Gegensatz zu formularbasierten Events XMLSpy-spezifisch sind. Jeder der oben aufgelisteten Ordner enthält eine Gruppe von Events, für die Event Handler Skripts geschrieben werden können.

So werden z.B. in der Abbildung unten Application Events gezeigt.

Um Zugriff auf das Event Handler Skript eines dieser Events zu erhalten, klicken Sie mit der rechten Maustaste auf das Event und wählen Sie im Kontextmenü den Befehl **Öffnen**. Das Skript wird im Hauptfenster angezeigt (*siehe Abbildung unten*) und kann dort bearbeitet werden. Nachdem Sie mit der Bearbeitung fertig sind, speichern Sie die Änderungen durch Klicken auf die Schaltfläche **Speichern** in der Symbolleiste des Skript-Editors.



Beachten Sie die folgenden Punkte:

- Bei Event Handlern müssen die Funktionsüberschriften den exakt richtigen Event-Namen haben, da der Event Handler sonst nicht aufgerufen wird.
- Sie können in Makros und Event Handlern lokale Variablen und Hilfsfunktionen definieren. Beispiel:

```

//return value: true allows editing
//return value: false disallows editing
var txtLocal;
function Helper()
{
    txtMessage = txtLocal;
    Application.ShowForm( "MsgBox" );
}

```



```
}  
function On_BeforeStartEditing( objXMLData)  
{  
    txtLocal = "On_BeforeStartEditing()";  
    Helper();  
}
```

- Damit Events verarbeitet werden, müssen in den Skripting-Optionen von XMLSpy die "Events abarbeiten" Optionen aktiviert sein. Nähere Informationen dazu finden Sie unter [Skripting-Projekte in XMLSpy](#).
- Siehe auch [Anmerkungen zur Programmierung](#).

Application Events

OnInitialize

Das `OnInitialize` Event wird ausgelöst, nachdem das Hauptfenster angezeigt wird und noch bevor ein Projekt geladen wird. Dieses Event wird nicht ausgelöst, wenn die Applikation nicht geladen werden kann.

OnRunning

Wenn die Applikation zur Gänze geladen ist und nachdem das `OnInitialize` Event ausgelöst wurde, wird das `OnRunning` Event ausgelöst.

OnShutdown

Dieses Event wird ausgelöst, nachdem alle offenen Projekte und alle Dokumente beim Beenden der Applikation geschlossen wurden. Das Hauptfenster wird nicht mehr angezeigt.

Beispiel

Das folgende Skript ist ein Event Handler für das `On_BeforeOpenProject` Event. Sie können damit ein Skript hinzufügen, das jedes Mal, bevor XMLSpy ein Projekt öffnet, ausgeführt wird. Das Beispielskript unten öffnet der Reihe nach alle XML-Dateien im Ordner "XML" des Projekts und validiert diese. Wenn die Validierung fehlschlägt, zeigt das Skript einen Validierungsfehler an und bricht den Vorgang ab. Wenn eine Datei gültig ist, wird sie geschlossen und die nächste Datei wird geöffnet.

Geben Sie für das `On_BeforeOpenProject()` Event das folgende Skript ein und speichern Sie das Skripting-Projekt.

```
function On_BeforeOpenProject()  
{  
    var bOK;  
    var nIndex, nCount;  
    var objItems, objXMLFolder = null;  
  
    objItems = Application.CurrentProject.RootItems;  
    nCount = objItems.Count;  
  
    // search for XML folder  
    for( nIndex = 1; nIndex <= nCount; nIndex++) {  
        var txtExtensions;  
        txtExtensions = objItems.Item( nIndex ).FileExtensions;  
  
        if( txtExtensions.indexOf( "xml" ) >= 0 )    {  
            objXMLFolder = objItems.Item( nIndex );  
            break;  
        }  
    }  
  
    // does XML folder exist?  
    if( objXMLFolder ) {
```

```
var objChild, objDoc;

nCount = objXMLFolder.ChildItems.Count;

// step through associated xml files
for(nIndex = 1; nIndex <= nCount; nIndex++) {
    objChild = objXMLFolder.ChildItems.Item(nIndex);

    try{
        objDoc = objChild.Open();

        // use JScript method to access out-parameters
        var strError = new Array(1);
        var nErrorPos = new Array(1);
        var objBadData = new Array(1);

        bOK = objDoc.IsValid(strError, nErrorPos, objBadData);

        if(! bOK) {
            // if the validation fails, we should display the
            // message from XMLSpy
            // of course we have to create the form "MsgBox" and
            // define the global txtMessage variable
            //
            // txtMessage = Position:" + nErrorPos[0] + "\n" + // strError[0];
            // txtMessage += "\n\nXML:\n" + objBadData[0].Name + ", " +
            //     objBadData[0].TextValue;
            //
            // Application.ShowForm("MsgBox");

            break;
        }

        objDoc.Close(true);
        objDoc = null;
    }
    catch(Err) {
        // displaying the error description here is a good idea

        // txtMessage = Err.Description;
        // Application.ShowForm("MsgBox");

        break;
    }
}
```

Testen des Event Handlers

Wechseln Sie in XMLSpy und öffnen Sie ein Projekt um zu sehen, wie das `BeforeOpenProject` Event behandelt wird.

1.6 Makros

Makros können sich wiederholende oder komplizierte Aufgaben automatisieren. In der Skripting-Umgebung können Sie ein Skript erstellen, das Applikationsfunktionen sowie eigene benutzerdefinierte Funktionen aufruft. Dadurch steht Ihnen eine gute Methode zum Automatisieren von Aufgaben in XMLSpy zur Verfügung. Dieser Abschnitt über Makros ist in die folgenden Unterabschnitte gegliedert:

- [Erstellen und Bearbeiten eines Makros](#): Hier wird beschrieben, wie man ein neues Makro erstellt und ein bestehendes bearbeitet.
- [Ausführen eines Makros](#): Hier wird erklärt, wie ein Makro von Skript-Editor und auch von der XMLSpy-Umgebung ausgeführt werden kann.
- [Debuggen](#): Hier wird beschrieben, wie Makros debugged werden können.

Wichtige Punkte zu Makros

Im Folgenden finden Sie eine Zusammenfassung der wichtigsten Punkte zu Makros.

- Sie können beliebig viele Makros zum aktiven Skripting-Projekt hinzufügen. Diese Makros werden in der Altova Skripting-Projektdatei (.asprj-Datei) gespeichert.
- In einem Makro verwendete Funktionen können als globale Deklaration gespeichert werden. Alle globale Deklarationen werden auch in der Altova Skripting-Projektdatei (.asprj-Datei) gespeichert.
- Das Makro kann durch Ausführung vom Skript-Editor aus getestet und debuggt werden.
- XMLSpy kann eine globale Skripting-Projektdatei haben und ein zweites Skripting-Projekt, das dem aktuell geladenen Projekt zugewiesen ist und jederzeit aktiv ist; die Makros stehen beiden Projekten zur Verfügung. Nähere Informationen dazu finden Sie unter [Ausführen eines Makros](#).

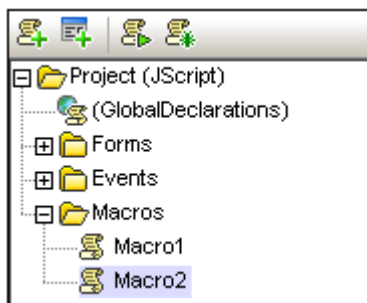
1.6.1 Erstellen und Bearbeiten eines Makros

Mit Hilfe der folgenden Operationen können Sie ein neues Makro erstellen und ein vorhandenes bearbeiten.

Erstellen eines neuen Makros

Klicken Sie mit der rechten Maustaste in der Skripting-Projektstruktur auf den Ordner "Macro" und wählen Sie im Kontextmenü den Befehl **Makro hinzufügen**. (Der Befehl **Makro hinzufügen** kann auch im Kontextmenü jedes Elements in der Skripting-Projektstruktur ausgewählt werden.) Klicken Sie alternativ dazu in der Symbolleiste der Skripting-Projektstruktur auf die Schaltfläche **Neues Makro**.

Das neu erstellte (und leere) Makro wird im Hauptfenster angezeigt und der Name des Makros wird in der Titelleiste des Skript-Editors angezeigt (*Abbildung unten*).



Benennen oder Umbenennen eines Makros

Um einem Makro einen Namen zu geben oder es umzubenennen, klicken Sie im Skripting-Projekt auf den Makronamen und drücken Sie die Taste F2 oder klicken Sie mit der rechten Maustaste auf den Makronamen und wählen Sie im Kontextmenü den Befehl "Umbenennen" aus.

Öffnen eines Makros

Um ein Makro zu öffnen, klicken Sie mit der rechten Maustaste im Ordner "Macros" der Skripting-Projektstruktur auf das Makro (*siehe Abbildung oben*) und wählen Sie den Befehl **Öffnen**. Das Makro wird im Hauptfenster angezeigt und sein Name wird in der Titelleiste des Skript-Editors angezeigt (*Abbildung unten*). Alternativ dazu können Sie ein Makro in der Skripting-Projektstruktur doppelklicken, um es im Hauptfenster zu öffnen.



Anmerkung: Wenn Sie in der Skripting-Projektstruktur auf ein Makrosymbol doppelklicken, wird das Makro im Hauptfenster geöffnet.

Bearbeiten des Makros

Um ein Makro zu bearbeiten, geben Sie Code ins Hauptfenster ein bzw. bearbeiten Sie diesen. Der folgende Code erstellt, zum Beispiel, das Formular `Form1` im Speicher und zeigt es dann an. `Form1` muss bereits (mit Hilfe der [Formularerstellung](#) des Scripting Editors) erstellt worden sein bevor dieses Makro ausgeführt wird.

```
objForm = CreateForm( 'Form1' );
objForm.ShowDialog();
```

Das folgende Makro werden mit Hilfe der Funktion `RemoveAllNamespaces` alle Namespaces im aktiven XML-Dokument entfernt.

```
if( Application.ActiveDocument != null) {

    RemoveAllNamespaces( Application.ActiveDocument.RootElement);
    Application.ActiveDocument.UpdateViews();
}
```

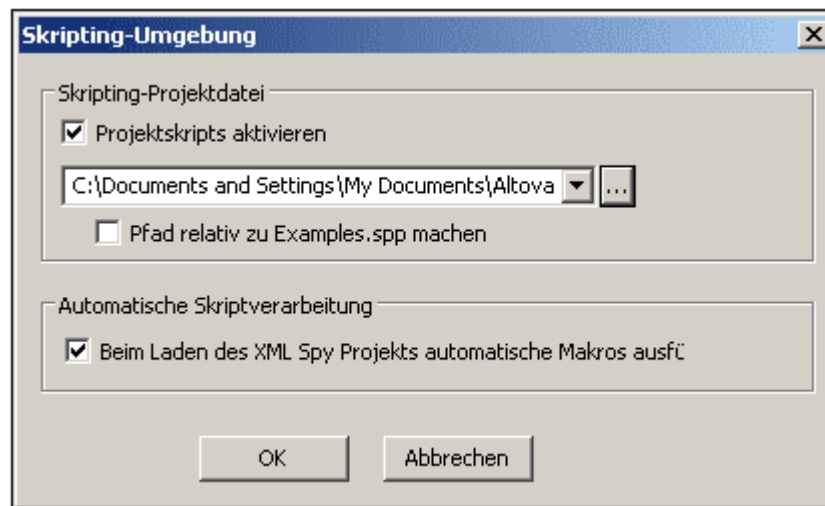
Die Funktion `RemoveAllNamespaces` selbst muss im "Global Declarations" Skript definiert werden. Nachdem die `RemoveAllNamespaces` Funktion definiert wurde, ist das Makro fertig und kann ausgeführt werden.

Anmerkung: Makros unterstützen Parameter oder Rückgabewerte nicht.

Definieren eines Makros als automatisches Makro

Wenn ein Makro als automatisches Makro definiert wurde, kann es automatisch ausgeführt werden, wenn: (i) XMLSpy gestartet wird oder (ii) ein Altova XMLSpy Projekt in XMLSpy geladen wird. Um festzulegen, ob bei beiden dieser Events automatische Makros ausgeführt werden sollen, aktivieren Sie im entsprechenden Dialogfeld im Bereich "Automatische Skriptverarbeitung" die Option *Automatische Markos ausführen*:

- *Beim Start von XMLSpy:* das Register "Skripting-Umgebung" des Dialogfelds "Optionen" (Menü **Extras | Optionen**) von XMLSpy
- *Wenn ein XMLSpy Projekt in XMLSpy geladen wird:* das Dialogfeld "Skripting-Umgebung" (*Abbildung unten*, Menübefehl **Projekt | Skriptingeinstellungen**).



Um ein Makro als automatisches Makro zu definieren, klicken Sie mit der rechten Maustaste in die Skripting-Projektstruktur und wählen Sie den Menübefehl **Als automatisches Makro definieren**. Dies ist ein Ein-/Aus-Befehl; um diese Einstellung für ein Makro zu entfernen, wählen Sie den Befehl erneut aus.

1.6.2 Ausführen eines Makros

Um ein Makro im Skript-Editor auszuführen, klicken Sie mit der rechten Maustaste in der Skripting-Projektstruktur auf das Makro und wählen Sie den Befehl **Makro ausführen**.

Es gibt drei verschiedene Arten, ein Makro von XMLSpy aus auszuführen:

- [Über das Menü Extras | Makros](#) von XMLSpy.
- [Durch Erstellung und Verwendung einer Symbolleiste-Schaltfläche oder eines Menübefehls](#) für ein Makro.
- [Durch Erstellung und Verwendung eines Menübefehls](#) für ein Makro.

Beachten Sie bitte: Es kann immer nur ein Makro gleichzeitig ausgeführt werden. Nachdem ein Makro (oder Event) ausgeführt wurde, wird das Skript geschlossen und globale Variablen verlieren ihre Werte.

Der XMLSpy Befehl zum Ausführen von Makros

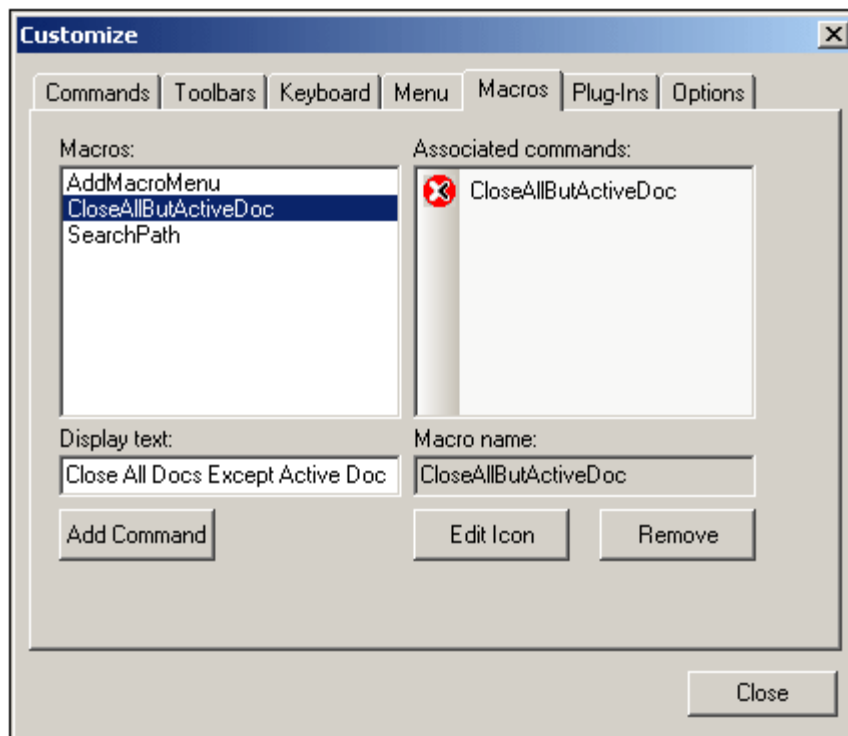
Mit dem Befehl **Extras | Makros** (Abbildung unten) wird ein Untermenü geöffnet, das die im aktuellen XMLSpy aktiven Skripting-Projekt definierten Makros enthält. Die aktiven Skripting-Projekte werden im Dialogfeld "Optionen" auf dem Register "Skripting-Umgebung" oder auf dem Register "Skripting-Umgebung" der Projekteinstellungen definiert.



Wählen Sie im Untermenü der verfügbaren Makros das zu startende Makro aus. Daraufhin wird das Makro ausgeführt.

Symbolleisten-Schaltfläche

Sie können eine Schaltfläche in der Symbolleiste oder einen Menübefehl erstellen, um ein ausgewähltes Makro auszuführen. Klicken Sie dazu auf **Extras | Anpassen | Makros**. Daraufhin wird das Dialogfeld "Anpassen" angezeigt (*Abbildung unten*).



Gehen Sie nun folgendermaßen vor:

1. Wählen Sie im Dialogfeld "Anpassen" auf dem Register "Makros" das gewünschte Makro aus dem Bereich "Makros" aus. In diesem Bereich werden die Makros im aktiven Skripting-Projekt (welches im Dialogfeld "Optionen" auf dem Register "Skripting-Umgebung" angezeigt wird) aufgelistet.
2. Geben Sie im Feld *Text anzeigen* den Namen der Schaltfläche ein. Dieser Name erscheint, wenn der Cursor über die Schaltfläche in der Symbolleiste platziert wird.
3. Klicken Sie auf **Befehl hinzufügen**, um den Befehl zur Liste der Befehle hinzuzufügen.
4. Wählen Sie den Befehl aus und klicken Sie auf **Symbol bearbeiten** um ein neues Schaltflächensymbol zu erstellen.
5. Ziehen Sie das fertige Symbol aus dem Fenster *Zugewiesene Befehle* in die Symbolleiste oder das Menü, wo sich der Cursor von einem Pfeil in einen I-Cursor oder eine Linie ändert.
6. Sie können Makros über das Dialogfeld "Anpassen" auf dem Register "Tastatur" sogar eigene Tastenkürzel zuweisen (*siehe Abbildung oben*).

Um die Symbolleisten-Schaltfläche zu entfernen, öffnen Sie im Dialogfeld "Anpassen" das Register "Makros" und ziehen Sie das Symbol aus der Symbolleiste in den Bereich *Zugewiesene Befehle*. Wählen Sie den Befehl im Bereich *Zugewiesene Befehle* aus und klicken Sie auf **Entfernen**, um den Befehl aus dem Bereich zu löschen.

Befehl im Menü "Extras"

Die XMLSpy API enthält eine Funktion `AddMacroMenuItem()`, um Makros als Menübefehle zum Menü **Extras** hinzuzufügen. Mit Hilfe dieser Funktion können Sie ein oder mehrere Makros zur

Liste der Makros für den Befehl **Extras | Makros** hinzufügen. Gehen Sie dabei normalerweise folgendermaßen vor:

1. Fügen Sie den Menübefehl durch Aufrufen der XMLSpy API-Funktion, `AddMacroMenuItem()` hinzu.

```
Application.AddMacroMenuItem("DeleteElements","Delete Elements  
Dialog");
```

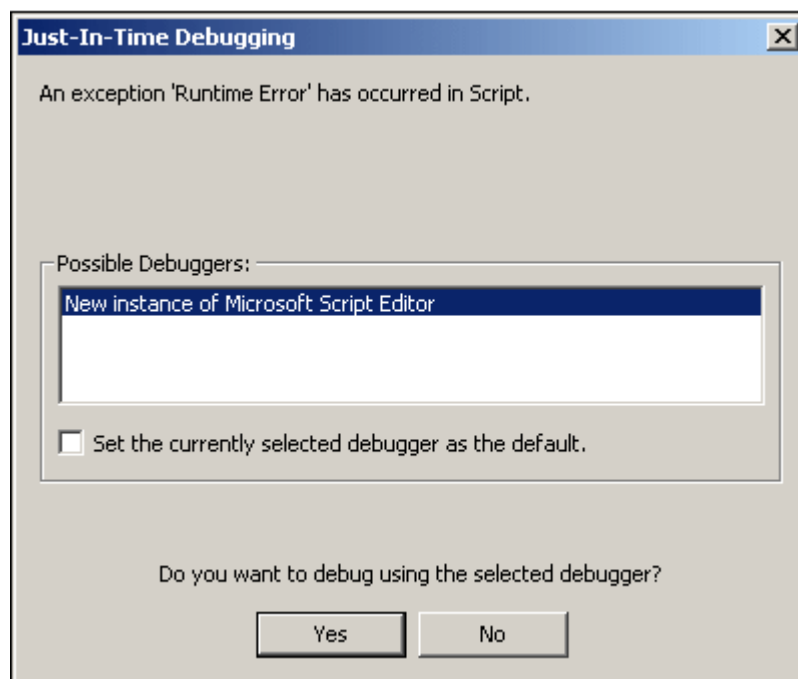
 - Der erste Parameter der Funktion (`DeleteElements` in der Beispielliste oben) ist der Name des Makros. Wenn Sie das Makro ausführen und es ein offenes Projekt gibt, das damit verknüpfte Skripts enthält, so sucht XMLSpy zuerst in den Projektskripts nach dem Makro.
Wenn es keine Projektskripts gibt, oder wenn XMLSpy das Makro nicht findet, sucht die Applikation in den globalen Skripts nach dem Makro.
 - Der zweite Parameter (`Delete Elements Dialog`) ist der Anzeigetext für den Menübefehl.
2. Setzen Sie das Menü **Extras** zurück, indem Sie `ClearMacroMenu()` aufrufen. Dadurch werden alle zuvor hinzugefügten Menübefehle entfernt.

Die beste Methode, diese beiden Funktionen aufzurufen, ist mit dem `Autorun` Makro des globalen Skripting-Projekts oder dem `On_OpenProject` Event.

1.6.3 Debuggen eines Makros

Sie können ein Makro mit Hilfe eines installierten Debuggers zu debuggen. Klicken Sie dazu mit der rechten Maustaste auf das Makro in der Skripting-Projektstruktur und wählen Sie den Befehl **Makro debuggen**.

Daraufhin wird das Dialogfeld "Just-In-Time" angezeigt (*Abbildung unten*), in dem die auf dem Rechner verfügbaren Debugger aufgelistet werden. Wählen Sie den gewünschten Debugger aus und klicken Sie auf **Yes**.



Der ausgewählte Debugger wird gestartet.

1.7 Anmerkungen zur Programmierung

Beachten Sie bitte die folgenden Anmerkungen zur Programmierung:

- Alle Namespaces und Typen der folgenden .NET Assemblies können standardmäßig im Microsoft .NET Framework aufgerufen werden:

```
System
System.Data
System.Design
System.Drawing
System.Windows.Forms
System.XML
```

Zusätzliche Assemblies können über das [Kontextmenü des Projekts](#) oder dynamisch im Quellcode (zur Laufzeit) mit Hilfe von [CLR.LoadAssembly](#) zum Skripting-Projekt hinzugefügt werden.

- Für Out-Parameter von Methoden des .NET Framework werden in JScript spezielle Variablen benötigt, z.B.:

```
// use JScript method to access out-parameters
var strError = new Array(1);
var nErrorPos = new Array(1);
var objBadData = new Array(1);
bOK = objDoc.IsValid( strError, nErrorPos, objBadData ); END
```

- Für Out-Parameter von Methoden der XMLSpy API werden in JScript spezielle Variablen benötigt. Unten sehen Sie einige Beispiele.

```
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
```

- .NET-Methoden, für die Integer-Argumente benötigt werden, sollten nicht direkt mit JScript Zahlenobjekten aufgerufen werden, bei denen es sich um Gleitkommawerte handelt.

Anstelle von:

```
var objCustomColor = CLR.Static( "System.Drawing.Color" ). FromArgb(
128, 128, 128 );
```

verwenden Sie:

```
var objCustomColor = CLR.Static( "System.Drawing.Color" ). FromArgb(
Math.floor( 128 ), Math.floor( 128 ), Math.floor( 128 ) );
```

- Zum Iterieren durch .NET Collections kann sowohl die JScript Enumerator-Technologie

als auch die .NET Iterator-Technologie verwendet werden:

Beispiel:

```
// 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-Vorlagen 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 Enum-Werte werden folgendermaßen aufgerufen:

```
var enumValStretch = CLR.Static( "System.Windows.Forms.ImageLayout"
).Stretch;
```

- Anstelle von numerischen Werten können jetzt Enumerations-Literale, wie in den Altova-Typbibliotheken definiert, verwendet werden.

```
objExportXMIFileDialog.XMIType = eXMI21ForUML23;
```

1.7.1 Vordefinierte Befehle

In diesem Abschnitt finden Sie eine Liste von:

- [vordefinierten Befehlen](#)
[alert](#)
[conform](#)
[doevents](#)
[createForm](#)
[lastform](#)
[prompt](#)
[showform](#)
[watchdog](#)
- und Befehlen zur [.NET-Interoperabilität](#).
[CLR.Create](#)
[CLR.Import](#)
[CLR.LoadAssembly](#)
[CLR.ShowImports](#)
[CLR.ShowLoadedAssemblies](#)
[CLR.Static](#)

Vordefinierte Befehle

Die folgenden vordefinierten Befehle stehen zur Verfügung.

ShowForm(strFormName : String)

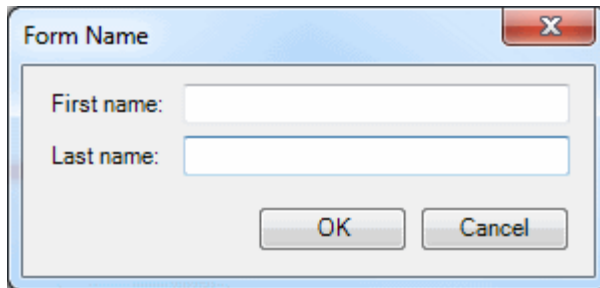
Instantiiert ein neues Formularobjekt anhand des angegebenen Formularnamens und zeigt es sofort als Dialogfeld an.

Rückgabewert: Eine Zahl, die für das generierte Dialogfeldergebnis DialogResult (`System.Windows.Forms.DialogResult`) steht.

Beispiel:

```
var dialogResult = ShowForm( "FormName" );
```

Zeigt das Formular "FormName" als Dialogfeld an:



Das Ergebnis (DialogResult) kann z.B. folgendermaßen überprüft werden:

```
if ( dialogResult == CLR.Static( "System.Windows.Forms.DialogResult" ).
    OK )
    alert( "ok" );
else
    alert( "cancel" );
```

CreateForm(strFormName : String)

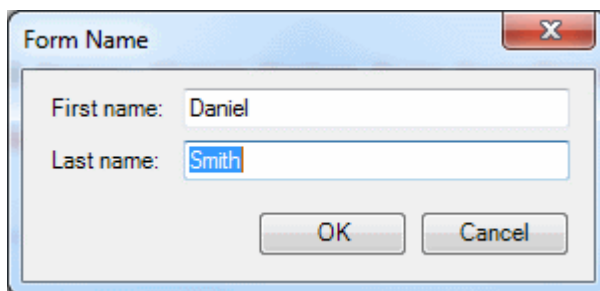
Instantiiert ein neues Formularobjekt anhand des vorgegebenen Formularnamens.

Rückgabewert: Das Formularobjekt (`System.Windows.Forms.Form`) des angegebenen Namens oder null, wenn kein Formular dieses Namens vorhanden ist.

Beispiel:

```
var myForm = CreateForm( "FormName" );
if ( myForm != null )
{
    myForm.textboxFirstName.Text = "Daniel";
    myForm.textboxLastName.Text = "Smith";
    var dialogResult = myForm.ShowDialog();
}
```

Zeigt das Formular "FormName" als Dialogfeld an - TextBoxes werden initialisiert:



Das Ergebnis (DialogResult) kann z.B. folgendermaßen überprüft werden:

:

```

if ( dialogResult == CLR.Static( "System.Windows.Forms.DialogResult" ).
OK )
    alert( "ok" );
else
    alert( "cancel" );

```

lastform

Mit Hilfe dieses globalen Felds kann das zuletzt erstellte Formularobjekt jederzeit wieder aufgerufen werden.

Rückgabewert: Gibt eine Referenz auf das letzte Formularobjekt (System.Windows.Forms.Form) zurück, das erfolgreich mittels CreateForm() oder ShowForm() instantiiert wurde.

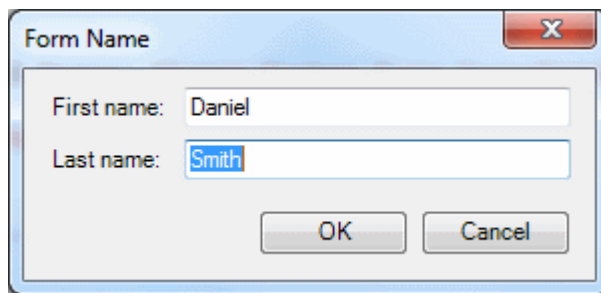
Beispiel:

```

CreateForm( "FormName" );
if ( lastform != null )
{
    lastform.textBoxFirstName.Text = "Daniel";
    lastform.textBoxLastName.Text = "Smith";
    var dialogResult = lastform.ShowDialog();
}

```

Zeigt das Formular "FormName" als Dialogfeld an - TextBoxes werden initialisiert (ähnlich wie CreateForm im Beispiel oben):

**doevents()**

Verarbeitet alle Windows-Meldungen, die sich derzeit in der Nachrichtenwarteschlange befinden.

Rückgabewert: Keiner

Beispiel:

```

for ( i=0; i < nLongLastingProcess; ++i )
{
    // do long lasting process

    doevents(); // process windows messages; give UI a chance to
update
}

```

watchdog(bEnable : boolean)

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.

Rückgabewert: Keiner

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
```

Verwendungstipp:

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.

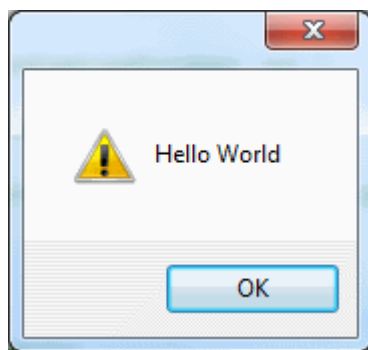
`alert(strMessage : String)` oder `MsgBox(strMessage : String)`

Zeigt eine bestimmte Meldung über ein Meldungsfeld an. Der Benutzer muss zum Fortfahren auf OK klicken.

Rückgabewert: Keiner

Beispiel:

```
alert( "Hello World" );
```



`confirm(strMessage : String)`

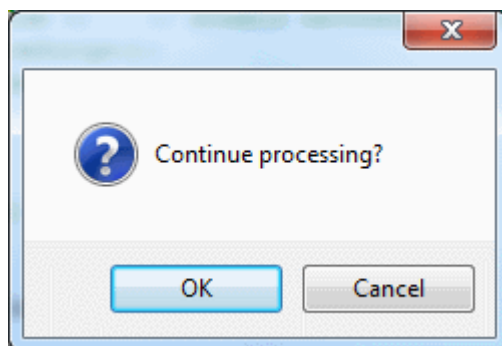
Öffnet ein Dialogfeld, in dem die angegebene Bestätigungsmeldung angezeigt wird.

Eine Bestätigungsbox dient oft dazu, etwas zu überprüfen oder zu akzeptieren. Der Benutzer muss entweder auf "OK" oder auf "Abbrechen" klicken, um fortfahren zu können.

Rückgabewert: Ein Boolescher Wert, der für die Antwort des Benutzers steht. Wenn der Benutzer auf "OK" klickt, gibt das Dialogfeld "true" zurück, wenn der Benutzer auf "Abbrechen" klickt, gibt das Dialogfeld "false" zurück.

Beispiel:

```
if ( confirm( "Continue processing?" ) == false )  
    return;
```



`prompt(strMessage : String, strDefault : String)`

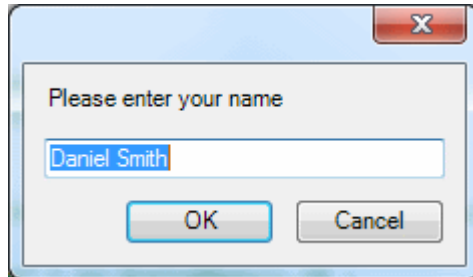
Öffnet ein Dialogfeld, in dem die angegebene Eingabeaufforderung sowie ein Textfeld-Steuererelement mit einer Standardantwort angezeigt wird.

Ein solches Aufforderungskästchen dient oft zur Eingabe eines String-Werts.

Rückgabewert: Ein String, der den Textfeldwert enthält oder Null, wenn der Benutzer auf "Abbrechen" geklickt hat.

Beispiel:

```
var name = prompt( "Please enter your name", "Daniel Smith" );
if ( name != null )
    alert( "Hello " + name + "!" );
```



Befehle zur .NET-Interoperabilität

Zur verbesserten Interoperabilität mit dem .NET Framework stehen unter CLR zusätzliche Funktionen zur Verfügung.

CLR.Import(strNamespaceCLR : String)

Dies ist das Skripting-Äquivalent zur C# / VB.Net Verwendung des Schlüsselworts *"imports"*. Damit können Sie den angegebenen Namespace in aufeinander folgenden Aufrufen wie z.B. `CLR.Create()` und `CLR.Static()` auslassen.

Rückgabewert: Keiner

Beispiel:

Anstatt immer den vollständigen qualifizierten Namen 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 Namespaces importiert und die Kurzform verwendet werden:

```
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 );
}
```

Bitte beachten Sie:

Durch den Import eines Namespace wird die entsprechende Assembly nicht zum Skripting-Projekt hinzugefügt bzw. geladen!

Assemblies können mittels zum Skripting-Projekt hinzugefügt oder mittels [CLR.LoadAssembly](#) dynamisch (zur Laufzeit) im Quellcode eingefügt werden.

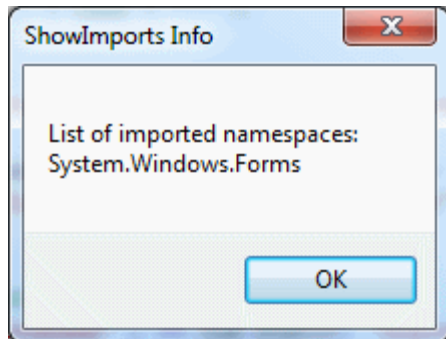
CLR.ShowImports()

Öffnet ein Meldungsdialogfeld, in dem die aktuell importierten Namespaces angezeigt werden. Der Benutzer muss zum Fortfahren auf OK klicken.

Rückgabewert: Keiner

Beispiel:

```
CLR.ShowImports();
```



CLR.LoadAssembly(strAssemblyNameCLR : String)

Lädt die .NET Assembly mit dem angegebenen Assembly-Namen oder Dateipfad.

Rückgabewert: ein Boolescher Wert. True, wenn die Assembly geladen werden konnte, false, wenn nicht.

Beispiel:

```
// 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" );
}
```

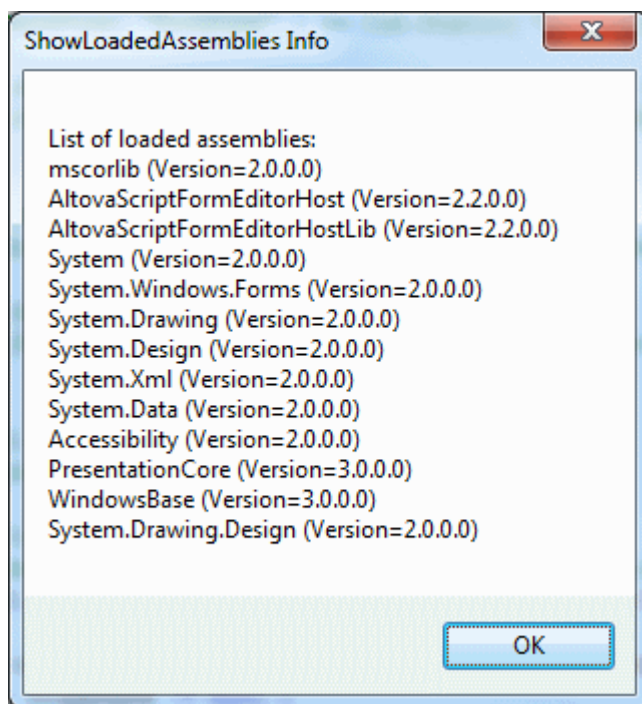
CLR.ShowLoadedAssemblies()

Öffnet ein Meldungsdialogfeld, in dem die aktuell geladenen Assemblies angezeigt werden. Der Benutzer muss zum Fortfahren auf OK klicken.

Rückgabewert: Keiner

Beispiel:

```
CLR.ShowLoadedAssemblies();
```



CLR.Create(strTypeNameCLR : String, constructor arguments ...)

Erstelle eine neue .NET-Objektinstanz für den angegebenen Typnamen. Wenn mehr als ein Argument übergeben wird, werden die nachfolgenden Argumente als Argumente für den Konstruktor des .NET-Objekts interpretiert.

Rückgabewert: Eine Referenz auf das erstellte .NET-Objekt

Beispiele:

```
var objArray = CLR.Create("System.Collections.ArrayList");

var newItem = CLR.Create("System.Windows.Forms.ListViewItem",
    "NewItemText");

var coll = CLR.Create("System.Collections.Generic.List<System.String>");

CLR.Import("System");
CLR.Import("System.Collections.Generic");
var dictionary = CLR.Create("Dictionary<String, Dictionary<String,
String >>");
```

CLR.Static(strTypeNameCLR : String)

Verschafft Zugriff auf .NET-Typen, die keine Instanzen haben und nur statische Mitglieder enthalten.

Rückgabewert: Eine Referenz auf das statische .NET-Objekt

Beispiele:

```
var enumValStretch = CLR.Static("System.Windows.Forms.ImageLayout")
    ).Stretch

var clipboard = CLR.Static("System.Windows.Clipboard");
clipboard.SetText("HelloClipboard");

if ( ShowForm("FormName") == CLR.Static(
    "System.Windows.Forms.DialogResult").OK )
    alert("ok");
```



```
else
    alert( "cancel" );
```

Verwendung von Formularen und Befehle

Formulare werden folgendermaßen verwendet:

Bei Formularobjekten kann die Formularelementenstruktur über Feldzugriff aufgerufen werden

Beispiel: Angenommen Sie haben das folgende Formular:

```
MyForm
  ButtonPanel
    OkButton
    CancelButton
  TextEditor
  AxMediaPlayer1

TrayComponents:
  MyTimer
```

Das Formular kann dann vom Skript aus instantiiert als:

```
var objForm = CreateForm( "MyForm" );
```

Um eine seiner Komponenten aufzurufen, kann der Feldaufruf verwendet werden:

```
objForm.ButtonPanel.OkButton.Enabled = false;
```

oder

```
objForm.TextEditor.Text = "Hello World";
```

Um Komponenten aus der Taskleiste aufzurufen, verwenden Sie die folgende Methode im Formularobjekt:

```
var objTrayComponent = <A form object>.GetTrayComponent( strComponentName
: String );
```

Um in unserem Beispiel eine Referenz zur Timer-Komponente abzurufen, um deren Verwendung zu aktivieren, verwenden Sie den folgenden Code:

```
var objTimer = objForm.GetTrayComponent( "MyTimer" );
objTimer.Enabled = true;
```

Bei ActiveX Controls kann das zugrunde liegende COM-Objekt über die OCX-Eigenschaft abgerufen werden:

```
var ocx = lastform.AxMediaPlayer1.OCX; // get underlying COM object
ocx.enableContextMenu = true;
ocx.URL = "mms://apasf.apa.at/fm4_live_worldwide";
```

1.8 Migration zum Skript-Editor 2010 und höheren Versionen

Im Skript-Editor von XMLSpy wird ab Version 2010 eine andere zugrunde liegende Technologie verwendet als in früheren Versionen. Daher müssen Skripting-Projekte, die mit früheren Versionen von XMLSpy erstellt wurden (vor Version 2010), geändert werden. Beachten Sie dabei die folgenden Punkte:

- Wenn ein früheres Skripting-Projekt (.prj-Datei) mit dem neuen Skript-Editor (Version 2010 und höher) geöffnet wird, wird das visuelle Layout von Formularen so gut wie möglich übernommen und die Skripts werden aus der .prj-Datei kopiert. Anschließend müssen Sie die Skripts an die neue im Skript-Editor verwendete Technologie, die in dieser Dokumentation beschrieben ist, anpassen.
- Das `TheView`-Objekt: In der alten Skripting-Umgebung gab es eine künstliche Eigenschaft namens `TheView`, die nur innerhalb von Event-Handletern aufgerufen werden konnte. Sie diente zum Aufrufen des Formulars, das das Event (entweder direkt oder über eines seiner untergeordneten Steuerelemente) auslöste. In der neuen Skripting-IDE gibt es diese künstliche Eigenschaft **nicht**. Statt dessen steht dieselbe Funktionalität und viele andere mit orthogonalen [vordefinierten Skripting-Hilfsfunktionen](#), kombiniert mit den Möglichkeiten des .NET-Framework zur Verfügung.
- Da alle Event-Handler in der neuen Skripting-Umgebung als ersten Parameter ein Absenderobjekt erhalten, steht die Quelle, die das Event ausgelöst hat, immer zur Verfügung. Durch Aufruf der .NET-Funktion `FindForm()` im Absenderobjekt kann das Formularobjekt jederzeit aufgerufen werden. Alternativ dazu kann die vordefinierte Eigenschaft `lastform` verwendet werden (wenn nur ein Formular verwendet wird). Beachten Sie, dass die Verwendung von `lastform` nicht auf Event-Handler beschränkt ist (wie dies bei `TheView` der Fall war). Diese Funktion kann überall im Skript-Code verwendet werden.

Unten finden Sie eine Liste von Methoden und Eigenschaften des `TheView`-Objekts. Zu jeder dieser Methoden und Eigenschaften gibt es in der neuen Skripting-Umgebung einen alternativen Mechanismus.

Methoden

Die folgenden Methoden wurden vom `TheView`-Objekt bereitgestellt und müssen, wie beschrieben, migriert werden:

`Cancel()`

In der neuen Skripting-Umgebung erzielen Sie dasselbe Ergebnis mit: `lastform.Close(); //`
`Use .NET Form.Close()`

`IsFormOpen(Name as String) as Boolean`

Da es bei .NET-Formularen einen Unterschied zwischen dem Anzeigen eines Formulars und der Instantiierung eines Formulars gibt, lässt sich das frühere Konzept nicht direkt übertragen. Stattdessen kann der Benutzer fragen, ob ein bestimmtes Formular gerade angezeigt wird. Zum Beispiel:

```
var objFormPencilSelector = CreateForm("PencilSelector");
var objFormColorSelector = CreateForm("ColorSelector");
...
// Anywhere in code ...

if(objFormColorSelector.Visible)
{
    ...
}
```

```
}
```

FormFind(Name as String) as object

Sie können in der neuen Skripting-Umgebung mehrere Formulare derselben Art instantiieren. In der alten Skripting-Umgebung durfte jedes Formular immer nur einmal vorhanden sein. Daher gibt es in der neuen Skripting-Umgebung kein Äquivalent zu `FormFind()`.

OpenDoc(File as String)

Dasselbe Ergebnis erhalten Sie mit: `Application.OpenDocument( File as String )`

PumpData()

Dies entspricht der vordefinierten Funktion `doevents()`, die alle gerade in der Nachrichtenwarteschlange befindlichen Nachrichten verarbeitet.

RunClick(), RunInitialize(), RunTerminate()

Für diese Methoden gibt es keine direkten Ersatzmethoden. Rufen Sie stattdessen die entsprechenden Handler direkt auf.

Eigenschaften

Die folgenden Eigenschaften wurde vom `TheView` Objekt bereitgestellt und müssen, wie beschrieben, migriert werden:

ToolTipText as String

Um in der neuen Skripting-Umgebung Tooltips verwenden zu können, können Sie die .NET Infrastruktur verwenden. Diese gestattet die detaillierte Steuerung des Verhaltens von Tooltips (Anpassen der Verzögerung, Wann diese angezeigt werden sollen usw.). Um z.B. Tooltips für ein Formular mit zwei Steuerelementen bereitzustellen, können Sie den folgenden Code zum Load Event Handler des Formulars hinzufügen:

```
//Occurs whenever the user loads the form.
function MyForm_Load( objSender, e_EventArgs )
{
    // Create the ToolTip and associate with the Form container.
    var toolTip = CLR.Create( "System.Windows.Forms.ToolTip" );

    // Set up the delays for the ToolTip.
    toolTip.AutoPopDelay = 3000;
    toolTip.InitialDelay = 1000;
    toolTip.ReshowDelay = 500;

    // Force the ToolTip text to be displayed whether or not
    // the form is active.
    toolTip.ShowAlways = true;

    // Set up the ToolTip text for several Controls.
    toolTip.SetToolTip( objSender.ProgressBar1,
        "Shows the progress of the operation" );
    toolTip.SetToolTip( objSender.Button1,
        "Click Button to start the processing" );
}
```

Color as Long

Da alle Formulare/Steuerelemente in der neuen Skripting-Umgebung .NET-Steuerelemente aus dem `System.Windows.Forms` Namespace sind, gibt es zahlreiche Möglichkeiten, Farben, Hintergrundbild, Schriftarten und andere visuelle Aspekte anzupassen. So hat z.B. jede visuelle Komponente die Eigenschaften `BackColor` und `ForeColor`, um das Aussehen zu ändern. Mit Hilfe des folgenden Handlers können Sie die Farbe einer Schaltfläche während der Laufzeit ändern.:

```
function TestForm_Button1_Click( objSender, e_EventArgs )  
{  
    objSender.BackColor = CLR.Static( "System.Drawing.Color" ).SlateBlue;  
}
```

Nähere Informationen zu diesem Thema finden Sie in der .NET-Dokumentation:
<http://msdn.microsoft.com/en-us/library/system.windows.forms.aspx>

2 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](#) Schnittstelle implementiert. Ein Beispiel mit C++ finden Sie unter [ATL-Beispieldateien](#). Die entsprechenden Dateien befinden sich im folgenden Ordner Ihrer XMLSpy Installation.

```
C:\Documents and Settings\\My Documents\Altova\XMLSpy2012  
\Examples\CppIDEPlugin
```

Ein Beispiel, in dem VisualBasic verwendet wird, finden Sie im folgenden Ordner Ihrer XMLSpy Installation:

```
C:\Documents and Settings\\My Documents\Altova\XMLSpy2012  
\Examples\XMLSpyPlugInActiveX
```

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:

Value name:	ProgID of the plug-in
Value type:	must be REG_SZ
Value data:	CLSID of the component

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 Sie bei der manuelle Registrierung auf Probleme stoßen, können Sie überprüfen, ob die CLSID Ihres Plug-In im "PlugIns" Schlüssel korrekt registriert ist. Wenn dies nicht der Fall ist, war der Name Ihrer Plug-In DLL 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.

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.

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](#) 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 16 x 16 Pixel haben und darf maximal 256 Farben enthalten. Die Bildreferenzen müssen in einer einzigen (<ImageFile>...) Zeile von links nach rechts angeordnet sein. 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. Ab Version 4.3 können Benutzeroberflächenelemente auch aus XMLSpy entfernt werden.

Struktur von Modification-Elementen

Alle Modification-Elemente bestehen aus den folgenden beiden Child-Elementen:

```
<Modification>
  <Action>Type of action</Action>
  <UIElement Type="type of UI element">
    </UIElement>
  </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.

Sie können Werte des **Action**-Elements kombinieren z.B. "Hide Remove"

Das Element **UIElement** beschreibt alle neuen oder bestehenden Benutzeroberflächenelemente für XMLSpy. Mögliche Elemente sind derzeit: neue Symbolleisten, Schaltflächen, Menüs oder Menübefehle. Das Attribut **type** definiert, welches Benutzeroberflächenelement vom XML-Element beschrieben wird.

Gemeinsame Child-Elemente von UIElement

Die Elemente "ID" und "Name" sind für alle verschiedenen Arten von XML UIElement-Fragmenten gültig. Es ist jedoch möglich, einen der Werte für einen bestimmten Typ von UIElement zu ignorieren. 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 ID-Fragment einen neuen Teil der Benutzeroberfläche beschreibt, so kann die ID beliebig gewählt werden und der Wert sollte weniger als 1000 sein. Das Element **Name** definiert den Textwert. Bestehende UI-Elemente können nur nach ihrem Namen identifiziert werden, z.B. Menüs und Menübefehle mit den dazugehörigen Untermenüs. 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. 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 kurzen 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 externen 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. **MenuID** 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 Möglichkeit, um das übergeordnete Element des Menübefehls zu definieren.

Der Wert des Elements **Parent** ist ein Pfad zu einem Menübefehl. Der Textwert des Elements "Parent" muss gleich dem **übergeordneten Menünamen** des Untermenüs sein, wobei der Name des Untermenüs durch einen Doppelpunkt getrennt wird. Wenn das Menü kein übergeordnetes Menü hat, da es kein Untermenü ist, stellen Sie dem Namen am Anfang einen Doppelpunkt voran. Das Attribut **type** muss den Wert "MenuItem" haben. Beispiel für ein **UIElement**, das einen Menübefehl definiert:

```
<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 in XMLSpy Trennlinien für Symbolleisten und Menüs hinzufügen, wenn der Wert des Elements **ID** auf 0 gesetzt ist.

2.4 ATL-Beispieldateien

Auf den folgenden Seiten wird beschrieben, wie Sie mit Hilfe von ATL eine einfache XMLSpy IDE Plug-In DLL erstellen. Dazu müssen Sie mit ATL und den Assistenten, die neue ATL-Objekte generieren, sowie MS VisualStudio vertraut sein.

Zum Aufrufen der API importiert die Implementierung die Typlibibliothek von XMLSpy. Der Code liest verschiedene Eigenschaften aus und ruft mit Hilfe des über die #import-Anweisung zur Verfügung gestellten Smart Pointer Methoden auf.

Außerdem werden im Beispielcode der MFC Class String 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 im Menü "File" den Befehl "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 im Menü "Insert" den Befehl "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 implementiert, wird auf den nächsten Seiten beschrieben.

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-A1F3-270692E08AFC ),

//     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 -----
    }
}

```

```
};
};
```

2.4.2 Klassendefinition

In der Klassendefinition des ATL-Objekts müssen einige Änderungen vorgenommen werden. Die Klasse muss von `IXMLSpyPlugIn` abgeleitet sein, die "Interface Map" benötigt einen Eintrag für `IXMLSpyPlugIn` und die Methoden der IDE Plug-In-Schnittstelle müssen deklariert 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 STDMETHODCALLTYPE OnCommand(long nID, IDispatch* pXMLSpy);

    virtual HRESULT STDMETHODCALLTYPE OnUpdateCommand(long nID, IDispatch* pXMLSpy,
        SPYUpdateAction* pAction);

    virtual HRESULT STDMETHODCALLTYPE OnEvent(long nEventID, SAFEARRAY** arrayParameters,
        IDispatch* pXMLSpy, VARIANT* pReturnValue);

    virtual HRESULT STDMETHODCALLTYPE GetUIModifications(BSTR* pModificationsXML);

    virtual HRESULT STDMETHODCALLTYPE GetDescription(BSTR* pDescription);
};

#endif // __XMLSPYIDEPLUGIN_H_
```

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;

        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 );
}
```

2.5 IXMLSpyPlugIn

Siehe auch

Methoden

[OnCommand](#)

[OnUpdateCommand](#)

[OnEvent](#)

[GetUIModifications](#)

[GetDescription](#)

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.

2.5.1 OnCommand

Siehe auch

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 Symbolleiste-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.

Example

```
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 (*.*)|*.*||"
            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

```

2.5.2 OnUpdateCommand

Siehe auch

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

```

2.5.3 OnEvent

Siehe auch

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:

<code>On_BeforeStartEditing</code>	<code>= 1</code>
<code>On_EditingFinished</code>	<code>= 2</code>
<code>On_FocusChanged</code>	<code>= 3</code>
<code>On_BeforeDrag</code>	<code>= 4</code>
<code>On_BeforeDrop</code>	<code>= 5</code>
<code>On_OpenProject</code>	<code>= 6</code>
<code>On_OpenDocument</code>	<code>= 7</code>
<code>On_CloseDocument</code>	<code>= 8</code>
<code>On_SaveDocument</code>	<code>= 9</code>

Events, die ab XMLSpy 4r4 verfügbar sind:

<code>On_DocEditDragOver</code>	<code>= 10</code>
<code>On_DocEditDrop</code>	<code>= 11</code>
<code>On_DocEditKeyDown</code>	<code>= 12</code>
<code>On_DocEditKeyUp</code>	<code>= 13</code>
<code>On_DocEditKeyPressed</code>	<code>= 14</code>
<code>On_DocEditMouseMove</code>	<code>= 15</code>
<code>On_DocEditButtonUp</code>	<code>= 16</code>
<code>On_DocEditButtonDown</code>	<code>= 17</code>
<code>On_DocEditContextMenu</code>	<code>= 18</code>
<code>On_DocEditPaste</code>	<code>= 19</code>
<code>On_DocEditCut</code>	<code>= 20</code>
<code>On_DocEditCopy</code>	<code>= 21</code>
<code>On_DocEditClear</code>	<code>= 22</code>
<code>On_DocEditSelectionChanged</code>	<code>= 23</code>

Events, die ab XMLSpy 2004 verfügbar sind:

<code>On_DocEditDragOver</code>	<code>= 10</code>

Events, die ab XMLSpy 2004r4 (Typbibliothek Version 1.4) verfügbar sind:

On_BeforeOpenProject	= 25
On_BeforeOpenDocument	= 26
On_BeforeSaveDocument	= 27
On_BeforeCloseDocument	= 28
On_ViewActivation	= 29
On_DocEditKeyboardEvent	= 30
On_DocEditMouseEvent	= 31

Events, die ab XMLSpy 2006 SP1 (Typbibliothek Version 1.5) verfügbar sind:

On_BeforeValidate	= 32

Events, die ab XMLSpy 2007 (Typbibliothek Version 1.6) verfügbar sind:

On_BeforeShowSuggestions	= 33
On_ProjectOpened	= 34
On_Char	= 35

Events, die ab XMLSpy 2009 (Typbibliothek Version 2.2) verfügbar sind:

On_Initialize	= 36
On_Running	= 37
On_Shutdown	= 38

Events, die ab XMLSpy 2012 (Typbibliothek Version 2.8) verfügbar sind:

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 Dokumentobjekt 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.

2.5.4 GetUI Modifications

Siehe auch

Deklaration: `GetUI Modifications()` als `String`

Beschreibung

Die `GetUI Modifications()` 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](#).

Beispiel

```
Public Function IXMLSpyPlugIn_GetUI Modifications() As String
    ' GetUI Modifications() 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_GetUI Modifications = strMods
        Else
            IXMLSpyPlugIn_GetUI Modifications = ""
        End If
    End If
End Function
```

2.5.5 GetDescription

Siehe auch

Deklaration: `GetDescription()` als `String`

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
```

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](#) 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 mit Skripting-Sprachen wie JavaScript und VBScript und von gängigen Entwicklungsumgebungen wie z.B. solchen, die C, C++, VisualBasic und Delphi verwenden, aufgerufen werden.

Ausführungsumgebungen für die Applikations-API

Die Applikations-API kann über die folgenden Ausführungsumgebungen aufgerufen werden:

- externe Programme (Beschreibung siehe [unten](#) und in diesem Abschnitt im Kapitel [Übersicht](#))
- vom integrierten Skript-Editor von XMLSpy aus. Eine Beschreibung zur Skripting-Umgebung finden Sie im Abschnitt [Skript-Editor](#).
- 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](#).
- über ein ActiveX Control, das verfügbar ist, wenn das [Integration Package](#) installiert wurde. Nähere Informationen dazu finden Sie im Abschnitt [ActiveX Integration](#).

Externe Programme

Im Abschnitt [Übersicht](#) 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 [Programmier Sprachen](#).

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](#) 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#](#) 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](#): 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.

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 XMLSpy im 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](#) (und in [C#](#) and [Java](#)).
- 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 Teil besteht aus einer [Übersicht](#), in der das Objektmodell für die API beschrieben wird. Weiters wird beschrieben, wie Sie die API über verschiedene [Programmiersprachen](#) aufrufen.
- Der zweite Teil ist ein Referenzteil ([Schnittstellen](#) und [Enumerationen](#)), der Beschreibungen der Schnittstellenobjekte der Applikations-API enthält.

3.1 Übersicht

Diese Übersicht über die Applikations-API ist in die folgenden Abschnitte gegliedert:

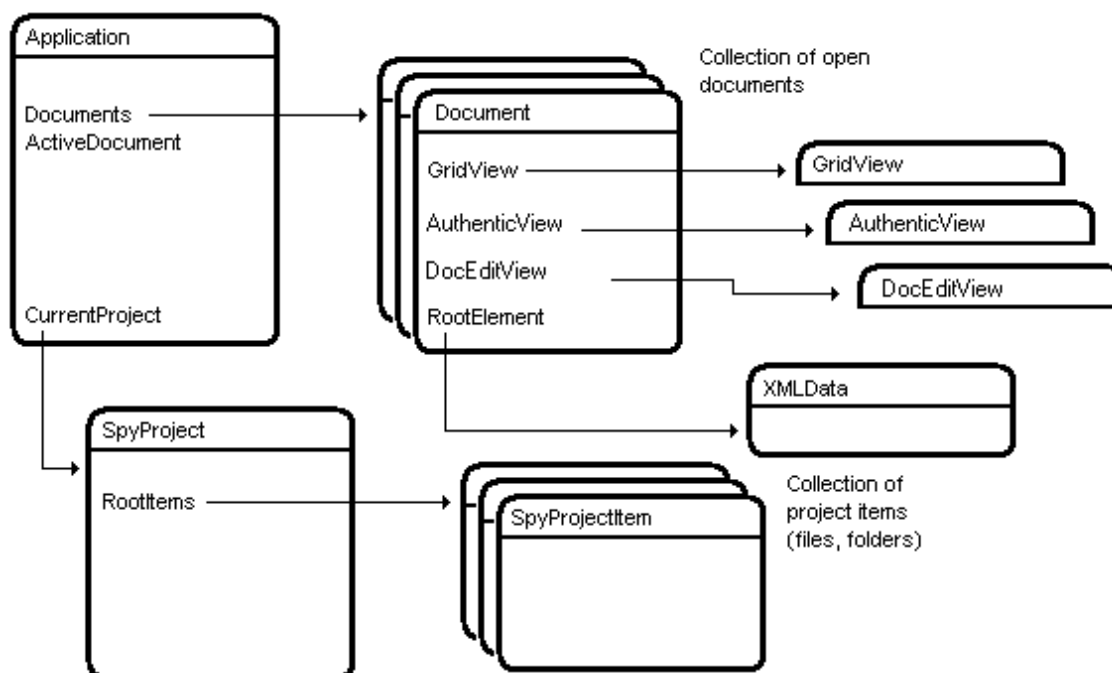
- [Das Objektmodell](#): Hier werden die Beziehungen zwischen den Objekten der Applikations-API beschrieben.
- [Programmiersprachen](#): Hier wird beschrieben, wie Sie die Funktionalitäten der Applikations-API über die gebräuchlichsten Programmiersprachen (JScript, VBScript, C# und Java) aufrufen können. Die Codefragmente aus den mitinstallierten Beispieldateien dienen zum Veranschaulichen einiger grundlegender Mechanismen.
- [DOM und XMLData](#): Hier wird die Beziehung zwischen der `XMLData`-Schnittstelle der Applikations-API und der DOM erläutert.
- [Nicht mehr verwendete Zeilenoperationen in der Authentic-Ansicht](#): Hier finden Sie Informationen zu nicht mehr verwendeten Objekten für Tabellenzeilenoperationen in der Authentic-Ansicht.
- [Nicht mehr verwendete Bearbeitungsoptionen in der Authentic-Ansicht](#): Hier finden Sie Informationen zu nicht mehr verwendeten Objekten für Bearbeitungsoperationen in der Authentic-Ansicht.

3.1.1 Objektmodell

Der Ausgangspunkt für alle Applikationen, die die Applikations-API verwenden, ist das [Application](#) Objekt. Dieses Objekt enthält allgemeine Methoden wie Import/Export-Funktionen und Referenzen auf offene Dokumente und das offene Projekt, falls vorhanden.

Um eine Instanz des `Application` Objekts zu erstellen, rufen Sie von VisualBasic aus `CreateObject("XMLSpy.Application")` oder von der Entwicklungsumgebung Ihre Wahl eine ähnliche Funktion auf, um ein COM-Objekt zu erstellen. Es müssen keine weiteren Objekte erstellt werden, um die komplette Applikations-API zu verwenden (dies ist auch gar nicht möglich). Alle anderen Schnittstellen werden über andere Objekte aufgerufen, wobei das `Application`-Objekt als Ausgangspunkt dient.

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](#) Schnittstelle öffnen.

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. Diese Grundfunktionen befinden sich in Dateien im Ordner API Examples und können sofort verwendet werden. Den Pfad zum Ordner "API Examples" finden Sie unten:

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/Altova/XMLSpy20XX/Examples/API/

JScript

Im JScript-Code werden die folgenden grundlegenden Funktionen verwendet:

- [Starten der Applikation oder Anhängen einer laufenden Instanz](#)
- [Einfacher Dokumentzugriff](#)
- [Iteration](#)
- [Fehlerbehandlung](#)
- [Events](#)
- [Import und Export von Daten](#)

VBScript

VBScript unterscheidet sich nur syntaktisch von JScript und funktioniert ansonsten auf genau die gleiche Art. Der unten aufgelistete Code enthält ein Beispiel für die Verwendung von VBScript. Nähere Informationen finden Sie in den [JScript-Beispielen](#).

- [Events](#): 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](#): Startet XMLSpy, das als Automation Server registriert ist, oder aktiviert das Programm, wenn XMLSpy bereits ausgeführt wird.
- [Open OrgChart.pxf](#): 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](#): Zeigt, wie die Applikation auf XMLSpy Events hört. Ist die Funktion aktiv, erscheint ein Meldungsfeld, nachdem ein Dokument geöffnet wurde.
- [Open ExpReport.xml](#): Öffnet ein weiteres Beispieldokument.
- [Toggle View Mode](#): 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](#): 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](#): Beendet XMLSpy.

Java

Sie können mit Hilfe von Java auf die Funktionalitäten der Application API zugreifen. [In diesem Abschnitt](#) 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](#)
 - [Java-Beispielprojekt](#)
 - [Starten und Beenden der Applikation](#)
 - [Einfacher Dokumentaufruf](#)
 - [Iterationen](#)
 - [Verwendung von Out-Parametern](#)
 - [Event Handler](#)

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](#)
- [Einfacher Dokumentzugriff](#)
- [Iteration](#)
- [Fehlerbehandlung](#)
- [Events](#)
- [Import und Export von Daten](#)
- [Beispiel: Bubble-Sortierung dynamischer Tabellen](#)

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 XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/Altova/XMLSpy20XX/Examples/API/

Die Beispieldateien können auf zwei Arten ausgeführt werden:

- *Über die Befehlszeile:* Öffnen Sie ein Eingabeaufforderungsfenster und geben Sie den Namen eines der Beispiel-Skripts ein (z.B. `Start.js`). Der Windows Scripting Host, der neuere Versionen von Windows (XP, Vista, 7) enthält, führt das Skript aus.
- *Über den Windows Explorer:* Navigieren Sie im Windows Explorer zur JScript-Datei und doppelklicken Sie darauf. Der Windows Scripting Host, der neuere Versionen von Windows (XP, Vista, 7) enthält, führt das Skript aus. Die Befehlskonsole wird nach Ausführung des Skript automatisch geschlossen.

Applikation starten

Mit dem unten aufgeführten JScript-Code wird die Applikation gestartet und beendet. Wenn bereits eine Instanz der Applikation ausgeführt wird, so wird diese Instanz aufgerufen.

Skript-Code

Der nachstehende JScript-Code enthält Erklärungen im Code.

```
// 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)
{
    Exit("Can't access or create XMLSpy.Application");
}
```

```
// if newly started, the application will start without its UI visible. Set it to
// visible.
objSpy.Visible = true;

WScript.Echo("Hello");

objSpy.Visible = false;           // will shutdown application if it has no more COM
connections                       connections
//objSpy.Visible = true;         // will keep application running with UI visible
```

Ausführen des Skripts

Der oben angeführte JScript-Code steht in der Datei `Start.js` im Ordner `JScript` im Ordner `API Examples` zur Verfügung:

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/ Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/ Altova/XMLSpy20XX/Examples/API/

Um das Skript auszuführen, starten Sie es über ein Eingabeaufforderungsfenster oder über den Windows Explorer.

Einfacher Dokumentaufruf

Im unten aufgelisteten JScript-Code wird gezeigt, wie man Dokumente öffnet, ein Dokument zum aktiven Dokument macht, durch die offenen Dokumente iteriert und Dokumente schließt.

Ausführen des Skripts

Der unten aufgelistete JScript-Code steht in der Datei `DocumentAccess.js` im Ordner `JScript` des Ordners `API Examples` zur Verfügung:

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/ Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/ Altova/XMLSpy20XX/Examples/API/

Um das Skript auszuführen, starten Sie es über ein Eingabeaufforderungsfenster oder über den Windows Explorer.

Skript-Code

Der nachstehende JScript-Code enthält Erklärungen im Code.

```
// Initialize application's COM object. This will start a new instance of the
// application and
// return its main COM object. Depending on COM settings, a the main COM object of an
// already
// running application might be returned.
try
{
    objSpy = WScript.GetObject("", "XMLSpy.Application");
}
catch(err)
```

```

{
    Exit("Can't access or create XMLSpy.Application");
}

// if newly started, the application will start without its UI visible. Set it to
// visible.
objSpy.Visible = true;

// ***** Accessing documents, setting the active document
// *****

// Locate examples via USERPROFILE shell variable. The path needs to be adapted to major
// release versions.
objWshShell = WScript.CreateObject("WScript.Shell");
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\My
Documents\\Altova\\XMLSpy2012\\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();

// ***** Two ways of iterating; close documents
// *****

// Go through all open documents using a JScript Enumerator
bRequiresSaving = false;
for (var iterDocs = new Enumerator(objSpy.Documents); !iterDocs.atEnd(); iterDocs.
moveNext())
    if (iterDocs.item().IsModified)
        bRequiresSaving = true;

// Go through all open documents using index-based access to the document collection
// and close each document. Comment out to leave documents open.

for (i = objSpy.Documents.Count; i > 0; i--)
    objSpy.Documents.Item(i).Close(false); // Closes the iterated documents

// ***** Shut down the application or not
// *****

//objSpy.Visible = false; // will shutdown application if it has no more COM
//connections
objSpy.Visible = true; // will keep application running with UI visible

```

Iteration

Im unten aufgelisteten JScript-Code wird gezeigt, wie man durch offene Dokumente iteriert.

Ausführen des Skripts

Sie können dieses Skript testen, indem Sie den unten aufgelisteten Code in eine Datei kopieren, ihr die Dateierweiterung `.js` geben und die Datei entweder über die Befehlszeile oder

den Windows Explorer ausführen. Sie können die Datei auch in den Ordner JScript des Ordners API Examples kopieren:

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/ Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/ Altova/XMLSpy20XX/Examples/API/

Um das Skript auszuführen, starten Sie es über ein Eingabeaufforderungsfenster oder über den Windows Explorer.

Skript-Code

Der nachstehende JScript-Code enthält Erklärungen im Code.

```
// Initialize application's COM object. This will start a new instance of the
// application and
// return its main COM object. Depending on COM settings, a the main COM object of an
// already
// running application might be returned.
try
{
    objSpy = WScript.GetObject("", "XMLSpy.Application");
}
catch(err)
{
    Exit("Can't access or create XMLSpy.Application");
}

// if newly started, the application will start without its UI visible. Set it to
// visible.
objSpy.Visible = true;

// ***** Accessing documents, setting the active document
// *****

// Locate examples via USERPROFILE shell variable. The path needs to be adapted to major
// release versions.
objWshShell = WScript.CreateObject("WScript.Shell");
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\My
Documents\\Altova\\XMLSpy2012\\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();

// ***** Two ways of iterating; close documents
// *****

// Go through all open documents using a JScript Enumerator
bRequiresSaving = false;
for (var iterDocs = new Enumerator(objSpy.Documents); !iterDocs.atEnd(); iterDocs.
moveNext())
```

```

        if (iterDocs.item().IsModified)
            bRequiresSaving = true;

// Go through all open documents using index-based access to the document collection
// and close each document. Comment out to leave documents open.

for (i = objSpy.Documents.Count; i > 0; i--)
    objSpy.Documents.Item(i).Close(false); // Closes the iterated documents

// ***** Shut down the application or not
// *****

//objSpy.Visible = false;           // will shutdown application if it has no more COM
connections                        // connections
objSpy.Visible = true;             // will keep application running with UI visible

```

Fehlerbehandlung

Die Applikations-API gibt Fehler auf zwei verschiedene Arten zurück:

- das von jeder API-Methode zurückgegebene `HRESULT`
- die `Info`-Schnittstelle der Applikations-API

Jede API-Methode gibt ein `HRESULT` zurück. Dieser 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 `Info` Schnittstelle, die ebenfalls von der Applikations-API unterstützt wird. Bei Auftreten eines Fehlers erstellt die API ein neues Objekt, das die `Info` Schnittstelle implementiert. Die aus der `Info` Schnittstelle stammenden Informationen werden in die Entwicklungsumgebung in ihren eigenen Fehlerbehandlungsmechanismus importiert.

JScript-Fehlerbehandlung

JScript bietet einen try-catch-Mechanismus zur Behandlung von Fehlern, die durch COM-Aufrufe ausgelöst wurden. Es wird ein Fehlerobjekt deklariert, das die erforderlichen Informationen enthält. In unten gezeigten Code sehen Sie, wie dabei vorgegangen wird.

```

// go through all open documents using a JScript Enumerator
bRequiresSaving = false;
for (var iterDocs = new Enumerator(objSpy.Documents); !iterDocs.atEnd(); iterDocs.moveNext())
{
    if (iterDocs.item().IsModified)
        bRequiresSaving = true;

    var strErrorText = new Array(1);
    var nErrorNumber = new Array(1);
    var errorData = new Array(1);

    if (!iterDocs.item().IsValid(strErrorText, nErrorNumber, errorData))
    {
        var text = 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.");
    }
}

```

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
function DocEvent_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;
objDoc = objSpy.Documents.OpenFile ("C:\\Program
Files\\Altova\\XMLSpy2012\\Examples\\OrgChart.xml", 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();

```

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](#) Objekt platziert. Die restlichen Funktionen können über die [Document](#) 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](#).

Beschreibungen

In den Unterabschnitten dieses Abschnitts werden die einzelnen Operationen ausführlich beschrieben.

- [Import aus einer Datenbank](#)
- [Export in eine Datenbank](#)
- [Import aus einer Textdatei](#)
- [Export in eine Textdatei](#)

Ausführen des Skripts

Der unten aufgelistete JScript-Code enthält Codefragmente zum Importieren und Exportieren in Datenbanken und Text. Der Code steht in der Datei `ImportExport.js` im Ordner `JScript` des Ordners `API Examples` zur Verfügung:

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/Altova/XMLSpy20XX/Examples/API/

Um das Skript auszuführen, starten Sie es über ein Eingabeaufforderungsfenster oder über den Windows Explorer.

Skript-Code

Der nachstehende JScript-Code enthält Codefragmente für den Import und Export in Datenbanken und Text.

```
// Initialize application's COM object. This will start a new instance of the
application and
// return its main COM object. Depending on COM settings, a the main COM object of an
already
```

```

// running application might be returned.
try
{
    objSpy = WScript.GetObject("", "XMLSpy.Application");
}
catch(err)
{
    Exit("Can't access or create XMLSpy.Application");
}

// if newly started, the application will start without its UI visible. Set it to
visible.
objSpy.Visible = true;

// ***** Import from database: Start *****

// Locate examples via USERPROFILE shell variable. The path needs to be adapted to major
release versions.
objWshShell = WScript.CreateObject("WScript.Shell");
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\My
Documents\\Altova\\XMLSpy2012\\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);
}

// ***** Import from database: End *****

// ***** Export to database: Start *****

//try
//{
    // 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);
    }
}

```

```

}

// ***** Export to database: End *****

// ***** Import from text: Start *****

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);
}

// ***** Import from text: End *****

// ***** Export to text: Start *****

//try
//{
    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);
}

// ***** Export to text: End *****

//objSpy.Visible = false;           // will shutdown application if it has no more COM
connections
objSpy.Visible = true; // will keep application running with UI visible

```

Dies sind die Schritte, um für den Import eine Verbindung zu einer vorhandenen Datenbank herzustellen:

1. Verwenden Sie ein [DatabaseConnection](#) Objekt und definieren Sie die Eigenschaften: Die Methode [Application.GetDatabaseSettings](#) 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](#) 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 entsprechende Spalte nicht importiert. Zusätzlich dazu können Sie die Eigenschaft [ElementListItem.ElementKind](#) 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](#):

```
objImpDocFromDB =  
objSpy.ImportFromDatabase(objImpSettings,objElementList);
```

Ausführen des Skripts

Sie können den unten aufgelistete JScript-Code testen, indem Sie ihn in eine Datei kopieren, ihr die Erweiterung `.js` geben und die Datei entweder über die Befehlszeile oder den Windows Explorer ausführen. Sie könnten die Datei in den Ordner `JScript` des Ordners `API Examples` kopieren:

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/ Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/ Altova/XMLSpy20XX/Examples/API/

Um das Skript auszuführen, starten Sie es über ein Eingabeaufforderungsfenster oder über den Windows Explorer.

Skript-Code

Im nachstehende Codefragment wird gezeigt, wie man Daten aus einer Datenbank importieren kann.

```

// Initialize application's COM object. This will start a new instance of the
// application and
// return its main COM object. Depending on COM settings, a the main COM object of an
// already
// running application might be returned.
try
{
    objSpy = WScript.GetObject("", "XMLSpy.Application");
}
catch(err)
{
    Exit("Can't access or create XMLSpy.Application");
}

// if newly started, the application will start without its UI visible. Set it to
// visible.
objSpy.Visible = true;

// ***** Import from database *****

// Locate examples via USERPROFILE shell variable. The path needs to be adapted to major
// release versions.
objWshShell = WScript.CreateObject("WScript.Shell");
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\My
Documents\\Altova\\XMLSpy2012\\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);
}

// ***** End *****

//objSpy.Visible = false; // will shutdown application if it has no more COM
//connections
objSpy.Visible = true; // will keep application running with UI visible

```

Um Daten in eine Datenbank zu exportieren, gehen Sie folgendermaßen vor:

1. Verwenden Sie ein [DatabaseConnection](#) 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](#) 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](#)" auf, um ein ExportSettings Objekt aufzurufen:

```
objExpSettings = objSpy.GetExportSettings()
```

- Erstellen Sie mit [Document.GetExportElementList](#) eine Elementliste. Mit Hilfe der Elementliste können Sie XML-Elemente aus dem Exportprozess ausnehmen. Außerdem enthält diese Liste in den Eigenschaften `RecordCount` und `FieldCount` Informationen über die Anzahl der Datensätze und Felder. Setzen Sie die [ExportSettings.ElementList](#) Eigenschaft auf diese Sammlung. Sie können die Elementliste auf `null/Nothing` (Standardeinstellung) setzen, um alle Elemente zu exportieren.
- Rufen Sie [Document.ExportToDatabase](#) auf, um den Export auszuführen. Die Beschreibung der `ExportToDatabase` Methode enthält auch ein Codebeispiel für einen Datenbankexport.

Ausführen des Skripts

Sie können den unten aufgelistete JScript-Code testen, indem Sie ihn in eine Datei kopieren, ihr die Erweiterung `.js` geben und die Datei entweder über die Befehlszeile oder den Windows Explorer ausführen. Sie könnten die Datei in den Ordner `JScript` des Ordners `API Examples` kopieren:

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/Altova/XMLSpy20XX/Examples/API/

Um das Skript auszuführen, starten Sie es über ein Eingabeaufforderungsfenster oder über den Windows Explorer.

Skript-Code

Im nachstehende Codefragment wird gezeigt, wie man Daten in eine Datenbank exportieren kann.

```
// Initialize application's COM object. This will start a new instance of the
// application and
// return its main COM object. Depending on COM settings, a the main COM object of an
// already
// running application might be returned.
try
{
    objSpy = WScript.GetObject("", "XMLSpy.Application");
}
catch(err)
{
    Exit("Can't access or create XMLSpy.Application");
}

// if newly started, the application will start without its UI visible. Set it to
// visible.
```

```

objSpy.Visible = true;

// ***** Export to database
// *****

//try
//{
    // 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);
    }
}

// ***** End *****

//objSpy.Visible = false;           // will shutdown application if it has no more COM
connections
objSpy.Visible = true; // will keep application running with UI visible

```

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](#) Objekt und definieren Sie die Eigenschaften:
Die Methode [Application.GetTextImportExportSettings](#) gibt ein neues Objekt zurück, das eine Textdatei für den Import definiert.

```
objImpSettings = objSpy.GetTextImportExportSettings();
```

Sie müssen zumindestens für die Eigenschaft `ImportFile` den 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";
```

Rufen Sie [Application.GetTextImportElementList](#) auf, um eine Sammlung aller Spalten in der Textdatei zu erhalten:

```
objElementList = objSpy.GetTextImportElementList(objImpSettings);
```

3. Starten Sie den Import mit [Application.ImportFromText](#).

```
objImpDocFromText = objSpy.ImportFromText(objImpSettings,objElementList
);
```

Ausführen des Skripts

Sie können den unten aufgelistete JScript-Code testen, indem Sie ihn in eine Datei kopieren, ihr die Erweiterung .js geben und die Datei entweder über die Befehlszeile oder den Windows Explorer ausführen. Sie könnten die Datei in den Ordner `JScript` des Ordners `API Examples` kopieren:

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/ Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/ Altova/XMLSpy20XX/Examples/API/

Um das Skript auszuführen, starten Sie es über ein Eingabeaufforderungsfenster oder über den Windows Explorer.

Skript-Code

Im nachstehende Codefragment wird gezeigt, wie man Daten aus einer Textdatei importieren kann.

```
// Initialize application's COM object. This will start a new instance of the
// application and
// return its main COM object. Depending on COM settings, a the main COM object of an
// already
// running application might be returned.
try
{
    objSpy = WScript.GetObject("", "XMLSpy.Application");
}
catch(err)
{
    Exit("Can't access or create XMLSpy.Application");
}

// if newly started, the application will start without its UI visible. Set it to
// visible.
objSpy.Visible = true;

// ***** Import from text *****

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\n" +
        "Description: " + err.description);
}

// ***** End *****
```



```
//objSpy.Visible = false;           // will shutdown application if it has no more COM
connections
objSpy.Visible = true; // will keep application running with UI visible
```

Um Daten in eine Textdatei zu exportieren, gehen Sie folgendermaßen vor:

1. Verwenden Sie ein [TextImportExportSettings](#) Objekt und definieren Sie die erforderlichen Eigenschaften.
2. Füllen Sie ein [ExportSettings](#) Objekt mit den erforderlichen Werten. Siehe Punkt 2 aus [Export in eine Datenbank](#).
3. Erstellen Sie eine Elementliste mit [Document.ExportElementList](#). Siehe Punkt 3 aus [Export in eine Datenbank](#).
4. Rufen Sie [Document.ExportToText](#) auf, um den Export auszuführen.

Ausführen des Skripts

Sie können den unten aufgelistete JScript-Code testen, indem Sie ihn in eine Datei kopieren, ihr die Erweiterung .js geben und die Datei entweder über die Befehlszeile oder den Windows Explorer ausführen. Sie könnten die Datei in den Ordner JScript des Ordners API Examples kopieren:

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/ Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/ Altova/XMLSpy20XX/Examples/API/

Um das Skript auszuführen, starten Sie es über ein Eingabeaufforderungsfenster oder über den Windows Explorer.

Skript-Code

Im nachstehende Codefragment wird gezeigt, wie man Daten in eine Textdatei exportieren kann.

```
// Initialize application's COM object. This will start a new instance of the
application and
// return its main COM object. Depending on COM settings, a the main COM object of an
already
// running application might be returned.
try
{
    objSpy = WScript.GetObject("", "XMLSpy.Application");
}
catch(err)
{
    Exit("Can't access or create XMLSpy.Application");
}

// if newly started, the application will start without its UI visible. Set it to
visible.
objSpy.Visible = true;

// ***** Export to text *****

//try
//{
    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);
}

// ***** End *****

//objSpy.Visible = false;           // will shutdown application if it has no more COM
connections
objSpy.Visible = true; // will keep application running with UI visible

```

Beispiel: Bubble-Sortierung dynamischer Tabellen

Mit dem folgenden JScript-Fragment wird jede dynamische Tabelle nach der Tabellenspalte sortiert, in der sich der Cursor gerade befindet. Die Sortierung erfolgt auf dem Bildschirm. Für alle ausgeführten Operationen steht der Undo-Puffer zur Verfügung.

Wenn Sie JScript auf Ihrem Computer ausführen können, was höchstwahrscheinlich der Fall ist, kopieren Sie den folgenden Code in eine Datei mit der Erweiterung ".js". Führen Sie das Skript aus, indem Sie im Windows Explorer darauf doppelklicken oder führen Sie es über die Befehlszeile aus.

```

// some useful XMLSpy enum constants
var spyAuthenticTag = 6;
var spyAuthenticTable = 9;
var spyAuthenticTableRow = 10;
var spyAuthenticTableColumn = 11;

var spyAuthenticRangeBegin = 2;

// example call for the sort table function
try
{
    var objSpy = WScript.GetObject("", "XMLSpy.Application");
    // If only Authentic is installed (and XMLSpy is not installed) use:
    // var objSpy = WScript.GetObject("", "AuthenticDesktop.Application")

    // use current selection to indicate which column to sort
    SortCurrentTable (objSpy.ActiveDocument.AuthenticView.Selection);
}
catch (err)
{
    WScript.Echo ("Please open a document in authentic view, and select a
        table column\n" +
        "Error : (" + (err.number & 0xffff) + ") " +
err.description); }

// we assume that XMLSpy is running, a document with a dynamic table
// is open, and the cursor is in a table column that will
// be used for sorting.
function SortCurrentTable (objCursor)

```

```

{
    if ( objCursor.IsInDynamicTable() )
    {
        // calculate current column index
        var nColIndex = 0;
        while ( true )
        {
            // go left column-by-column
            try { objCursor.GotoPrevious(spyAuthenticTableColumn); }
            catch (err) { break; }
            nColIndex++;
        }

        // count number of table rows, so the bubble loops become simpler.
        // goto begin of table
        var objTableStart =
        objCursor.ExpandTo(spyAuthenticTable).CollapsToBegin().Clone();
        var nRows = 1;
        while ( true )
        {
            // go down row-by-row
            try { objTableStart.GotoNext(spyAuthenticTableRow); }
            catch (err) { break; }
            nRows++;
        }

        // bubble sort through table
        for ( var i = 0; i < nRows - 1; i++)
        {
            // select correct column in first table row
            var objBubble =
            objCursor.ExpandTo(spyAuthenticTable).CollapsToBegin().Clone();
            objBubble.Goto ( spyAuthenticTableColumn, nColIndex,
            spyAuthenticRangeBegin).ExpandTo( spyAuthenticTag);

            // bubble this row down as far as necessary
            for ( var j = 0; j < nRows - i - 1; j++)
            {
                var strField1 = objBubble.Text;
                // now look for the comparison table cell: start of next row and right
                // of the correct column
                var strField2 = objBubble.GotoNext(spyAuthenticTableRow).
                    Goto ( spyAuthenticTableColumn, nColIndex,
                    spyAuthenticRangeBegin).
                    ExpandTo( spyAuthenticTag).Text;
                if ( strField1 > strField2 )
                {
                    objBubble.MoveRowUp(); // swap the rows
                    // and re-calculate objBubble to select the cell to bubble
                    objBubble.GotoNext(spyAuthenticTableRow).
                        Goto ( spyAuthenticTableColumn, nColIndex,
                        spyAuthenticRangeBegin).
                        ExpandTo( spyAuthenticTag);
                }
            }
        }
    }
    else
        WScript.Echo ("please, select a table cell first");
}

```

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](#) sowie ein [Beispiel](#).

Informationen über andere Funktionalitäten finden Sie in den unten aufgeführten JScript-Beispielen:

- [Starten der Applikation oder Anhängen einer laufenden Instanz](#)
- [Einfacher Dokumentaufruf](#)
- [Iteration](#)
- [Fehlerbehandlung](#)
- [Events](#)
- [Import und Export von Daten](#)

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.

```
' the event handler function
Function DocEvent_OnBeforeCloseDocument(objDocument)
    Call WScript.Echo("received event - before closing document")
End Function

' 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")

' create document object and connect to its events
objSpy.Visible = True
Set objDoc = objSpy.Documents.OpenFile ("C:\\Program
Files\\Altova\\XMLSPY2004\\Examples\\OrgChart.xml", False)
Call WScript.ConnectObject(objDoc, "DocEvent_")

' keep running while waiting on the event
' in the meantime close the document in XMLSPY manually
Call WScript.Echo ("sleeping for 10 seconds ...")
Call WScript.Sleep (10000)

Set objDoc = Nothing
Call WScript.Echo ("stopped listening for event")
Call objSpy.Quit
```

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.

```
Rem
-----
----
Rem VBScript example that demonstrates how to use events.
Rem
-----
----

' Event handler for OnSelectionChanged event of AuthenticView
Function AuthenticViewEvent_OnSelectionChanged( objAuthenticRange)
    If objAuthenticRange.FirstTextPosition <>
objAuthenticRange.LastTextPosition Then
        Call WScript.Echo( "Selection: " & objAuthenticRange.Text &
vbNewLine & vbNewLine & "Close this dialog.")
    Else
        Call WScript.Echo( "Cursor position: " &
objAuthenticRange.FirstTextPosition & vbNewLine & vbNewLine & "Close
this dialog.")
    End If
End Function

' Here starts the main code.
' Find out user's personal folder and locate one of the installed
XMLSpy 2011 examples.
Set WshShell = WScript.CreateObject( "WScript.Shell")
personalFolder = WshShell.ExpandEnvironmentStrings( "%UserProfile%")
xmlspyExamplesFolder = personalFolder & "\My
Documents\Altova\XMLSpy2011\Examples\"
docPath = xmlspyExamplesFolder & "OrgChart.xml"

' Start/access XMLSpy and connect to its automation interface.
Set objSpy = GetObject( "", "XMLSpy.Application")
' Make the UI of XMLSpy visible.
objSpy.Visible = True

' Create object to access windows file system and test if the our
document exists.
Set fso = CreateObject( "Scripting.FileSystemObject")
If fso.FileExists(docPath) Then
    ' open the document
    Call objSpy.Documents.OpenFile( docPath, False)
    set objDoc = 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.
    set objView = objDoc.AuthenticView
    Call WScript.ConnectObject( objView, "AuthenticViewEvent_")
    Call WScript.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.
set objView = Nothing
Else
Call WScript.Echo("The file " & docPath & " does not exist.")
End If

' shut down XMLSpy when this script ends
objSpy.Visible = False

```

C#

Sie können mit Hilfe von C# auf die Funktionalitäten der Application API zugreifen. Zur Erstellung des C#-Code können Sie Visual Studio 2008 oder Visual Studio 2010 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, in dem eine Liste der installierten COM-Komponenten angezeigt wird. Wählen Sie die XMLSpy Typbibliothekkomponente aus der Liste aus, um sie hinzuzufügen.
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 C#-Ordner des Ordners API Examples finden:

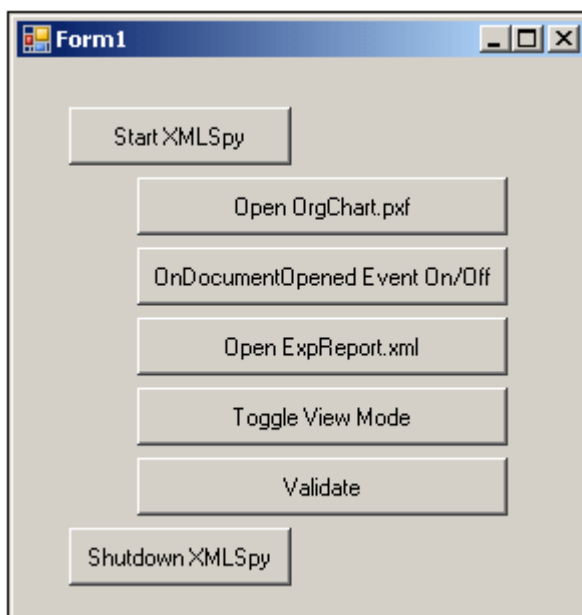
Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/ Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/ Altova/XMLSpy20XX/Examples/API/

Sie können das Projekt innerhalb von Visual Studio 2008 oder Visual Studio 2010 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.

Funktionen im Codebeispiel unten

In diesem Beispiel sehen Sie eine einfache Benutzeroberfläche (*Abbildung unten*) mit Schaltflächen, über die grundlegende XMLSpy-Operationen aufgerufen werden:



- [Start XMLSpy](#): Startet XMLSpy, das als Automation Server registriert ist, oder aktiviert das Programm, wenn XMLSpy bereits ausgeführt wird.
- [Open OrgChart.pxf](#): 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](#): Zeigt, wie die Applikation auf XMLSpy Events hört. Ist die Funktion aktiv, erscheint ein Meldungsfield, nachdem ein Dokument geöffnet wurde.
- [Open ExpReport.xml](#): Öffnet ein weiteres Beispieldokument.
- [Toggle View Mode](#): 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](#): Validiert das aktive Dokument und zeigt das Ergebnis in einem Meldungsfield an. Im Code wird gezeigt, wie Fehler und COM-Ausgabeparameter behandelt werden.
- [Shutdown XMLSpy](#): 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` (Visual Studio 2008) bzw. auf die Datei `AutomateXMLSpy_VS2010.sln` (Visual Studio 2010). 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
                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 );

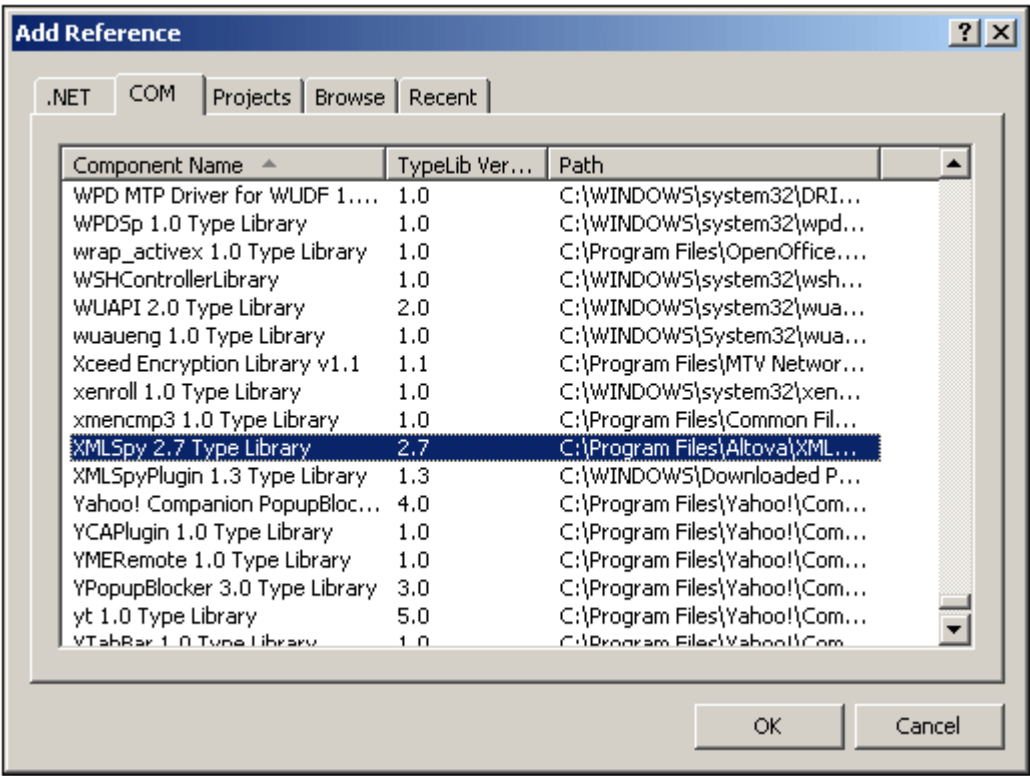
            bEventHandlerIsRegistered = !bEventHandlerIsRegistered;
        }
    }
}

```

Hinzufügen einer Referenz zur Application API

Um die Typbibliothek der Applikations-API als Referenz in einem .NET-Projekt hinzuzufügen, gehen Sie folgendermaßen vor:

1. Klicken Sie, während das .NET-Projekt geöffnet ist, auf den Befehl **Projekt | Verweis hinzufügen**. Daraufhin wird das Dialogfeld "Verweis hinzufügen" (*Abbildung unten*) angezeigt, in dem Sie eine Liste der installierten COM-Komponenten sehen.



- 2. Wählen Sie in der Komponentenliste den Eintrag XMLSpy X.Y Type Library aus und klicken Sie auf OK.

Deklarieren Sie eine Variable zum Aufrufen der XMLSpy API

```
// An instance of XMLSpy is accessed via its automation interface.  
XMLSpyLib.Application XMLSpy;
```

Starten und Beenden der Applikation

In den nachstehenden Codefragmenten wurden den Schaltflächen im Beispiel [AutomateXMLSpy](#) die Methoden `StartXMLSpy_Click` und `ShutdownXMLSpy_Click` zugewiesen. Sie starten bzw. beenden die Applikation. Dieses Beispiel befindet sich im Ordner C# des Ordners API Examples (siehe Datei *Form1.cs*):

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/ Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/ Altova/XMLSpy20XX/Examples/API/

Sie können das Projekt von Visual Studio 2008 oder Visual Studio 2010 aus kompilieren und ausführen.

Starten von XMLSpy

Im folgenden Codefragment aus dem Beispiel [AutomateXMLSpy](#) 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](#) 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;
        }
    }
}
```

Öffnen von Dokumenten

In den unten stehenden Codefragmenten (aus dem Beispiel [AutomateXMLSpy](#)) 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\(string, boolean\)](#).

Das [AutomateXMLSpy Beispiel](#) (siehe Datei *form1.cs*) befindet sich im Ordner C# des Ordners API Examples:

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/Altova/XMLSpy20XX/Examples/API/

Sie können das Projekt von Visual Studio 2008 oder Visual Studio 2010 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.

Iterieren durch offene Dokumente

Im unten gezeigten Codefragment (aus dem Beispiel [AutomateXMLSpy](#); 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](#) befindet sich im Ordner C# des Ordners API Examples:

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/ Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/ Altova/XMLSpy20XX/Examples/API/

Sie können das Projekt von Visual Studio 2008 oder Visual Studio 2010 aus kompilieren und ausführen.

Fehler und COM-Ausgabeparameter

Im unten gezeigten Codefragment (aus dem Beispiel [AutomateXMLSpy](#)) wird gezeigt, wie man Fehler und COM-Ausgabeparameter behandelt. In der Methode [XMLSpy.ActiveDocument.IsValid\(ref strErrorText, ref nErrorNumber, ref errorData\)](#) werden Ausgabeparameter verwendet, mit Hilfe derer im unten gezeigten Codefragment der Text einer Fehlermeldung generiert wird.

Das [AutomateXMLSpy Beispiel](#) (siehe Datei *form1.cs*) befindet sich im Ordner C# des Ordners API Examples:

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/ Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/ Altova/XMLSpy20XX/Examples/API/

Sie können das Projekt von Visual Studio 2008 oder Visual Studio 2010 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
            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);
    }
}

```

Events

Im unten gezeigten Codefragment (aus dem Beispiel [AutomateXMLSpy](#)) wird der Code für zwei Event Handler aufgelistet. Das [AutomateXMLSpy Beispiel](#) (siehe Datei *form1.cs*) befindet sich im Ordner C# des Ordners API Examples:

Windows XP	C:/Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/ Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C:/Users/<Benutzername>/Documents/ Altova/XMLSpy20XX/Examples/API/

Sie können das Projekt von Visual Studio 2008 oder Visual Studio 2010 aus kompilieren und ausführen.

```

// 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;
    }
}

```

Java

Sie können mit Hilfe von Java-Code auf die Applikations-API zugreifen. Damit Sie direkt vom Java-Code aus Zugriff auf den XMLSpy Automation Server haben, müssen sich die unten aufgelisteten Bibliotheken im `classpath` befinden. Sie sind im Applikationsordner im Ordner JavaAPI installiert.

- AltovaAutomation.dll: ein JNI Wrapper für Altova Automation Server
- 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](#).

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.
- **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 <code>SetEncoding</code>	<code>setFileEncoding</code>
AuthenticView, Methode <code>Goto</code>	<code>gotoElement</code>
AuthenticRange, Methode <code>Goto</code>	<code>gotoElement</code>
AuthenticRange, Methode <code>Clone</code>	<code>cloneRange</code>

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](#)
- [Starten und Beenden der Applikation](#)
- [Einfacher Dokumentaufruf](#)
- [Iterationen](#)
- [Verwendung von Out-Parametern](#)
- [Event Handler](#)

Java-Beispielprojekt

Das XMLSpy-Installationspaket enthält ein Java-Beispielprojekt, das Sie im Java-Ordner des Ordners API Examples finden:

Windows XP	C: /Dokumente und Einstellungen/<Benutzername>/Eigene Dateien/ Altova/XMLSpy20XX/Examples/API/
Windows Vista, Windows 7	C: /Users/<Benutzername>/Documents/ Altova/XMLSpy20XX/Examples/API/

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:

<code>AltovaAutomation.dll</code>	Java-COM Bridge: DLL-Teil
<code>AltovaAutomation.jar</code>	Java-COM Bridge: Java-Bibliotheksteil
<code>XMLSpyAPI.jar</code>	Java-Klassen der XMLSpy API
<code>RunXMLSpy.java</code>	Java-Beispielquellcode
<code>BuildAndRun.bat</code>	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.
<code>.classpath</code>	Hilfdatei Eclipse-Projekt
<code>.project</code>	Eclipse-Projektdatei
<code>XMLSpyAPI_JavaDoc.zip</code>	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](#): Startet XMLSpy, das als Automation Server registriert ist, bzw. aktiviert das Programm, wenn XMLSpy bereits ausgeführt wird.
- [OrgChart.pxf öffnen](#): Navigiert zu einem der mit XMLSpy installierten Beispieldokumente und öffnet es.
- [Iterieren durch und Ändern des Ansichtsmodus](#): 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](#): Validiert das aktive Dokument und zeigt das Ergebnis in einem Meldungsfeld an. Im Code wird gezeigt, wie Ausgabeparameter verwendet werden.
- [Event-Behandlung](#): Zeigt, wie XMLSpy Events behandelt werden.
- [XMLSpy beenden](#): 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 Eingabeaufforderungsfenster, 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.

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 DocumentEventsDefaultHandler()
080 {
081     @Override
082     public boolean
083     onBeforeCloseDocument( Document i_ipDoc) throws AutomationException
084     {
085         System.out.println( "Document
086 " + i_ipDoc.getName() + " requested closing.");
087
088         // Allow closing of document
089         return true;
090     }
091 });
092 doc.close( true);
093 doc = null;
094
095 System.out.println( "Watch XMLSpy! ");
096 }
097 catch ( AutomationException e)
098 {
099     // e.printStackTrace();
100 }
101 finally
102 {
103     // Make sure that XMLSpy can shut down properly.
104     if ( xmlSpy != null)
105         xmlSpy.dispose();
106
107     // Since the COM server was made visible and still is visible,
108     // it will keep running, and needs to be closed manually.
109     System.out.println( "Now close XMLSpy! ");
110 }
111 }
112 }
113 }
114 }
115 }

```

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 }
```

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 );
```

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() )
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     }
```

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         validationErrorText.setByRefFlag();
16         JVariant validationErrorCount = new
17
18         JVariant.JIntVariant(0);
19         validationErrorCount.setByRefFlag();
20         JVariant validationErrorXMLData = new
21
22         JVariant.JIDispatchVariant(0);
23         validationErrorXMLData.setByRefFlag();
24         if (!doc.isValid(validationErrorText,
25             validationErrorCount, validationErrorXMLData))
26             System.out.println("Document
27
28             " + doc.getName() + " is not wellformed - " +
29
30             validationErrorText.getStringValue());
31         else
32             System.out.println("Document
33
34             " + doc.getName() + " is wellformed.");
35     }
```

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
04 // class
05 // from DocumentEvents and implement all methods of this interface.
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;

```

3.1.3 DOM und XML Data

Über die `XMLData` Schnittstelle 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 `XMLData` Schnittstelle 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 `XMLData` und 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
- 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;
}

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;
}

```

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. `RowAppend` aufgerufen werden. (`RowAppend` hat 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
<code>IsRowInsertEnabled</code>	RowInsert , wird nun ersetzt durch AuthenticRange.InsertRow	Operation "Zeile einfügen"
<code>IsRowAppendEnabled</code>	RowAppend , wird nun ersetzt durch AuthenticRange.AppendRow	Operation "Zeile anhängen"
<code>IsRowDeleteEnabled</code>	RowDelete , wird nun ersetzt durch AuthenticRange.DeleteRow	Operation "Zeile löschen"

IsRowMoveUpEnabled	RowMoveUp , wird nun ersetzt durch AuthenticRange.MoveRowUp	XML-Daten eine Zeile nach oben verschieben
IsRowMoveDownEnabled	RowMoveDown , wird nun ersetzt durch AuthenticRange.MoveRowDown	XML-Daten eine Zeile nach unten verschieben
IsRowDuplicateEnabled	RowDuplicate , wird nun ersetzt durch AuthenticRange.DuplicateRow	Aktuell ausgewählte Zeile duplizieren

3.2 Schnittstellen

Objekthierarchie

[Application](#)

[SpyProject](#)

[SpyProjectItems](#)

[SpyProjectItem](#)

[Documents](#)

[Document](#)

[GridView](#)

[AuthenticView](#)

[AuthenticRange](#)

[AuthenticDataTransfer](#) (vormals DocEditDataTransfer)

[OldAuthenticView](#) (vormals DocEditView, **wird nicht mehr verwendet**, jetzt ersetzt durch [AuthenticView](#) und [AuthenticRange](#))

[AuthenticSelection](#) (vormals DocEditSelection, **wird nicht mehr verwendet**, jetzt ersetzt durch [AuthenticRange](#))

[AuthenticEvent](#) (vormals DocEditEvent, **wird nicht mehr verwendet**)

[AuthenticDataTransfer](#) (vormals DocEditDataTransfer)

[TextView](#)

[XMLData](#)

[Dialogs](#)

[CodeGeneratorDlg](#)

[FileSelectionDlg](#)

[SchemaDocumentationDlg](#)

[GenerateSampleXMLDlg](#)

[DTDSchemaGeneratorDlg](#)

[FindInFilesDlg](#)

[WSDLDocumentationDlg](#)

[WSDL20DocumentationDlg](#)

[XBRLDocumentationDlg](#)

[DatabaseConnection](#)

[ExportSettings](#)

[TextImportExportSettings](#)

[ElementList](#)

[ElementListItem](#)

[Enumerations](#)

Beschreibung

Dieses Kapitel enthält die Referenz der XMLSpy 1.5 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.

3.2.1 Application

See also

Methoden

[GetDatabaseImportElementList](#)

[GetDatabaseSettings](#)

[GetDatabaseTables](#)

[ImportFromDatabase](#)

[CreateXMLSchemaFromDBStructure](#)

[GetTextImportElementList](#)
[GetTextImportExportSettings](#)
[ImportFromText](#)

[ImportFromWord](#)

[ImportFromSchema](#)

[GetExportSettings](#)

[NewProject](#)
[OpenProject](#)

[AddMacroMenuItem](#)
[ClearMacroMenu](#)

[ShowForm](#)

[ShowApplication](#)

[URLDelete](#)
[URLMakeDirectory](#)

[FindFiles](#)

[Quit](#)

Eigenschaften

[Application](#)
[Parent](#)

[ActiveDocument](#)
[Documents](#)

[CurrentProject](#)

[Dialogs](#)

[WarningNumber](#)
[WarningText](#)

[Status](#)
[MajorVersion](#)
[MinorVersion](#)
[Edition](#)
[IsAPISupported](#)
[ServicePackVersion](#)

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

```
Dim objSpy As Application
Set objSpy = CreateObject("XMLSpy.Application")
```

Events

OnBeforeOpenDocument

Siehe auch

Event: *OnBeforeOpenDocument*(*objDialog* als [FileSelectionDlg](#))

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 [FileSelectionDlgDialogAction](#) von *io_objDialog* auf dem Standardwert [spyDialogOK](#). Um das Öffnen des Dokuments abubrechen, setzen Sie die Eigenschaft auf [spyDialogCancel](#).

Beispiele

Im Folgenden finden Sie Beispiele dafür, wie ein Skript für dieses Event erstellt werden kann.

XMLSpy scripting environment - VBScript:

```
Function On_BeforeOpenDocument(objDialog)
EndFunction
```

XMLSpy scripting environment - JScript:

```
function On_BeforeOpenDocument(objDialog)
{
}
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugin.OnEvent (26, ...) //nEventId=26
```

OnBeforeOpenProject

Siehe auch

Event: *OnBeforeOpenProject*(*objDialog* als [FileSelectionDlg](#))

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 [FileSelectionDlgDialogAction](#) von *io_objDialog* auf dem Standardwert [spyDialogOK](#). Um das Öffnen des Projekts abubrechen, setzen Sie die Eigenschaft auf [spyDialogCancel](#).

Beispiele

Im Folgenden sehen Sie einige Beispiele, wie ein Skript für dieses Event erstellt werden kann.

XMLSpy scripting environment - VBScript:

```
Function On_BeforeOpenProject(objDialog)
EndFunction
```

XMLSpy scripting environment - JScript:

```
function On_BeforeOpenProject(objDialog)
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (25, ...) //nEventId=25
```

OnDocumentOpened

Siehe auch

Event: OnDocumentOpened(objDocument als [Document](#))

Beschreibung

Dieses Event wird ausgelöst, wenn ein Dokument in XMLSpy geöffnet wird. Voraussetzung ist, dass eine Datei über das Dialogfeld OpenFile oder OpenURL geöffnet oder eine neue Datei erstellt oder eine Datei in XMLSpy gezogen 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.

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_OpenDocument(objDocument)
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_OpenDocument(objDocument)
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (7, ...) //nEventId=7
```

OnProjectOpened

Siehe auch

Event: OnProjectOpened(objProject als [SpyProject](#))

Beschreibung

Dieses Event wird ausgelöst, wenn ein Projekt in XMLSpy geöffnet 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)
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_OpenProject(objProject)
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (6, ...) //nEvent=6
```

ActiveDocument

Siehe auch

Eigenschaft: [ActiveDocument](#) als [Document](#)

Beschreibung

Referenz auf das aktive Dokument. Wenn kein Dokument geöffnet ist, ist `ActiveDocument` null (nothing).

Fehler

- 1111 Das application-Objekt ist nicht mehr gültig.
- 1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

AddMacroMenuItem

Siehe auch

Methode: [AddMacroMenuItem](#)([strMacro](#) als String, [strDisplayText](#) als String)

Beschreibung

Fügt einen Menübefehl zum Menü **Extras** hinzu. Dieser neue Menübefehl ruft das von `strMacro` definierte Makro auf. Siehe auch [Aufrufen von Makros](#) von XMLSpy".

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.

Application

Siehe auch

Eigenschaft: [Application](#) als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

- 1111 Das application-Objekt ist nicht mehr gültig.
- 1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

ClearMacroMenu

Siehe auch

Methode: `ClearMacroMenu()`

Rückgabewert

Keiner

Beschreibung

Entfernt alle Menübefehle aus dem Menü **Extras**. Siehe auch Aufrufen von Makros von XMLSpy".

Fehler

1111 Das application-Objekt ist nicht mehr gültig.

CreateXMLSchemaFromDBStructure

Siehe auch

Methode: `CreateXMLSchemaFromDBStructure(pImportSettings als DatabaseConnection, pTables als ElementList)`

Beschreibung

`CreateXMLSchemaFromDBStructure` erstellt anhand der in `pImportSettings` definierten Datenbank für die in `pTables` definierten Tabellen ein neues/neue Schema-Dokument(e), das/die die Struktur der Datenbanktabellen beschreibt/beschreiben.

Der Parameter `pTables` definiert, 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](#).

Fehler

1112 Es wurde eine ungültige Datenbank definiert.

1120 Der Datenbankimport ist fehlgeschlagen.

CurrentProject

Siehe auch

Eigenschaft: `CurrentProject` als [SpyProject](#)

Beschreibung

Referenz auf das aktive Dokument. Wenn kein Projekt geöffnet ist, ist `CurrentProject` gleich Null (nothing).

Fehler

1111 Das application-Objekt ist nicht mehr gültig.

1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Dialogs

Siehe auch

Eigenschaft: [Dialogs](#) als [Dialogs](#) (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.

Documents

Siehe auch

Eigenschaft: [Documents](#) als [Documents](#)

Beschreibung

Sammlung aller offenen Dokumente. Siehe auch Einfacher Dokumentzugriff.

Fehler

- 1111 Das application-Objekt ist nicht mehr gültig.
- 1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Edition

Siehe auch

Eigenschaft: [Edition](#) als String

Beschreibung

Gibt die Edition der Applikation, also z.B. Enterprise, Professional oder Basic zurück.

Fehler

- 1111 Das application-Objekt ist nicht mehr gültig.
- 1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

FindInFiles

Siehe auch

Method: [FindInFiles](#)([pSettings](#) als [FindInFilesDlg](#)) als [FindInFilesResults](#)

Beschreibung

Gibt ein [FindInFilesResults](#) 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.

GetDatabaseImportElementList

Siehe auch

Methode: `GetDatabaseImportElementList(pImportSettings als DatabaseConnection)` als [ElementList](#)

Beschreibung

Die Funktion gibt eine Sammlung von [ElementListItems](#) zurück, wobei die [ElementListItem.Name](#) Eigenschaften die Namen der Felder enthalten, die für den Import ausgewählt werden können und die [ElementListItem.ElementKind](#) Eigenschaften, je nachdem, welcher Wert in [DatabaseConnection.AsAttributes](#) übergeben wird, entweder auf `spyXMLDataAttr` oder `spyXMLDataElement` initialisiert werden. Diese Liste dient als Filter dafür, was schließlich durch einen zukünftigen Aufruf von [ImportFromDatabase](#) importiert wird. Verwenden Sie [ElementList.RemoveElement](#) um Felder vom Export auszuschließen.

Für die Herstellung der Verbindung zur Datenbank muss eines der Objekte [DatabaseConnection.File](#), [DatabaseConnection.ADOConnection](#) oder [DatabaseConnection.ODBCConnection](#) sowie [DatabaseConnection.SQLSelect](#) ausgefüllt werden. Verwenden Sie die Eigenschaft [DatabaseConnection.AsAttributes](#) um [ElementListItem.ElementKind](#) der erzeugten Elementliste entweder auf `spyXMLDataAttr` oder `spyXMLDataElement` zu initialisieren.

Beispiel

Siehe Beispiel unter [ImportFromDatabase](#).

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.

GetDatabaseSettings

Siehe auch

Methode: `GetDatabaseSettings()` als [DatabaseConnection](#)

Beschreibung

`GetDatabaseSettings` erstellt ein neues Objekt für Datenbankeinstellungen. Mit Hilfe des Objekts werden die Datenbankverbindungsparameter für die Methoden [GetDatabaseTables](#), [GetDatabaseImportElementList](#), [ImportFromDatabase](#), [ImportFromSchema](#) und [ExportToDatabase](#) definiert.

Beispiel

Siehe Beispiel von [ImportFromDatabase](#).

Fehler

- 1111 Das application-Objekt ist nicht mehr gültig.
- 1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

GetDatabaseTables

Siehe auch

Methode: `GetDatabaseTables(pImportSettings als DatabaseConnection)` als [ElementList](#)

Beschreibung

`GetDatabaseTables` liest die Tabellennamen aus der in *pImportSettings* definierten Datenbank. Für die Datenbankverbindung muss eine der folgenden Eigenschaften ausgefüllt werden [DatabaseConnection.File](#), [DatabaseConnection.ADOConnection](#) oder [DatabaseConnection.ODBCConnection](#). Alle anderen Eigenschaften werden ignoriert. Die Funktion gibt eine Sammlung von `ElementListItem`s zurück, wobei die [ElementListItem.Name](#) Eigenschaften die Namen der in der angegebenen Datenbank gespeicherten Tabellen enthält. Die restlichen Eigenschaften von [ElementListItem](#) 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

```
Dim objImpSettings As DatabaseConnection
Set objImpSettings = objSpy.GetDatabaseSettings
objImpSettings.ADOConnection = TxtADO.Text

'store table names in list box
ListTables.Clear

Dim objList As ElementList
Dim objItem As ElementListItem
On Error GoTo ErrorHandler
Set objList = objSpy.GetDatabaseTables(objImpSettings)

For Each objItem In objList
    ListTables.AddItem objItem.Name
Next
```

GetExportSettings

Siehe auch

Methode: `GetExportSettings()` als [ExportSettings](#) (schreibgeschützt)

Beschreibung

`GetExportSettings` erstellt 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](#) und die Beispiele unter [Import und Export](#).

Fehler

- 1111 Das application-Objekt ist nicht mehr gültig.
- 1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

GetTextImportElementList

Siehe auch

Methode: `GetTextImportElementList(pImportSettings als TextImportExportSettings)` als `ElementList`

Beschreibung

`GetTextImportElementList` ruft Importinformationen über die in `pImportSettings` definierte Textdatei ab. Die Funktion gibt eine Sammlung von `ElementListItems` zurück, wobei die `ElementListItem.Name` 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` auf `False`. Die erzeugte Elementliste enthält allgemeine Spaltennamen wie 'Field1' usw.

Siehe auch [Import und Export von Daten](#).

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
' -----
Dim objImpSettings As TextImportExportSettings
Set objImpSettings = objSpy.GetTextImportExportSettings

objImpSettings.ImportFile = "C:\ImportMe.txt"
objImpSettings.HeaderRow = False

Dim objList As ElementList
Set objList = objSpy.GetTextImportElementList(objImpSettings)

'exclude first column
objList.RemoveItem 1

Dim objImpDoc As Document
On Error Resume Next
Set objImpDoc = objSpy.ImportFromText(objImpSettings, objList)
CheckForError
```

GetTextImportExportSettings

Siehe auch

Methode: `GetTextImportExportSettings()` als `TextImportExportSettings` (schreibgeschützt)

Beschreibung

`GetTextImportExportSettings` erstellt ein neues Objekt der allgemeinen Import- und Exporteinstellungen für Textdateien. Siehe auch Beispiel zu [Application.GetTextImportElementList](#) und [Import und Export](#).

Siehe auch [Import und Export von Daten](#).

Fehler

- 1111 Das application-Objekt ist nicht mehr gültig.
- 1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

ImportFromDatabase

Siehe auch

Methode: `ImportFromDatabase(pImportSettings als DatabaseConnection
pElementList als ElementList) als Document`

Rückgabewert

Erstellt ein neues Dokument, das die aus der Datenbank importierten Daten enthält.

Beschreibung

`ImportFromDatabase` importiert Daten aus einer Datenbank, wie in `pImportSettings` definiert und erstellt ein neues Dokument, das diese Daten enthält. Eine der folgenden Eigenschaften muss dazu ausgefüllt werden: [DatabaseConnection.File](#), [DatabaseConnection.ADOConnection](#) oder [DatabaseConnection.ODBCConnection](#) und [DatabaseConnection.SQLSelect](#). Zusätzlich dazu können Sie mittels [DatabaseConnection.AsAttributes](#), [DatabaseConnection.ExcludeKeys](#), [DatabaseConnection.IncludeEmptyElements](#) und [NumberDateTimeFormat](#) weitere Parameter für den Import eingeben.

Der Parameter `pElementList` definiert, 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](#) und [GetDatabaseImportElementList](#) 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

```
Dim objImpSettings As DatabaseConnection
Set objImpSettings = objSpy.GetDatabaseSettings

objImpSettings.ADOConnection = strADOConnection
objImpSettings.SQLSelect = "SELECT * FROM MyTable"

Dim objDoc As Document
On Error Resume Next
Set objDoc = objSpy.ImportFromDatabase(objImpSettings,
```

```
objSpy.GetDatabaseImportElementList( objImpSettings))
' CheckForError here
```

ImportFromSchema

Siehe auch

Methode: `ImportFromSchema(pImportSettings als DatabaseConnection, strTable als String, pSchemaDoc als Document)` als [Document](#)

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 `pImportSettings` definierten Datenbank. Dazu muss eine der folgenden Eigenschaften ausgefüllt werden [DatabaseConnection.File](#), [DatabaseConnection.ADOConnection](#) oder [DatabaseConnection.ODBCConnection](#). Zusätzlich dazu können Sie mittels [DatabaseConnection.AsAttributes](#), [DatabaseConnection.ExcludeKeys](#) und [NumberDateTimeFormat](#) 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](#) und [GetDatabaseTables](#) 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.

ImportFromText

Siehe auch

Methode: `ImportFromText(pImportSettings als TextImportExportSettings, pElementList als ElementList)` als [Document](#)

Beschreibung

`ImportFromText` importiert die Textdatei, wie in `pImportSettings` definiert. Der Parameter `pElementList` kann als Importfilter verwendet werden. Übergeben Sie entweder die durch einen vorherigen Aufruf von [GetTextImportElementList](#) zurückgegebene Liste oder den Wert `nil` um alle Spalten zu importieren. Um den Import unnötiger Spalten zu vermeiden,

entfernen Sie die entsprechenden Feldnamen mit Hilfe von [ElementList.RemoveElement](#) aus `ElementList` bevor Sie `ImportFromText` aufrufen.

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](#)

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
' -----

Dim objImpSettings As TextImportExportSettings
Set objImpSettings = objSpy.GetTextImportExportSettings

objImpSettings.ImportFile = strFileName
objImpSettings.HeaderRow = False

Dim objImpDoc As Document
On Error Resume Next
Set objImpDoc = objSpy.ImportFromText(objImpSettings,
    objSpy.GetTextImportElementList(objImpSettings))

CheckForError

```

ImportFromWord

Siehe auch

Methode: `ImportFromWord(strFile als String)` als [Document](#)

Beschreibung

`ImportFromWord` importiert das MS Word-Dokument `strFile` in 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.

IsAPISupported

Siehe auch

Eigenschaft: `IsAPISupported` als 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.

MajorVersion

See also

Eigenschaft: `MajorVersion` als Integer

Beschreibung

Gibt die Hauptreleasenummer der Applikation zurück.

Fehler

- 1111 Das application-Objekt ist nicht mehr gültig.
- 1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

MinorVersion

See also

Eigenschaft: `MinorVersion` als 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.

NewProject

Siehe auch

Methode: `NewProject(strPath als String, bDiscardCurrent als Boolean)`

Beschreibung

`NewProject` erstellt ein neues Projekt.

Wenn es bereits ein offenes Projekt gibt, das geändert wurde und `bDiscardCurrent` "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.

OpenProject

Siehe auch

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

`OpenProject` öffnet 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`.

Parent

Siehe auch

Eigenschaft: `Parent` als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

- 1111 Das application-Objekt ist nicht mehr gültig.
- 1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Quit

Siehe auch

Method: `Quit()`

Rückgabewert

Keiner

Beschreibung

Diese Methode beendet XMLSpy. Alle geänderten Dokumente werden ohne Speicherung der Änderungen geschlossen. Dies gilt 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.

ReloadSettings

Siehe auch

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.

RunMacro

Siehe auch

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

Fehler

1111 Das application-Objekt ist nicht mehr gültig.

ScriptingEnvironment

Siehe auch

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.

ServicePackVersion

See also

Eigenschaft: [ServicePackVersion](#) als 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.

ShowApplication**Siehe auch**

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.

ShowFindInFiles**Siehe auch**

Methode: `ShowFindInFiles`(*pSettings* als `FindFilesDlg`) 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.

ShowForm**Siehe auch**

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 *strFormName* an.

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..

Status**Siehe auch**

Eigenschaft: `Status` als [ENUMApplicationStatus](#)

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.

URLDelete**Siehe auch**

Methode: `URLDelete`(`strURL` als String, `strUser` als String, `strPassword` als String)

Rückgabewert

Keiner

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.

URLMakeDirectory**Siehe auch**

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.

Visible

Siehe auch

Eigenschaft: `Visible` als `VARIANT_BOOL`

Beschreibung

Setzt oder ruft das Sichtbarkeitsattribut von XMLSpy ab. Diese Standard-Automation-Eigenschaft macht die Verwendung von [ShowApplication](#) überflüssig.

Fehler

- 1110 Das application-Objekt ist nicht mehr gültig.
- 1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

WarningNumber

Siehe auch

Eigenschaft: `WarningNumber` als integer

Beschreibung

Einige Methoden füllen die Eigenschaft `WarningNumber` mit zusätzlichen Informationen, wenn ein Fehler auftritt.

Derzeit wird diese Eigenschaft nur mit [Documents.OpenFile](#) befüllt.

Fehler

- 1111 Das application-Objekt ist nicht mehr gültig.
- 1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

WarningText

Siehe auch

Eigenschaft: `WarningText` als String

Beschreibung

Einige Methoden befüllen die Eigenschaft `WarningText` mit zusätzlichen Informationen, wenn ein Fehler auftritt.

Derzeit wird diese Eigenschaft nur mit [Documents.OpenFile](#) befüllt.

Fehler

- 1111 Das application-Objekt ist nicht mehr gültig.
- 1100 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

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.

CountItems

Methode: `CountItems()` `nltems` als long

Rückgabewert

Gibt die Anzahl der Menüoptionen zurück.

Fehler

2501 Ungültiges Objekt.

DeleteItem

Methode: `DeleteItem`(IndexPosition als long)

Rückgabewert

Löscht den Menüeintrag mit der im ersten Parameter übergebenen Indexposition.

Fehler

2501 Ungültiges Objekt
2502 Ungültiger Index

GetItemText

Methode: `GetItemText`(IndexPosition als long) Menübefehlsname als String

Rückgabewert

Ruft den Namen des Menübefehls mit der im ersten Parameter übergebenen Indexposition ab.

Fehler

2501 Ungültiges Objekt
2502 Ungültiger Index

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

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

3.2.3 AuthenticDataTransfer

Umbenannt von **DocEditDataTransfer** in **AuthenticDataTransfer**

Das `DocEditView` Objekt wird in `ObAuthenticView` 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](#), [DocEditSelection](#), [DocEditEvent](#) und [DocEditDataTransfer](#) verfügbaren Funktionalitäten stehen nun über [AuthenticView](#), [AuthenticRange](#) und [AuthenticDataTransfer](#) 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.

Siehe auch

Methoden

[getData](#)

Eigenschaften

[dropEffect](#)
[ownDrag](#)
[type](#)

Beschreibung

Die Events `OnDragOver` und `OnBeforeDrop` liefern mit einer Instanz des Typs `AuthenticDataTransfer` Informationen 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 `Unknown`-Schnittstelle unterstützt, zur Verfügung.

dropEffect

Siehe auch

Eigenschaft: `dropEffect` als 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](#) auf TRUE setzen, wenn Sie

noch die veraltete `AuthenticEvent` Schnittstelle verwenden).

Fehler

2101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

getData

Siehe auch

Methode: `getData()` als Variant

Beschreibung

Ruft die Daten im Zusammenhang mit dem gezogenen Objekt ab. Abhängig von [AuthenticDataTransfer.type](#) handelt es sich hierbei entweder um einen String oder um einen COM Interface Pointer vom Typ `Unknown`.

Fehler

2101 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

ownDrag

Siehe auch

Eigenschaft: `ownDrag` als 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.

type

Siehe auch

Eigenschaft: `type` als String (schreibgeschützt)

Beschreibung

Enthält den Datentyp, den Sie mit der [DocEditDataTransfer.getData](#) 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.

3.2.4 AuthenticEventContext

Über die `EventContext` Schnittstellen haben Sie Zugriff auf viele Eigenschaften des Kontexts, in dem ein Makro ausgeführt wird.

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.

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.

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.

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

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.

IsAvailable

Methode: `IsAvailable()` als Boolean

Rückgabewert

Gibt "true" zurück, wenn `EventContext` definiert ist, andernfalls "false".

Fehler

- 2201 Ungültiges Objekt.

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

3.2.5 AuthenticRange

Siehe auch

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

[AppItain](#)
[FirstTextPosition](#)
[FirstXMLData](#)
[FirstXMLDataOffset](#)
[LastTextPosition](#)
[LastXMLData](#)
[LastXMLDataOffset](#)
[Parent](#)

[Collapse](#)
[CollapseToBegin](#)
[CollapseToEnd](#)
[ExpandTo](#)
[Goto](#)
[GotoNext](#)
[GotoPrevious](#)
[IsEmpty](#)
[IsEqual](#)

Methoden

[MoveBegin](#)
[MoveEnd](#)
[NextCursorPosition](#)
[PreviousCursorPosition](#)
[Select](#)
[SelectNext](#)
[SelectPrevious](#)
[SetFromRange](#)

Die folgende Tabelle enthält eine Liste der Methoden zur Bearbeitung von Inhalt. Die meisten dieser Methoden können über Kontextmenüs aufgerufen werden.

Eigenschaften

[Text](#)

Edit operations

[Copy](#)
[Cut](#)
[Delete](#)
[IsCopyEnabled](#)
[IsCutEnabled](#)
[IsDeleteEnabled](#)
[IsPasteEnabled](#)
[Paste](#)

Dynamic table operations

[AppendRow](#)
[DeleteRow](#)
[DuplicateRow](#)
[InsertRow](#)
[IsFirstRow](#)
[IsIndynamicTable](#)
[IsLastRow](#)
[MoveRowDown](#)
[MoveRowUp](#)

Die folgenden Methoden liefern die Funktionen der Authentic-Eingabehilfenfenster für Bereichsobjekte.

Operations of the entry helper windows

Elements	Attributes	Entities
CanPerformActionWin	GetElementAttributeValue	GetEntityNames
CanPerformAction	GetElementAttributeNames	IsEntity
PerformAction	GetElementHierarchy	
	HasElementAttribute	
	IsTextStateApplied	
	SetElementAttributeValue	

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](#), 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](#), [AuthenticView.DocumentEnd](#) und [AuthenticView.WholeDocument](#) zur Verfügung. Die flexibelste Methode ist [AuthenticView.Goto](#). 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 Positioneigenschaften 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](#) und [LastTextPosition](#). 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](#), [FirstXMLDataOffset](#), [LastXMLData](#) und [LastXMLDataOffset](#). 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:

- Mit Hilfe der [Text](#)-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](#), [Copy](#), [Paste](#) und [Delete](#).
- Tabellenoperationen für Tabellen, die dynamisch wachsen können.
- Methoden, die den Funktionalitäten der Authentic-Eingabehilfefenster zugeordnet sind.
- Aufrufe der [XMLData](#) Objekte des zugrunde liegenden Dokuments, um diese direkt zu bearbeiten.

AppendRow**Siehe auch****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 Anhängoperation erfolgreich war. Anderfalls wird *false* zurückgegeben.

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtobjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' -----
' XMLSpy scripting environment - VBScript
' Append row at end of current dynamically growable table
' -----
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' check if we can insert something
If objRange.IsInDynamicTable Then
    objRange.AppendRow
    ' objRange points to beginning of new row
    objRange.Select
End If
```

Application**Siehe auch****Eigenschaft:** [Application](#) als [Application](#) (schreibgeschützt)**Beschreibung**

Ruft das XMLSpy application-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 Ansichtobjekt ist nicht mehr gültig
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

CanPerformAction**Siehe auch****Methode:** [CanPerformAction](#) (*eAction* als [SPYAuthenticActions](#), *strElementName* als String) als Boolean**Beschreibung**

[CanPerformAction](#) und 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](#) verwenden.

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtobjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 2007 Es wurde eine ungültige Aktion definiert.

Beispiele

Siehe [PerformAction](#).

CanPerformActionWith**Siehe auch**

Methode: [CanPerformActionWith](#) (*eAction* als [SPYAuthenticActions](#), *out_arrElementNames* als Variant)

Beschreibung

[PerformActionWith](#) und 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](#).

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtobjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 2007 Es wurde eine ungültige Aktion definiert.

Beispiele

Siehe [PerformAction](#).

Clone**Siehe auch**

Methode: [Clone\(\)](#) als [AuthenticRange](#)

Beschreibung

Gibt eine Kopie des Bereichsobjekts zurück.

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsbjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

CollapsToBegin

Siehe auch

Methode: `CollapsToBegin()` als [AuthenticRange](#)

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 Ansichtsbjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

CollapsToEnd

Siehe auch

Methode: `CollapsToEnd()` als [AuthenticRange](#)

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 Ansichtsbjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Copy

Siehe auch

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 Ansichtsbjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Cut

Siehe auch

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 Ansichtobjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Delete

Siehe auch

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 Ansichtobjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

DeleteRow

Siehe auch

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 Ansichtobjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' -----
' XMLSpy scripting environment - VBScript
' Delete selected row from dynamically growing table
' -----
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection
```



```

' check if we are in a table
If objRange.IsInDynamicTable Then
    objRange.DeleteRow
End If

```

DuplicateRow

Siehe auch

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 Ansichtobjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

```

' -----
' XMLSpy scripting environment - VBScript
' duplicate row in current dynamically growable table
' -----
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' check if we can insert something
If objRange.IsInDynamicTable Then
    objRange.DuplicateRow
    ' objRange points to beginning of new row
    objRange.Select
End If

```

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

ExpandTo**Siehe auch**

Methode: `ExpandTo` (`eKind` als [SPYAuthenticElementKind](#)), als [AuthenticRange](#)

Beschreibung

Wählt das gesamte Element vom Typ `eKind` aus, 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.

FirstTextPosition**Siehe auch**

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](#). 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 links-Taste.

Wenn Sie `FirstTextPosition` auf einen höheren Wert als den der aktuellen [LastTextPosition](#) setzen, wird [LastTextPosition](#) auf den neuen Wert von `FirstTextPosition` gesetzt.

Tipp: Seien Sie beim Setzen der Textcursorposition vorsichtig, da dies im Vergleich zur Cursorpositionierung mittels `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

```
' -----
' XMLSpy scripting environment - VBScript
' -----
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = 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
Dim objRange
' we need to get a (any) range object first
```

```

Set objRange = objAuthenticView.DocumentBegin
objRange.FirstTextPosition = nDocStartPosition
objRange.LastTextPosition = nDocEndPosition

' let's check if we got it right
If objRange.isEqual(objAuthenticView.WholeDocument) Then
    MsgBox "Test using direct text cursor positioning was ok"
Else
    MsgBox "Ooops! "
End If

```

FirstXMLData

Siehe auch

Eigenschaft: [FirstXMLData](#) als [XMLData](#)

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](#) Attribut definiert.

Immer wenn Sie [FirstXMLData](#) auf ein neues Datenobjekt setzen, wird [FirstXMLDataOffset](#) auf die erste Cursorposition innerhalb dieses Elements gesetzt. Nur [XMLData](#)-Objekte, die eine Cursorposition haben, können verwendet werden. Wenn Sie [FirstXMLData](#) / [FirstXMLDataOffset](#) auf eine Position setzen, die größer ist als der aktuelle Wert von [LastXMLData](#) / [LastXMLDataOffset](#) wird zweiteres auf die neue Anfangsposition verschoben.

Tipp: Sie können das zugrunde liegende XML-Dokument in Fällen, in denen die mit dem [AuthenticRange](#)-Objekt verfügbaren Methoden nicht ausreichen, mit Hilfe der Eigenschaften [FirstXMLData](#) und [LastXMLData](#) direkt aufrufen und bearbeiten

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtobjekt ist nicht gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 2008 Interner Fehler
- 2009 Das [XMLData](#)-Objekt kann nicht aufgerufen werden.

Beispiele

```

' -----
' XMLSpy scripting environment - VBScript
' show name of currently selected XMLData element
' -----
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objXMLData
Set objXMLData = 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.
Set objXMLData = objXMLData.Parent
MsgBox "Current selection selects element " & objXMLData.Name

```

FirstXMLDataOffset**Siehe auch****Eigenschaft:** [FirstXMLDataOffset](#) as Long**Beschreibung**

Setzt oder ruft den Versatz der Cursorposition innerhalb des Elements [FirstXMLData](#) für den Anfang des Bereichs ab. Die Versatzpositionen basieren auf den von der Eigenschaft [Text](#) 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 besonders 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](#) eine Position nach dem aktuellen [LastXMLData](#) / [LastXMLDataOffset](#) 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.

Beispiele

```
' -----
' XMLSpy scripting environment - VBScript
' Select the complete text of an XMLData element
' using XMLData based selection and ExpandTo
' -----
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

' first we use the XMLData based range properties
' to select all text of the first XMLData element
' in the current selection
Dim objRange
Set objRange = 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
Dim objRange2
Set objRange2 = objAuthenticView.Selection.ExpandTo(spyAuthenticTag)

' were we successful?
If objRange.IsEqual(objRange2) Then
    objRange.Select()
Else
    MsgBox "Oops"
```

End If

GetElementAttributeNames

Siehe auch

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](#) und [SetElementAttributeValue](#) zu setzen oder abzurufen.

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtobjekt ist nicht mehr gültig.
- 2005 Invalid element name was specified.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

Siehe [SetElementAttributeValue](#).

GetElementAttributeValue

Siehe auch

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 unterstützt wird, aber keinen Wert zugewiesen hat, wird der leere String zurückgegeben. Um die Namen der von einem Element unterstützten Attribute herauszufinden, verwenden Sie [GetElementAttributeNames](#), oder [HasElementAttribute](#).

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtobjekt 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](#).

GetElementHierarchy

Siehe auch

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](#) und damit in Zusammenhang stehenden Methoden aufgerufen werden.

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtobjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

C#-Beispiele

```

namespace ConsoleApplication1
{
    class Program
    {
        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](#).

GetEntityNames

Siehe auch

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](#).

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](#) und [InsertEntity](#).

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

Goto**Siehe auch**

Methode: `Goto (eKind als SPYAuthenticElementKind, nCount als Long, eFrom als SPYAuthenticDocumentPosition) als AuthenticRange`

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.

GotoNext**Siehe auch**

Methode: `GotoNext (eKind als SPYAuthenticElementKind) als AuthenticRange`

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 Ansichtsbjekt 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

```

' -----
' XMLSpy scripting environment - VBScript
' Scan through the whole document word-by-word
' -----

Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objRange
Set objRange = objAuthenticView.DocumentBegin
Dim bEndOfDocument
bEndOfDocument = False

On Error Resume Next
While Not bEndOfDocument
    objRange.GotoNext(spyAuthenticWord).Select
    If ((Err.number - vbObjectError) = 2003) Then
        bEndOfDocument = True
        Err.Clear
    ElseIf (Err.number <> 0) Then
        Err.Raise ' forward error
    End If
Wend

```

GotoNextCursorPosition**Siehe auch**

Methode: `GotoNextCursorPosition()` als [AuthenticRange](#)

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 Ansichtsbjekt ist nicht mehr gültig.
- 2003 Das Ziel liegt hinter dem Ende des Dokuments.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

GotoPrevious**Siehe auch**

Methode: `GotoPrevious` (*eKind* als [SPYAuthenticElementKind](#)) als [AuthenticRange](#)

Beschreibung

Setzt den Bereich auf den Beginn des Elements vom Typ *eKind*, welches sich vor dem Beginn des aktuellen Bereichs befindet. Die Methode gibt das geänderte Bereichsobjekt zurück.

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsbjekt 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

```

' -----
' XMLSpy 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 bEndOfDocument
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

```

GotoPreviousCursorPosition

Siehe auch

Methode: `GotoPreviousCursorPosition()` als [AuthenticRange](#)

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 Ansichtsbjekt ist nicht mehr gültig.
- 2004 Das Ziel liegt vor dem Anfang des Dokuments.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

HasElementAttribute

Siehe auch

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 Ansichtsbjekt ist nicht mehr gültig.
- 2005 Es wurde ein ungültiger Elementname angegeben.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

InsertEntity

Siehe auch

Methode: `InsertEntity` (*strEntityName* als String)

Beschreibung

Ersetzt die Bereichsauswahl durch die angegebene Entity. Die angegebene Entity muss einer der von [GetEntityNames](#) zurückgegebenen Entity-Namen sein.

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsbjekt ist nicht mehr gültig.
- 2005 Es wurde ein unbekannter Name für den Eintrag angegeben.

Beispiele

```
' -----
' XMLSpy scripting environment - VBScript
' Insert the first entity in the list of available entities
' -----
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' first we get the names of all available entities as they
' are shown in the entry helper of XMLSpy
Dim arrEntities
objRange.GetEntityNames arrEntities

' we insert the first one of the list
If UBound(arrEntities) >= 0 Then
    objRange.InsertEntity arrEntities(0)
Else
    MsgBox "Sorry, no entities are available for this document"
End If
```

InsertRow

Siehe auch

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, andernfalls ist das Ergebnis *false*.

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsbjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' -----  
' XMLSpy scripting environment - VBScript  
' Insert row at beginning of current dynamically growing table  
' -----  
Dim objRange  
' we assume that the active document is open in authentic view mode  
Set objRange = Application.ActiveDocument.AuthenticView.Selection  
  
' check if we can insert something  
If objRange.IsInDynamicTable Then  
    objRange.InsertRow  
    ' objRange points to beginning of new row  
    objRange.Select  
End If
```

IsCopyEnabled

Siehe auch

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 Ansichtsbjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

IsCutEnabled

Siehe auch

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 Ansichtsbjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

IsDeleteEnabled

Siehe auch

Eigenschaft: `IsDeleteEnabled` als Boolean (schreibgeschützt)

Beschreibung

Überprüft, ob die Löschoption für diesen Bereich unterstützt wird.

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsbjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

IsEmpty

Siehe auch

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 Ansichtobjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

IsEqual

Siehe auch

Methode: `IsEqual(objCmpRange als AuthenticRange)` 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.

IsFirstRow

Siehe auch

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 Ansichtobjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

IsInDynamicTable

Siehe auch

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 Ansichtsbjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

IsLastRow**Siehe auch**

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 Ansichtsbjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

IsPasteEnabled**Siehe auch**

Eigenschaft: `IsPasteEnabled` als Boolean (schreibgeschützt)

Beschreibung

Überprüft, ob die Einfügeoperationen für diesen Bereich unterstützt wird.

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsbjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

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.

IsTextStateApplied**Siehe auch**

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 Ansichtobjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

LastTextPosition

Siehe auch

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`. 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 links-Taste.

Wenn Sie `LastTextPosition` auf einen niedrigeren Wert als den der aktuellen `FirstTextPosition` setzen, wird `FirstTextPosition` auf den neuen Wert von `LastTextPosition` gesetzt.

Tipp: Seien Sie beim Setzen der Textcursorposition vorsichtig, da dies im Vergleich zur Cursorpositionierung mittels `XMLData` eine kostspielige Operation ist.

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtobjekt ist nicht gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 2006 Es wurde eine Textposition außerhalb des Dokuments definiert.

Beispiele

```
' -----
' XMLSpy scripting environment - VBScript
' -----

Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = 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
Dim objRange
' we need to get a (any) range object first
Set objRange = objAuthenticView.DocumentBegin
objRange.FirstTextPosition = nDocStartPosition
objRange.LastTextPosition = nDocEndPosition

' let's check if we got it right
If objRange.IsEqual(objAuthenticView.WholeDocument) Then
    MsgBox "Test using direct text cursor positioning was ok"
Else
    MsgBox "Oops! "
End If
```

LastXMLData

Siehe auch

Eigenschaft: [LastXMLData](#) als [XMLData](#)

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](#) Attribut definiert.

Immer wenn Sie [LastXMLData](#) auf ein neues Datenobjekt setzen, wird [LastXMLDataOffset](#) auf die letzte Cursorposition innerhalb dieses Elements gesetzt. Nur [XMLData](#)-Objekte, die eine Cursorposition haben, können verwendet werden. Wenn Sie [LastXMLData](#) / [LastXMLDataOffset](#) auf eine Position setzen, die niedriger ist als der aktuelle Wert von [FirstXMLData](#) / [FirstXMLDataOffset](#) wird weiteres wird auf die neue Endposition verschoben.

Tipp: Sie können das zugrunde liegende XML-Dokument in Fällen, in denen die mit dem [AuthenticRange](#)-Objekt verfügbaren Methoden nicht ausreichen, mit Hilfe der Eigenschaften [FirstXMLData](#) und [LastXMLData](#) 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.

LastXMLDataOffset

Siehe auch

Eigenschaft: [LastXMLDataOffset](#) als Long

Beschreibung

Setzt oder ruft den Versatz der Cursorposition innerhalb des Elements [LastXMLData](#) für das Ende des Bereichs ab.

Die Versatzpositionen basieren auf den von der Eigenschaft [Text](#) 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 besonders 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](#) / [LastXMLDataOffset](#) eine Position vor dem aktuellen [FirstXMLData](#)

/ [FirstXMLDataOffset](#) wählt, wird zweiteres an die neue Endposition verschoben.

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsbjekt ist nicht gültig.
- 2005 Es wurde ein ungültiger Versatz definiert.
Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

```
' -----
' XMLSpy scripting environment - VBScript
' Select the complete text of an XMLData element
' using XMLData based selection and ExpandTo
' -----
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

' first we use the XMLData based range properties
' to select all text of the first XMLData element
' in the current selection
Dim objRange
Set objRange = 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
Dim objRange2
Set objRange2 = objAuthenticView.Selection.ExpandTo(spyAuthenticTag)

' were we successful?
If objRange.IsEqual(objRange2) Then
    objRange.Select()
Else
    MsgBox "Oops"
End If
```

MoveBegin

Siehe auch

Methode: [MoveBegin](#) (*eKind* als [SPYAuthenticElementKind](#), *nCount* als Long) als [AuthenticRange](#)

Beschreibung

Verschiebt den Anfang des Bereichs an den Beginn des *nCount* Elements vom Typ *eKind*. Die Zählung beginnt 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 Ansichtsbjekt 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.

MoveEnd

Siehe auch

Methode: `MoveEnd` (*eKind* als `SPYAuthenticElementKind`, *nCount* als Long) als `AuthenticRange`

Beschreibung

Verschiebt das Ende des Bereichs an den Beginn des *nCount* Elements vom Typ *eKind*. Die Zählung beginnt 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 Ansichtsbjekt 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.

MoveRowDown

Siehe auch

Methode: `MoveRowDown()` als Boolean

Beschreibung

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 Ansichtsbjekt ist nicht mehr gültig.
2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

MoveRowUp

Siehe auch

Method: `MoveRowUp()` als 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 Ansichtsbjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

Siehe [JScript - Bubble-Sortierung dynamischer Tabellen](#).

Parent**Siehe auch**

Eigenschaft: `Parent` als [AuthenticView](#) (schreibgeschützt)

Beschreibung

Ruft die Ansicht auf, die dieses Bereichsobjekt enthält.

Fehler

- 2001 Das Authentic-Bereichsobjekt oder das damit in Zusammenhang stehende Ansichtsbjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Paste**Siehe auch**

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 Ansichtsbjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

PerformAction**Siehe auch**

Methode: `PerformAction` (`eAction` als [SPYAuthenticActions](#), `strElementName` als String) als Boolean

Beschreibung

`PerformAction` und 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](#).

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

```
' -----
' XMLSpy scripting environment - VBScript
' Insert the innermost element
' -----

Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' we determine the elements that can be inserted at the current position
Dim arrElements()
objRange.CanPerformActionWith spyAuthenticInsertBefore, arrElements

' we insert the first (innermost) element
If UBound(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"
End If
```

Select

Siehe auch

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

```
' -----
' XMLSpy scripting environment - VBScript
' -----

Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

' set current selection to end of document
objAuthenticView.DocumentEnd.Select()
```

SelectNext**Siehe auch**

Methode: `SelectNext` (*eKind* als [SPYAuthenticElementKind](#)) als [AuthenticRange](#)

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 Ansichtsbjekt 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

```

' -----
' XMLSpy scripting environment - VBScript
' Scan through the whole document word-by-word
' -----

Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objRange
Set objRange = objAuthenticView.DocumentBegin
Dim bEndOfDocument
bEndOfDocument = False

On Error Resume Next
While Not bEndOfDocument
    objRange.SelectNext(spyAuthenticWord).Select
    If ((Err.number - vbObjectError) = 2003) Then
        bEndOfDocument = True
        Err.Clear
    ElseIf (Err.number <> 0) Then
        Err.Raise ' forward error
    End If
Wend

```

SelectPrevious**Siehe auch**

Methode: `GotoPrevious` (*eKind* als [SPYAuthenticElementKind](#)) als [AuthenticRange](#)

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 Ansichtsbjekt 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

```

' -----
' XMLSpy 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 bEndOfDocument
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
        Err.Raise ' forward error
    End If
Wend

```

SetElementAttributeValue**Siehe auch**

Methode: `SetElementAttributeValue` (*strElementName* als String,
strAttributeName als String, *strAttributeValue* als String)

Beschreibung

Ruft den Wert des in *strAttributeName* definierten Attributs für das mit *strElementName* identifizierte Element auf. 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](#) oder [HasElementAttribute](#).

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

```

' -----
' XMLSpy scripting environment - VBScript
' Get and set element attributes
' -----

Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' first we find out all the elements below the beginning of the range
Dim arrElements
objRange.GetElementHierarchy arrElements

```

```

If IsArray(arrElements) Then
    If UBound(arrElements) >= 0 Then
        ' we use the top level element and find out its valid attributes
        Dim arrAttrs()
        objRange.GetElementAttributeNames arrElements(0), arrAttrs

        If UBound(arrAttrs) >= 0 Then
            ' we retrieve the current value of the first valid
attribute
            Dim strAttrVal
            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
        End If
    End If
End If

```

SetFromRange

Siehe auch

Methode: `SetFromRange` (*objSrcRange* als [AuthenticRange](#))

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.

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.

Text**Siehe auch**

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.

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](#), [Delete](#) oder [PerformAction](#).

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.

3.2.6 AuthenticView**Siehe auch****Eigenschaften**

[Application](#)
[AsXMLString](#)
[DocumentBegin](#)
[DocumentEnd](#)
[Event](#)
[MarkupStyle](#)
[Parent](#)
[Section](#)
[XMLDataRoot](#)
[WholeDocument](#)

Methoden

[Goto](#)
[IsRedoEnabled](#)
[IsUndoEnabled](#)
[Print](#)
[Redo](#)
[Undo](#)
[UpdateXMLInstanceEntities](#)

Events

[OnBeforeCopy](#)
[OnBeforeCut](#)
[OnBeforeDelete](#)
[OnBeforeDrop](#)
[OnBeforePaste](#)
[OnDragOver](#)
[OnKeyboardEvent](#)
[OnMouseEvent](#)
[OnSelectionChanged](#)

Beschreibung

Über [AuthenticView](#) und seine untergeordneten Objekte [AuthenticRange](#) und [AuthenticDataTransfer](#) steht 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](#))
[AuthenticEvent](#) (der alte Name war `DocEditEvent`) [Interfaces](#)

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](#). Verwenden Sie für einfache Auswahldefinitionen die Eigenschaften [DocumentBegin](#), [DocumentEnd](#) oder [WholeDocument](#) und die [Goto](#) Methode für komplexere Auswahldefinitionen. Informationen zum Navigieren relativ zu einem bestimmten Dokumentbereich finden Sie unter den Methoden und Eigenschaften des [AuthenticRange](#) Objekts.

Beispiele

```
' -----
' XMLSpy scripting environment - VBScript
' secure access to authentic view object
' -----

Dim objDocument
Set objDocument = Application.ActiveDocument
If (Not objDocument Is Nothing) Then
    ' we have an active document, now check for view mode
    If (objDocument.CurrentViewMode <> spyViewAuthentic) Then
        If (Not objDocument.SwitchViewMode (spyViewAuthentic)) Then
            MsgBox "Active document does not support authentic view
mode"
        Else
            ' now it is safe to access the authentic view object
            Dim objAuthenticView
            Set objAuthenticView = objDocument.AuthenticView
            ' now use the authentic view object

        End If
    End If
Else
    MsgBox "No document is open"
End If
```

Events

OnBeforeCopy

Siehe auch

Event: `OnBeforeCopy()` als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticBeforeCopy()
    'On_AuthenticBeforeCopy=False 'to disable operation
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_AuthenticBeforeCopy()
{
    //On_AuthenticBeforeCopy=False 'to disable operation
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugin.OnEvent (21, ...) //nEvent=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.

OnBeforeCut**Siehe auch**

Event: `OnBeforeCut()` als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticBeforeCut()
    'On_AuthenticBeforeCut=False 'to disable operation
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_AuthenticBeforeCut()
{
    //return False; //to disable operation
}
```

XMLSpy 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.

OnBeforeDelete**Siehe auch**

Event: `OnBeforeDelete()` als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticBeforeDelete()
    'On_AuthenticBeforeDelete=False 'to disable operation
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_AuthenticBeforeDelete()
{
    //return False; //to disable operation
}
```

XMLSpy 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.

OnBeforeDrop

Siehe auch

Event: `OnBeforeDrop` (`i_nXPos` als Long, `i_nYPos` als Long, `i_ipRange` als `AuthenticRange`, `i_ipData` als `cancel Boolean`)

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticBeforeDrop(nXPos, nYPos, objRange, objData)
    'On_AuthenticBeforeDrop=False to disable operation
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
from On_AuthenticBeforeDrop(nXPos, nYPos, objRange, objData)
{
    //disable to disable operation
}
```

XMLSpy 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 Zusammenhang 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)
Dim objSpy As XMLSpyLib.Application
Set objSpy = GetObject("", "XMLSpy.Application")

' this is the event callback routine connected to the OnBeforeDrop
' event of object objView
Private Function objView_OnBeforeDrop( ByVal i_nXPos As Long, ByVal i_nYPos
As Long,
                                     ByVal i_ipRange As IAuthenticRange,
                                     ByVal i_ipData As
IAuthenticDataTransfer) As Boolean

    If (Not i_ipRange Is Nothing) Then
        MsgBox ("Dropping on content is prohibited");
        Return False;
    Else
        Return True;
    End If
```

```

End Function

' use VBA keyword WithEvents to connect to object-level event
Dim WithEvents objView As XMLSpyLib.AuthenticView
Set objView = objSpy.ActiveDocument.AuthenticView

' continue here with something useful ...
' and serve the windows message loop

```

OnBeforePaste

Siehe auch

Event: `OnBeforePaste` (*objData* als Variant, *strType* als String) als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```

Function On_AuthenticBeforePaste(objData, strType)
    'On_AuthenticBeforePaste=False 'to disable operation
EndFunction

```

XMLSpy Skripting-Umgebung - JScript:

```

function On_AuthenticBeforePaste(objData, strType)
{
    //true/false to enable/disable operation
}

```

XMLSpy 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.

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. (Beachten Sie bitte, dass das Event beim Speichern des Dokuments und nicht beim Speichern des SPS-Designs in StyleVision ausgelöst wird.)

OnDragOver

Siehe auch

Event: `OnDragOver` (*nXPos* als Long, *nYPos* als Long, *eMouseEvent* als [SPYMouseEvent](#),

objRange als [AuthenticRange](#), *objData* als [AuthenticDataTransfer](#)) als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticDragOver(nXPos, nYPos, eMouseEvent, objRange,  
    objData)  
    'On_AuthenticDragOver=False 'to disable operation  
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
from On_AuthenticDragOver(nXPos, nYPos, eMouseEvent, objRange, objData)  
{  
    //true/false to disable operation  
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugin.OnEvent (10, ...) //nEvent=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 abubrechen. 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)  
Dim objSpy As XMLSpyLib.Application  
Set objSpy = GetObject("", "XMLSpy.Application")  
  
' this is the event callback routine connected to the OnDragOver  
' event of object objView  
Private Function objView_OnDragOver(ByVal i_nXPos As Long, ByVal i_nYPos As  
Long,  
    ByVal i_eMouseEvent As SPYMouseEvent,  
    ByVal i_ipRange As IAuthenticRange,  
    ByVal i_ipData As  
IAuthenticDataTransfer) As Boolean  
  
    If (((i_eMouseEvent And spyShiftKeyDownMask) <> 0) And  
        (Not i_ipRange Is Nothing)) Then  
        MsgBox ("Floating over element " &  
i_ipRange.FirstXMLData.Parent.Name);  
    End If  
  
    Return True;  
End Function
```

```
' use VBA keyword WithEvents to connect to object-level event
Dim WithEvents objView As XMLSpyLib.AuthenticView
Set objView = objSpy.ActiveDocument.AuthenticView

' continue here with something useful ...
' and serve the windows message loop
```

OnKeyboardEvent

Siehe auch

Event: OnKeyboardEvent (*eKeyEvent* als [SPYKeyEvent](#), *nKeyCode* als Long, *nVirtualKeyStatus* als Long) als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticKeyboardEvent(eKeyEvent, nKeyCode,
nVirtualKeyStatus)
    'On_AuthenticKeyboardEvent=True 'to cancel bubbling of event
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_AuthenticKeyboardEvent(eKeyEvent, nKeyCode, nVirtualKeyStatus)
{
    //return true; //to cancel bubbling of event
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugin.OnEvent (30, ...) //nEventId=30
```

Beschreibung

Dieses Event wird für WM_KEYDOWN, WM_KEYUP und 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](#) definierten Bit-Masken, um eine Überprüfung der unterschiedlichen Tasten und ihrer Kombinationen durchzuführen.

OnLoad

Event: OnLoad ()

Beschreibung: Mit OnLoad können einige Funktionalitäten der Authentic-Ansicht eingeschränkt werden, wie im Beispiel unten gezeigt:

```
function On_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-Identifizier bearbeitet. Siehe [vollständige Liste](#).

OnMouseEvent

Siehe auch

Event: `OnMouseEvent` (`nXPos` als Long, `nYPos` als Long, `eMouseEvent` als `SPYMouseEvent`, `objRange` als `AuthenticRange`) als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticMouseEvent(nXPos, nYPos, eMouseEvent, objRange)
    'On_AuthenticMouseEvent=True 'to cancel bubbling of event
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_AuthenticMouseEvent(nXPos, nYPos, eMouseEvent, objRange)
{
    //return true to cancel bubbling of event
}
```

XMLSpy 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` 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`.

OnSelectionChanged

Siehe auch

Event: `OnSelectionChanged` (`objNewSelection` als `AuthenticRange`)

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_AuthenticSelectionChanged(objNewSelection)
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```

from On_AuthenticSelectionChanged(objNewSelection)
{
}

```

XMLSpy 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)
Dim objSpy As XMLSpyLib.Application
Set objSpy = GetObject("", "XMLSpy.Application")

' this is the event callback routine connected to the OnSelectionChanged
' event of object objView
Private Sub objView_OnSelectionChanged (ByVal i_ipNewRange As
XMLSpyLib.IAuthenticRange)
    MsgBox ("new selection: " & i_ipNewRange.Text)
End Sub

' use VBA keyword WithEvents to connect to object-level event
Dim WithEvents objView As XMLSpyLib.AuthenticView
Set objView = objSpy.ActiveDocument.AuthenticView

' continue here with something useful ...
' and serve the windows message loop

```

OnToolBarButtonClicked

Event: OnToolBarButtonClicked (Button identifier)

Beschreibung: OnToolBarButtonClicked wird ausgelöst, wenn der Benutzer auf eine Symbolleisten-Schaltfläche klickt. Über den Parameter Schaltflächen-Idenfier, kann ermittelt werden, auf welche Schaltfläche geklickt wurde. Unten sehen Sie die Liste der vordefinierten Schaltflächen-Idenfier:

- 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.

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](#).

`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
function On_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.

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`.

Application

Siehe auch

Eigenschaft: `Application` als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

- 2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

AsXMLString

Siehe auch

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 `XMLString` der aktuellen Schemadatei nicht entspricht, wird Fehler 2011 zurückgegeben.

Fehler

- 2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
- 2011 `AsXMLString` wurde auf einen Wert gesetzt, der gemäß der aktuellen Schemadatei kein gültiger XML-Code ist.

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.

CreateXMLNode

Methode: `CreateXMLNode (nKind als SPYXMLDataKind) als XMLData`

Rückgabewert

Die Methode gibt das neue `XMLData` Objekt zurück.

Beschreibung

Um ein neues `XMLData` Objekt zu erstellen, verwenden Sie die Methode `CreateXMLNode()`. Siehe auch "Verwendung von `XMLData`".

Fehler

- 2000 Ungültiges Objekt.
- 2012 XML-Node kann nicht erstellt werden.

DisableAttributeEntryHelper

Methode: `DisableAttributeEntryHelper()`

Beschreibung

`DisableAttributeEntryHelper()` deaktiviert die Attribut-Eingabehilfe in XMLSpy, Authentic Desktop und dem Authentic Browser Plug-in.

Fehler

- 2000 Ungültiges Objekt.

DisableElementEntryHelper

Methode: `DisableElementEntryHelper()`

Beschreibung

`DisableElementEntryHelper()` deaktiviert die Element-Eingabehilfe in XMLSpy, Authentic Desktop und dem Authentic Browser Plug-in.

Fehler

2000 Ungültiges Objekt.

DisableEntityEntryHelper

Methode: `DisableEntityEntryHelper()`

Beschreibung

`DisableEntityEntryHelper()` deaktiviert die Entity-Eingabehilfe in XMLSpy, Authentic Desktop und dem Authentic Browser Plug-in.

Fehler

2000 Ungültiges Objekt.

DocumentBegin

Siehe auch

Eigenschaft: `DocumentBegin` als [AuthenticRange](#) (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.

DocumentEnd

Siehe auch

Eigenschaft: `DocumentEnd` als [AuthenticRange](#) (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.

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.

EvaluateXPath

Methode: `EvaluateXPath` (XMLData als [XMLData](#), 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.

Event

Siehe auch

Eigenschaft: `Event` als [AuthenticEvent](#) (schreibgeschützt)

Beschreibung

Über diese Eigenschaft erhalten Sie auf dieselbe Art wie bei [ObjAuthenticView.event](#) Zugriff auf die Parameter des letzten Event. Da nun alle Events für die Skripting-Umgebung und externe Clients mit Parameters verfügbar sind, sollte diese `Event` Eigenschaft 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.

EventContext

Eigenschaft: `EventContext()` als `EventContext`

Beschreibung

Über die Eigenschaft [EventContext](#) erhalten Sie Zugriff auf den Kontext des laufenden Makros. Nähere Informationen dazu finden Sie unter der Beschreibung zur [EventContext](#) Schnittstelle.

Fehler

- 2000 Ungültiges Objekt.

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-Identifizier gekennzeichnete Schaltfläche ([siehe Liste oben](#)) 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](#) 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-Identifizier.

Goto**Siehe auch**

Methode: `Goto` (*eKind* als [SPYAuthenticElementKind](#), *nCount* als Long, *eFrom* als [SPYAuthenticDocumentPosition](#)) als [AuthenticRange](#)

Beschreibung

Ruft ein Bereichsobjekt auf, das auf den Anfang 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

```
' -----
' XMLSpy scripting environment - VBScript
' -----

Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

On Error Resume Next
Dim objRange
' goto beginning of first table in document
Set objRange = objAuthenticView.Goto (spyAuthenticTable, 1,
spyAuthenticDocumentBegin)
If (Err.number = 0) Then
```

```
objRange.Select()  
Else  
    MsgBox "No table found in document"  
End If
```

IsRedoEnabled

Siehe auch

Eigenschaft: [IsRedoEnabled](#) als Boolean (schreibgeschützt)

Beschreibung

True, wenn Wiederhol-Schritte verfügbar sind und [Redo](#) möglich ist.

Fehler

- 2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

IsUndoEnabled

Siehe auch

Eigenschaft: [IsUndoEnabled](#) als Boolean (schreibgeschützt)

Beschreibung

True, wenn Rückgängig-Schritte verfügbar sind und [Undo](#) möglich ist.

Fehler

- 2000 Das Authentic-Ansichtsobjekt ist nicht mehr gültig.
- 2005 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

MarkupVisibility

Siehe auch

Eigenschaft: [MarkupVisibility](#) als [SPYAuthenticMarkupVisibility](#)

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.

Parent

Siehe auch

Eigenschaft: [Parent](#) als [Document](#) (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.

Print**Siehe auch**

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.

Redo**Siehe auch**

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.

Selection**Siehe auch**

Eigenschaft: `Selection` als [AuthenticRange](#)

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

```
' -----  
' XMLSpy scripting environment - VBScript  
' -----  
Dim objAuthenticView  
' we assume that the active document is open in authentic view mode  
Set objAuthenticView = 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
End If

```

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-Identifizier gekennzeichnete Schaltfläche ([siehe Liste oben](#)) 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](#) 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-Identifizier.

Undo

Siehe auch

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.

UpdateXMLInstanceEntities

Siehe auch

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

```
// -----  
// XMLSpy scripting environment - JavaScript  
// -----  
if(Application.ActiveDocument &&  
( Application.ActiveDocument.CurrentViewMode == 4))  
{  
    var objDocType;  
    objDocType =  
Application.ActiveDocument.DocEditView.XMLRoot.GetFirstChild(10);  
  
    if(objDocType)  
    {  
        var objEntity = Application.ActiveDocument.CreateChild(14);  
        objEntity.Name = "child";  
        objEntity.TextValue = "SYSTEM \"child.xml\"";  
        objDocType.AppendChild( objEntity);  
  
        Application.ActiveDocument.AuthenticView.UpdateXMLInstanceEntities();  
    }  
}
```

WholeDocument

Siehe auch

Eigenschaft: [WholeDocument](#) als [AuthenticRange](#) (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.

XMLDataRoot

Siehe auch

Eigenschaft: [XMLDataRoot](#) als [XMLData](#) (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.

3.2.7 CodeGeneratorDlg

Siehe auch

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](#)

[Parent](#)

Eigenschaften zur Auswahl der Programmiersprache

[ProgrammingLanguage](#)

[TemplateFileName](#)

[CompatibilityMode](#)

Einstellungen für C++-Code

[CPPSettings_DOMType](#)

[CPPSettings_LibraryType](#)

[CPPSettings_UseMFC](#)

[CPPSettings_GenerateVC6ProjectFile](#)

[CPPSettings_GenerateVSProjectFile](#)

Einstellungen für C#-Code

[CSharpSettings_ProjectType](#)

Behandlung von Dialogfeldern für die oben genannten Codegenerierungseigenschaften

[PropertySheetDialogAction](#)

Eigenschaften zur Auswahl des Ausgabepfads

[OutputPath](#)

[OutputPathDialogAction](#)

Darstellung des Ergebnisses

[OutputResultDialogAction](#)

Beschreibung

Mit Hilfe dieses Objekts können Sie die Generierung von Programmcode für Schemadateien konfigurieren. Die Methode [GenerateProgramCode](#) erwartet einen CodeGeneratorDlg als Parameter, um die Codegenerierung sowie die damit in Zusammenhang stehenden Interaktionen des Benutzers zu konfigurieren.

Application**Siehe auch**

Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Eigenschaft: [Application](#) als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

2200 Das Objekt ist nicht mehr gültig.

2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

CompatibilityMode

Eigenschaft: [CompatibilityMode](#) als Boolean

Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Beschreibung

Wird auf `true` gesetzt, um Code zu generieren, der mit XMLSpy 2005R3 kompatibel ist. Wird auf `false` gesetzt, 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.

CPPSettings_DOMType

Eigenschaft: [CPPSettings_DOMType](#) als [SPYDOMType](#)

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.

CPPSettings_GenerateVC6ProjectFile

Eigenschaft: [CPPSettings_GenerateVC6ProjectFile](#) als 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.

CPPSettings_GenerateGCCMakefile

Eigenschaft: [CPPSettings_GenerateGCCMakefile](#) als 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.

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.

CPPSettings_GenerateVSProjectFile

Eigenschaft: [CSharpSettings_GenerateVSProjectFile](#) als [SPYProjectType](#)

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.

CPPSettings_LibraryType

Eigenschaft: [CPPSettings_LibraryType](#) als [SPYLibType](#)

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.

CPPSettings_UseMFC

Eigenschaft: [CPPSettings_UseMFC](#) als 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.

CSharpSettings_ProjectType

Eigenschaft: [CSharpSettings_ProjectType](#) als [SPYProjectType](#)

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.

OutputPath

Eigenschaft: [OutputPath](#) als 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.

OutputPathDialogAction

Eigenschaft: [OutputPathDialogAction](#) als [SPYDialogAction](#)

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 diesen Wert auf `spyDialogUserInput(2)`, damit das Dialogfeld standardmäßig mit dem aktuellen Wert der Eigenschaft [OutputPath](#) angezeigt wird. 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.

OutputResultDialogAction

Eigenschaft: [OutputResultDialogAction](#) als [SPYDialogAction](#)

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.

Parent**Siehe auch**

Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Eigenschaft: `Parent` als [Dialogs](#) (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.

ProgrammingLanguage

Eigenschaft: `ProgrammingLanguage` als [ProgrammingLanguage](#)

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](#) auf die entsprechende mit XMLSpy installierte 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.

PropertySheetDialogAction

Eigenschaft: `PropertySheetDialogAction` als [SPYDialogAction](#)

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.

TemplateFileName

Eigenschaft: [TemplateFileName](#) als 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 XMLSpy Installationsverzeichnisses sind Vorlagendateien für C++, C# oder Java enthalten.

Wenn Sie diese Eigenschaft auf eine der Codegenerierungsvorlagendateien Ihrer XMLSpy Installation setzen, wird die Eigenschaft [ProgrammingLanguage](#) automatisch auf den entsprechenden Wert gesetzt.

Fehler

- 2200 Das Objekt ist nicht mehr gültig.
2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

3.2.8 DatabaseConnection

Siehe auch

Eigenschaften für den Import und Export

[File](#) oder
[ADOConnection](#) oder
[ODBCConnection](#)

Eigenschaften nur für den Import

[DatabaseKind](#)
[SQLSelect](#)
[AsAttributes](#)
[ExcludeKeys](#)
[IncludeEmptyElements](#)
[NumberDateTimeFormat](#)
[NullReplacement](#)
[CommentIncluded](#)

Eigenschaften nur für den Export

[CreateMissingTables](#)
[CreateNew](#)
[TextFieldLen](#)
[DatabaseSchema](#)

Eigenschaften für die Generierung eines XML-Schemas anhand der Datenbankstruktur

[PrimaryKeys](#)
[ForeignKeys](#)
[UniqueKeys](#)
[SchemaExtensionType](#)
[SchemaFormat](#)
[ImportColumnsType](#)

Beschreibung

[DatabaseConnection](#) definiert 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.

ADOConnection

Siehe auch

Eigenschaft: `ADOConnection` als `String`

Beschreibung

Die Eigenschaft `ADOConnection` enthält einen Connection String. Verwenden Sie entweder diese Eigenschaft oder [ODBCConnection](#) oder [File](#), um eine Datenbank zu referenzieren.

Fehler

Es werden keine Fehlercodes zurückgegeben.

Beispiel

```
Dim objSpyConn As DatabaseConnection
Set objSpyConn = objSpy.GetDatabaseSettings

Dim objADO As DataLinks
Set objADO = CreateObject("DataLinks")

If Not (objADO Is Nothing) Then
    Dim objConn As Connection
    Set objConn = objADO.PromptNew
    objSpyConn.ADOConnection = objConn.ConnectionString
End If
```

AsAttributes

Siehe auch

Eigenschaft: `AsAttributes` als `Boolean`

Beschreibung

Setzen Sie `AsAttributes` auf "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.GetDatabaseImportElements](#) verwendet.

Fehler

Es werden keine Fehlercodes zurückgegeben.

CommentIncluded

Siehe auch

Eigenschaft: `CommentIncluded` als `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.

CreateMissingTables

Siehe auch

Eigenschaft: [CreateMissingTables](#) als 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.

CreateNew

Siehe auch

Eigenschaft: [CreateNew](#) als Boolean

Beschreibung

Setzen Sie `CreateNew` auf "true", wenn Sie beim Export eine neue Datenbank erstellen möchten. Eine möglicherweise bereits vorhandene Datenbank wird dabei überschrieben. Siehe auch [DatabaseConnection.File](#). Der Standardwert ist "false". Diese Eigenschaft wird nur beim Export in eine Datenbank verwendet.

Fehler

Es werden keine Fehlercodes zurückgegeben.

DatabaseKind

Siehe auch

Eigenschaft: [DatabaseKind](#) als [SPYDatabaseKind](#)

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.

DatabaseSchema

Siehe auch

Eigenschaft: [DatabaseSchema](#) als 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 Datenbank verwendet.

Fehler

Es werden keine Fehlercodes zurückgegeben.

ExcludeKeys

Siehe auch

Eigenschaft: [ExcludeKeys](#) als `Boolean`

Beschreibung

Setzen Sie `ExcludeKeys` auf "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.

File

Siehe auch

Eigenschaft: [File](#) als `String`

Beschreibung

Die Eigenschaft `File` definiert 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](#) oder [ADOConnection](#) um die Datenbank zu referenzieren.

Siehe auch [Import und Export](#).

Fehler

Es werden keine Fehlercodes zurückgegeben.

ForeignKeys

Siehe auch

Eigenschaft: [ForeignKeys](#) als `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.

ImportColumnsType

Siehe auch

Eigenschaft: [ImportColumnsType](#) als [SPYImportColumnsType](#)

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.

IncludeEmptyElements**Siehe auch**

Eigenschaft: [IncludeEmptyElements](#) als `Boolean`

Beschreibung

Setzen Sie `IncludeEmptyElements` auf "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.

NullReplacement**Siehe auch**

Eigenschaft: [NullReplacement](#) als `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.

NumberDateTimeFormat**Siehe auch**

Eigenschaft: [NumberDateTimeFormat](#) als [SPYNumberDateTimeFormat](#)

Beschreibung

Die Eigenschaft `NumberDateTimeFormat` definiert das Format von Datums- und Uhrzeitwerten. Die Standardeinstellung ist [spySystemLocale](#). Diese Eigenschaft wird nur beim Import von Datenbanken verwendet.

Fehler

Es werden keine Fehlercodes zurückgegeben.

ODBCConnection**Siehe auch**

Eigenschaft: `ODBCConnection` als `String`

Beschreibung

Die Eigenschaft `ODBCConnection` enthält einen ODBC Connection String. Verwenden Sie entweder diese Eigenschaft oder [ADOConnection](#) oder [File](#) um eine Datenbank zu referenzieren.

Fehler

Es werden keine Fehlercodes zurückgegeben.

PrimaryKeys

Siehe auch

Eigenschaft: `PrimaryKeys` als `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.

SchemaExtensionType

Siehe auch

Eigenschaft: `SchemaExtensionType` als [SPYSchemaExtensionType](#)

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](#).

Fehler

Es werden keine Fehlercodes zurückgegeben.

SchemaFormat

See also

Eigenschaft: `SchemaFormat` als [SPYSchemaFormat](#)

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](#).

Fehler

Es werden keine Fehlercodes zurückgegeben.

SQLSelect

Siehe auch

Eigenschaft: [SQLSelect](#) als [String](#)

Beschreibung

Die SQL-Abfrage für den Import ist in der Eigenschaft [SQLSelect](#) gespeichert. Diese Eigenschaft wird nur beim Import von Datenbanken verwendet. Siehe auch [Import und Export](#).

Fehler

Es werden keine Fehlercodes zurückgegeben.

TextFieldLen

Siehe auch

Eigenschaft: [TextFieldLen](#) als [Integer](#)

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.

UniqueKeys

Siehe auch

Eigenschaft: [UniqueKeys](#) als [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.

3.2.9 Dialogs

Siehe auch

Eigenschaften und Methoden

Standard-Automation-Eigenschaften

[Application](#)

[Parent](#)

Verschiedene Dialog-Objekte

[CodeGeneratorDlg](#)

[FileSelectionDlg](#)

[SchemaDocumentationDlg](#)

[GenerateSampleXMLDlg](#)

[DTDSchemaGeneratorDlg](#)
[FindInFilesDlg](#)
[WSDLDocumentationDlg](#)
[WSDL20DocumentationDlg](#)
[XBRLDocumentationDlg](#)

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.

Application

Siehe auch

Eigenschaft: [Application](#) als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

- 2300 Das Objekt ist nicht mehr gültig.
- 2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

CodeGeneratorDlg

Siehe auch

Ist nur in der Enterprise Edition verfügbar/aktiv. Bei Aufruf über eine andere Version wird ein Fehler zurückgegeben.

Eigenschaft: [CodeGeneratorDlg](#) als [CodeGeneratorDlg](#) (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.

FileSelectionDlg

Siehe auch

Eigenschaft: [FileSelectionDlg](#) als [FileSelectionDlg](#) (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.

Parent**Siehe auch**

Eigenschaft: `Parent` als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

- 2300 Das Objekt ist nicht mehr gültig.
- 2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

SchemaDocumentationDlg**Siehe auch**

Eigenschaft: `SchemaDocumentationDlg` als [SchemaDocumentationDlg](#) (schreibgeschützt)

Beschreibung

Ruft eine neue Instanz eines Dialogfeldobjekts ab, das die Generierung von Schemadokumentation parametrisiert. Informationen zur Verwendung finden Sie unter [Document.GenerateSchemaDocumentation](#).

Fehler

- 2300 Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
- 2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

GenerateSampleXMLDlg**Siehe auch**

Eigenschaft: `GenerateSampleXMLDlg` als [GenerateSampleXMLDlg](#) (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](#).

Fehler

- 2300 Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
- 2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

DTDSchemaGeneratorDlg**Siehe auch**

Eigenschaft: `DTDSchemaGeneratorDlg` als [DTDSchemaGeneratorDlg](#) (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](#).

Fehler

- 2300 Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
- 2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

FindInFilesDlg

Siehe auch

Eigenschaft: `FindInFilesDlg` als [FindInFilesDlg](#) (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](#).

Fehler

- 2300 Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
- 2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

WSDLDocumentationDlg

Siehe auch

Eigenschaft: `WSDLDocumentationDlg` als [WSDLDocumentationDlg](#) (schreibgeschützt)

Beschreibung

Ruft eine neue Instanz eines Dialogfeldobjekts ab, das die Generierung von WSDL-Dokumentation parametrisiert. Informationen zur Verwendung finden Sie unter [Document.GenerateWSDLDocumentation](#).

Fehler

- 2300 Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
- 2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

WSDL20DocumentationDlg

Siehe auch

Eigenschaft: `WSDL20DocumentationDlg` als [WSDL20DocumentationDlg](#) (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.GenerateWSD20LDocumentation](#).

Fehler

- 2300 Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

XBRLDocumentationDlg

Siehe auch

Eigenschaft: [XBRL20DocumentationDlg](#) als [XBRL20DocumentationDlg](#) (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.GenerateXBRLDocumentation](#).

Fehler

- 2300 Das Dialogs-Objekt oder eines seiner übergeordneten Objekte ist nicht mehr gültig.
2301 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

3.2.10 Document

Siehe auch

Eigenschaften und Methoden

Standard-Automation-Eigenschaften

[Application](#)
[Parent](#)

Verschiedene Dokumenteigenschaften und Methoden

[SetActiveDocument](#)
[Encoding](#)
[SetEncoding \(obsolete\)](#)
[Suggestions](#)

XML-Validierung

[IsWellFormed](#)
[IsValid](#)
[SetExternalIsValid](#)

Dokumentkonvertierung und -transformation

[AssignDTD](#)
[AssignSchema](#)
[AssignXSL](#)
[AssignXSLFO](#)
[ConvertDTDOrSchema](#)
[ConvertDTDOrSchemaEx](#)
[GenerateDTDOrSchema](#)
[GenerateDTDOrSchemaEx](#)
[CreateSchemaDiagram](#)
[ExecuteXQuery](#)
[TransformXSL](#)
[TransformXSLEx](#)
[TransformXSLFO](#)
[GenerateProgramCode](#) (nur Enterprise Edition)
[GenerateSchemaDocumentation](#)

[GenerateSampleXML](#)
[GenerateWSDL20Documentation](#)
[GenerateWSDLDocumentation](#)
[GenerateXBRLDocumentation](#)
[ConvertToWSDL20](#)

Dokumentexport

[GetExportElementList](#)
[ExportToText](#)
[ExportToDatabase](#)
[CreateDBStructureFromXMLSchema](#)
[GetDBStructureList](#)

Dateispeicherung und -benennung

[FullName](#)
[Name](#)
[Path](#)
[GetPathName \(obsolete\)](#)
[SetPathName \(obsolete\)](#)
[Title](#)
[IsModified](#)
[Saved](#)
[SaveAs](#)
[Save](#)
[SaveInString](#)
[SaveToURL](#)
[Close](#)

Aufruf einer Ansicht

[CurrentViewMode](#)
[SwitchViewMode](#)
[AuthenticView](#)
[GridView](#)
[DocEditView \(obsolete\)](#)

Zugriff auf XMLData

[RootElement](#)
[DataRoot](#)
[CreateChild](#)
[UpdateViews](#)
[StartChanges](#)
[EndChanges](#)
[UpdateXMLData](#)

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](#)
[Application.Documents](#)

Verwenden Sie eine der folgenden Methoden, um ein neues Dokument in XMLSpy zu öffnen:

[Documents.OpenFile](#)
[Documents.OpenURL](#)
[Documents.OpenURLDialog](#)
[Documents.NewFile](#)
[Documents.NewFileFromText](#)

[SpyProjectItem.Open](#)
[Application.ImportFromDatabase](#)
[Application.ImportFromSchema](#)
[Application.ImportFromText](#)
[Application.ImportFromWord](#)
[Document.ConvertDTDOrSchema](#)
[Document.GenerateDTDOrSchema](#)

Events

OnBeforeSaveDocument

Siehe auch

Event: *OnBeforeSaveDocument*(*objDocument* als [Document](#), *objDialog* als [FileSelectionDlg](#))

XMLSpy Skripting-Umgebung - VBScript:

```

Function On_BeforeSaveDocument(objDocument, objDialog)
EndFunction

' 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)
EndFunction

```

XMLSpy 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)
{
}

```

XMLSpy 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 [FileSelectionDlgDialogAction](#) von *io_objDialog* auf ihrem Standardwert [spyDialogOK](#). Um das Speichern des Dokuments abzubrechen, setzen Sie diese Eigenschaft auf [spyDialogCancel](#).

OnBeforeCloseDocument

Siehe auch

Event: `OnBeforeCloseDocument(objDocument als Document) als Boolean`

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_BeforeCloseDocument(objDocument)
    'On_BeforeCloseDocument=False 'to prohibit closing of document
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_BeforeCloseDocument(objDocument)
{
    //return false; 'to prohibit closing of document/
}
```

XMLSpy 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.

OnBeforeValidate

Siehe auch

Event: `OnBeforeValidate(objDocument als Document, bOnLoading als Boolean, bOnCommand als Boolean) als Boolean`

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_BeforeValidate(objDocument, bOnLoading, bOnCommand)
    On_BeforeValidate=bCancelDefaultValidation 'set by the script if
necessary
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_BeforeValidate(objDocument, eViewMode, bActivated)
{
    return bCancelDefaultValidation //set by the script if necessary
}
```

XMLSpy 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 [SetExternallsValid](#) das Validierungsergebnis definieren.

`bOnLoading` ist "true", wenn das Event bei der Anfangsvalidierung beim Laden des Dokuments ausgelöst wird.

`bOnCommand` ist "true", wenn der Benutzer in der Symbolleiste oder im Menü den Befehl "Validieren" ausgewählt hat.

Verfügbar mit Typbibliothek Version 1.5

OnCloseDocument

Siehe auch

Event: OnCloseDocument(*objDocument* als [Document](#))

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_Close Document(objDocument)
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_Close Document(objDocument)
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (8, ...) //nEventId=8
```

Beschreibung

Dieses Event wird ausgelöst, wenn ein Dokument geschlossen wird. Ändern Sie das Dokument nicht von diesem Event aus.

OnViewActivation

Siehe auch

Event: OnViewActivation(*objDocument* als [Document](#), *eViewMode* als [SPYViewModes](#), *bActivated* als Boolean)

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_ViewActivation(objDocument, eViewMode, bActivated)
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_ViewActivation(objDocument, eViewMode, bActivated)
{
}
```

XMLSpy 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](#).

Application

Siehe auch

Eigenschaft: Application als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

- 1400 Das Objekt ist nicht mehr gültig.
- 1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

AssignDTD**Siehe auch**

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.

Siehe auch Simple document access.

Fehler

- 1400 Das Objekt ist nicht mehr gültig.
- 1409 Dem Dokument darf keine DTD zugewiesen werden.

AssignSchema**Siehe auch**

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.

Siehe auch Simple document access.

Fehler

- 1400 Das Objekt ist nicht mehr gültig.
- 1409 Dem Dokument darf keine Schemadatei zugewiesen werden.

AssignXSL**Siehe auch**

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

AssignXSLFO

Siehe auch

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

AsXMLString

Siehe auch

Eigenschaft: `AsXMLString` als 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.

AuthenticView

Siehe auch

Methode: `AuthenticView` als `AuthenticView` (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` und `DocEditView` 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` und seinen Unterobjekten enthalten, sodass `DocEditView` nicht mehr benötigt wird.

Fehler

- 1400 Das Objekt ist nicht mehr gültig.
- 1417 Das Dokument muss im Authentic-Ansichtsmodus geöffnet sein.

Beispiele

```
' -----  
' XMLSpy scripting environment - VBScript  
' secure access to authentic view object
```

```

' -----
Dim objDocument
Set objDocument = Application.ActiveDocument
If (Not objDocument Is Nothing) Then
    ' we have an active document, now check for view mode
    If (objDocument.CurrentViewMode <> spyViewAuthentic) Then
        If (Not objDocument.SwitchViewMode (spyViewAuthentic)) Then
            MsgBox "Active document does not support authentic view
mode"
        Else
            ' now it is safe to access the authentic view object
            Dim objAuthenticView
            Set objAuthenticView = objDocument.AuthenticView
            ' now use the authentic view object

        End If
    End If
Else
    MsgBox "No document is open"
End If

```

Close**Siehe auch**

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.

ConvertDTDOrSchema**Siehe auch**

Methode: `ConvertDTDOrSchema` (*nFormat* als [SPYDTDSchemaFormat](#), *nFrequentElements* als [SPYFrequentElements](#))

Parameter

`nFormat`

Setzt das Schemaausgabeformat auf DTD oder W3C.

`nFrequentElements`

Erstellt komplexe Elemente als Elemente oder complexTypes.

Beschreibung

`ConvertDTDOrSchema` nimmt das vorhandene Schemaformat und konvertiert es in ein anderes Format.

Zur Feinabtimung einer DTD / Schemakonvertierung verwenden Sie [ConvertDTDOrSchemaEx](#).

Fehler

- 1400 Das Objekt ist nicht mehr gültig.
- 1412 Fehler bei der Konvertierung.

ConvertDTDOrSchemaEx

Siehe auch

Methode: `ConvertDTDOrSchemaEx` (*nFormat* als [SPYDTDSchemaFormat](#), *nFrequentElements* als [SPYFrequentElements](#), *sOutputPath* als String, *nOutputPathDialogAction* als [SPYDialogAction](#))

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

`ConvertDTDOrSchemaEx` nimmt ein vorhandenes Schemaformat und konvertiert es in ein anderes Format.

Fehler

- 1400 Das Objekt ist nicht mehr gültig.
- 1412 Fehler bei der Konvertierung.

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/Ordern 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. Siehe [Document.CurrentViewMode](#). und [SPYViewModes](#).

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.

CreateChild

Siehe auch

Methode: `CreateChild` (*nKind* als [SPYXMLDataKind](#)) als [XMLData](#)

Rückgabewert

Die Methode gibt das neue `XMLData` Objekt zurück.

Beschreibung

Um ein neues `XMLData` Objekt zu erstellen, verwenden Sie die Methode `CreateChild()`. Siehe auch Verwendung von `XMLData`.

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.

CreateDBStructureFromXMLSchema

Siehe auch

Methode: `CreateDBStructureFromXMLSchema` (*pDatabase* als [DatabaseConnection](#), *pTables* als [ElementList](#), *bDropTableWithExistingName* als `Boolean`) als `String`

Beschreibung

`CreateDBStructureFromXMLSchema` exportiert die angegebenen Tabellen in die angegebene Datenbank. Die Funktion gibt die SQL-Anweisungen zurück, die zur Durchführung der Änderungen erforderlich waren.

Siehe auch [GetDBStructureList](#).

Fehler

- 1429 Datenbankauswahl fehlt.
- 1430 Der Dokumentexport ist fehlgeschlagen.

CreateSchemaDiagram

Siehe auch

Method: `CreateSchemaDiagram` (*nKind* als [SPYSchemaDefKind](#), *strName* als `String`, *strFile* als `String`)

Rückgabewert

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](#). und [SPYViewModes](#).

Fehler

- 1400 Das Objekt ist nicht mehr gültig.
- 1414 Das Diagramm konnte nicht gespeichert werden.
- 1415 Es wurde ein ungültiger Schemadefinitionstyp angegeben.

CurrentViewMode**Siehe auch**

Methode: [CurrentViewMode](#) als [SPYViewModes](#)

Beschreibung

Die Eigenschaft enthält den aktuellen Ansichtsmodus des Dokuments. Siehe auch [Document.SwitchViewMode](#).

Fehler

- 1400 Das Objekt ist nicht mehr gültig.
- 1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

DataRoot**Siehe auch**

Eigenschaft: [DataRoot](#) als [XMLData](#) (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](#). Wenn [CurrentViewMode](#) nicht `spyViewGrid` oder `spyViewAuthentic` ist, ist möglicherweise ein [UpdateXMLData](#) erforderlich, um Zugriff auf die neuesten [XMLData](#) zu erhalten.

Fehler

- 1400 Das document-Objekt ist nicht mehr gültig.
- 1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

DocEditView**Siehe auch**

Methode: [DocEditView](#) als [DocEditView](#)

Beschreibung

Enthält eine Referenz auf das aktuelle Authentic Objekt.

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.

Encoding

Siehe auch

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](#).

Mit dieser Eigenschaft wird die Methode [SetEncoding](#) 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.

EndChanges

Siehe auch

Methode: [EndChanges](#)()

Beschreibung

Verwenden Sie die Methode [EndChanges](#) um alle Änderungen seit dem letzten Aufruf von [Document.StartChanges](#) anzuzeigen.

Fehler

- 1400 Das Objekt ist nicht mehr gültig.

ExecuteXQuery

Siehe auch

Methode: [ExecuteXQuery](#) (*strXMLFileName* als String)

Beschreibung

Führt die im Dokument enthaltene XQuery-Anweisung aus. Verwendet die XML-Datei, die als

XML-Quelle für die Transformation definiert wurde. Wenn Ihr XQuery-Script keine XML-Quelle verwendet, setzen Sie den Parameter `stXMLFileName` auf 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 `stXMLFileName` definierte Datei nicht oder ist ungültig.

ExportToDatabase

Siehe auch

Methode: `ExportToDatabase` (*pFromChild* als [XMLData](#), *pExportSettings* als [ExportSettings](#), *pDatabase* als [DatabaseConnection](#))

Beschreibung

`ExportToDatabase` exportiert das XML-Dokument beginnend mit dem Element `pFromChild`. Der Parameter `pExportSettings` definiert das Exportverhalten (siehe [Application.GetExportSettings](#)). Der Parameter `pDatabase` definiert das Ziel des Exports (siehe [Application.GetDatabaseSettings](#)). `UpdateXMLData()` wird eventuell indirekt benötigt, da Sie [XMLData](#) 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

```
Dim objDoc As Document
Set objDoc = objSpy.ActiveDocument

'set the behaviour of the export with ExportSettings
Dim objExpSettings As ExportSettings
Set objExpSettings = objSpy.GetExportSettings

'set the destination with DatabaseConnection
Dim objDB As DatabaseConnection
Set objDB = objSpy.GetDatabaseSettings

objDB.CreateMissingTables = True
objDB.CreateNew = True
objDB.File = "C:\Export.mdb"

objDoc.ExportToDatabase objDoc.RootElement, objExpSettings, objDB
If Err.Number <> 0 Then
    a = MsgBox("Error: " & (Err.Number - vbObjectError) & Chr(13) &
        "Description: " & Err.Description)
End If
```

ExportToText

Siehe auch

Methode: `ExportToText` (*pFromChild* als [XMLData](#), *pExportSettings* als [ExportSettings](#), *pTextSettings* als [TextImportExportSettings](#))

Beschreibung

`ExportToText` exportiert Tabellendaten aus dem Dokument beginnend mit `pFromChild` in eine oder mehrere Textdateien. Die Spalten der erzeugten Tabellen werden in alphabetischer Reihenfolge nach den Namen der Tabellenspalten generiert. Verwenden Sie [GetExportElementList](#), um zu sehen, welche Daten exportiert werden. Der Parameter `pExportSettings` definiert die näheren Einzelheiten für den Export. Setzen Sie die Eigenschaft [ExportSettings.ElementList](#) auf die - möglicherweise geänderte - Liste, die von [GetExportElementList](#) zurückgegeben wird, damit nicht alle enthaltenen Tabellen exportiert werden. Der Parameter `pTextSettings` definiert die Optionen für den Textexport und -import. Sie müssen die Eigenschaft [TextImportExportSettings.DestinationFolder](#) setzen, bevor Sie `ExportToText` aufrufen. [UpdateXMLData\(\)](#) wird eventuell indirekt benötigt, da Sie [XMLData](#) als Parameter an diese Funktion übergeben müssen.

Siehe auch [Import and export of data](#).

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
' -----

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.GetTextExportSettings
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
```

FullName

Siehe auch

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](#) und [SetPathName](#) überflüssig.

Fehler

- 1400 Das document-Objekt ist nicht mehr gültig.
- 1402 Es wurde ein leerer String als vollständiger Dateiname definiert.

GenerateDTDOrSchema

Siehe auch

Methode: [GenerateDTDOrSchema](#) (*nFormat* als [SPYDTDSchemaFormat](#), *nValuesList* als integer, *nDetection* als [SPYTypeDetection](#), *nFrequentElements* als [SPYFrequentElements](#))

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 werden? Verwenden Sie diesen Wert *spyGlobalComplexType* um sie als globale Elemente zu definieren. 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](#).

Beachten Sie, dass diese Funktionalität nicht nur in der ZIP-Ansicht verfügbar ist. Siehe [Document.CurrentViewMode](#) und [SPYViewModes](#).

Fehler

- 1400 Das Objekt ist nicht mehr gültig.
- 1407 Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.

GenerateDTDOrSchemaEx

Siehe auch

Methode: [GenerateDTDOrSchemaEx](#) (*objDlg* als [DTDSchemaGeneratorDlg](#)) als [Document](#)

Beschreibung

Verwenden Sie diese Methode, um automatisch eine DTD oder ein Schema für das aktuelle XML-Dokument zu generieren. Mit Hilfe eines [DTDSchemaGeneratorDlg](#) 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](#) und [SPYViewModes](#).

Fehler

- 1400 Das Objekt ist nicht mehr gültig.
- 1407 Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.

GenerateProgramCode

Methode: [GenerateProgramCode](#) (*objDlg* als [CodeGeneratorDlg](#))

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](#) 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.

GenerateSampleXML

Methode: [GenerateSampleXML](#) (*objDlg* als [GenerateSampleXMLDlg](#)) als [Document](#)

Beschreibung

Generiert eine XML-Beispieldatei, wenn das Dokument ein Schema oder eine DTD ist. Verwenden Sie [Dialogs.GenerateSampleXMLDlg](#) um eine initialisierte Gruppe von Optionen aufzurufen.

Verfügbar mit Typbibliothek Version 1.5

Fehler

- 1400 Das document-Objekt ist nicht mehr gültig.

GenerateSchemaDocumentation

Methode: [GenerateSchemaDocumentation](#) (*objDlg* als [SchemaDocumentationDlg](#))

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](#) um eine initialisierte Gruppe von Optionen aufzurufen. Sie müssen zumindest die Eigenschaft [SchemaDocumentationDlg.OutputFile](#) definieren, bevor Sie mit der Generierung beginnen können. Beachten Sie, dass diese Funktion nur für die Schema-Ansicht zur Verfügung steht. Siehe [Document.CurrentViewMode](#) und [SPYViewModes](#).

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

GenerateWSDL20Documentation

Methode: `GenerateWSDL20Documentation` (*objDlg* als `WSDL20DocumentationDlg`)

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](#) um eine initialisierte Gruppe von Optionen aufzurufen. Sie müssen zumindest die Eigenschaft `WSDL20DocumentationDlg.OutputFile` definieren, bevor Sie mit der Generierung beginnen können. Beachten Sie, dass diese Funktionalität nur für die WSDL-Ansicht zur Verfügung steht. Siehe [Document.CurrentViewMode](#) und [SPYViewModes](#).

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

GenerateWSDLDocumentation

Methode: `GenerateWSDLDocumentation` (*objDlg* als [WSDLDocumentationDlg](#))

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](#) um eine initialisierte Gruppe von Optionen aufzurufen. Sie müssen zumindest die Eigenschaft [SchemaDocumentationDlg.OutputFile](#) definieren, bevor Sie mit der Generierung beginnen können. Beachten Sie, dass diese Funktionalität nur für die WSDL-Ansicht zur Verfügung steht. Siehe [Document.CurrentViewMode](#) und [SPYViewModes](#).

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

GenerateXBRLDocumentation

Methode: `GenerateXBRLDocumentation` (*objDlg* als [XBRLDocumentationDlg](#))

Beschreibung

Generiert Dokumentation für eine WSDL-Definitionsdatei im HTML-, MS-Word- oder RTF-Format. Der Parameter "objDlg" dient zum Parametrisieren des Generierungsvorgangs. Verwenden Sie [DialogsXBRLDocumentationDlg](#) um eine initialisierte Gruppe von Optionen aufzurufen. Sie müssen zumindest die Eigenschaft `XBRLDocumentationDlg.OutputFile` definieren, bevor Sie mit der Generierung beginnen können. Beachten Sie, dass diese Funktionalität nur für die XBRL-Ansicht zur Verfügung steht. Siehe [Document.CurrentViewMode](#) und [SPYViewModes](#).

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

GetDBStructureList

Siehe auch

Methode: `GetDBStructureList` (*pDatabase* als [DatabaseConnection](#)) als [ElementList](#)

Beschreibung

`GetDBStructureList` erstellt anhand des Schema-Dokuments eine Sammlung von Elementen, für die Tabellen in der definierten Datenbank erstellt werden. Die Funktion gibt eine Sammlung von [ElementListItems](#) zurück, deren [ElementListItem.Name](#) Eigenschaften die Namen der Tabellen enthalten.

Siehe auch [CreateDBStructureFromXMLSchema](#).

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.

GetExportElementList

Siehe auch

Methode: `GetExportElementList` (*pFromChild* als [XMLData](#), *pExportSettings* als [ExportSettings](#)) als [ElementList](#)

Beschreibung

`GetExportElementList` erstellt eine Elementsammlung für den Export aus dem Dokument. Der Export wird durch die Einstellungen in `pExportSettings` definiert und beginnt beim Element `pFromChild`. Die Funktion gibt eine Sammlung von `ElementItems` zurück, wobei die Eigenschaften `ElementListItem.Name` die Namen der Tabellen enthalten, die aus dem Dokument exportiert werden können. Die Eigenschaft `ElementListItem.FieldCount` enthält die Anzahl der Spalten in der Tabelle. Die Eigenschaft `ElementListItem.RecordCount` enthält die Anzahl der Datensätze in der Tabelle. Die Eigenschaft `ElementListItem.ElementKind` wird nicht verwendet. `UpdateXMLData()` wird eventuell indirekt benötigt, da Sie `XMLData` als Parameter an diese Funktion übergeben müssen.

Siehe auch [Import und Export von Daten](#).

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.

GetPathName (nicht mehr verwendet)

Wird ersetzt durch [Document.FullName](#)

```
// ----- javascript sample -----  
// instead of:  
// strPathName = Application.ActiveDocument.GetPathName();  
// use now:  
strPathName = Application.ActiveDocument.FullName;
```

Siehe auch

Methode: `GetPathName()` als String

Beschreibung

Die Methode `GetPathName` ruft den Pfad des aktiven Dokuments auf.

Siehe auch [Document.SetPathName](#) (nicht mehr verwendet).

GridView

Siehe auch

Eigenschaft: `GridView` als [GridView](#)

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.

IsModified

Siehe auch

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.

IsValid

Siehe auch

Methode: `IsValid` (`strError` als Variant, `nErrorPos` als Variant, `pBadData` als Variant) als Boolean

Rückgabewert

True, wenn das Dokument gültig ist, false, wenn es nicht gültig ist.

Beschreibung

`IsValid()` validiert das Dokument anhand des damit verknüpften Schemas bzw. der DTD. `strError` gibt dieselbe Fehlermeldung zurück wie XMLSpy, wenn Sie die Datei im Editor validieren.

`nErrorPos` wird nicht mehr verwendet und steht nur noch aus Kompatibilitätsgründen zur Verfügung. `pBadData` ist eine Referenz auf das [XMLData](#) Objekt, das den Fehler ausgelöst hat. Es steht nur zur Verfügung, wenn das Dokument gerade in der Textansicht, der Grid-Ansicht oder der Authentic-Ansicht angezeigt wird.

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.

Beispiele

For an example written in VBA see Overview - Simple document access

```
' -----  
' XMLSpy scripting environment - VBScript  
' -----  
Dim errorText  
Dim errorPos  
Dim badData  
  
' validate the active document  
Set doc = Application.ActiveDocument  
valid = doc.IsValid(errorText, errorPos, badData)
```

```

If (Not valid) Then
    ' create the validation error message text
    Text = errorText

    If ( Not IsEmpty(badData) ) Then
        Text = Text & "(" & badData.Name & "/" & badData.TextValue & ")"
    End If

    Call MsgBox("validation error[" & errorPos & "]: " & Text)
End If

// -----
// XMLSpy scripting environment - JScript
// -----
// define as arrays to support their usage as return parameters
var errorText = new Array(1);
var errorPos = new Array(1);
var badData = new Array(1);

// validate the active document
var doc = Application.ActiveDocument;
var valid = doc.IsValid(errorText, errorPos, badData);

if (! valid)
{
    // compose the error description
    var text = errorText;

    // access that XMLData object only if filled in
    if ( badData[0] != null)
        text += "(" + badData[0].Name + "/" + badData[0].TextValue + ")"
;

    MsgBox("validation error[" + errorPos + "]: " + text);
}

```

IsWellFormed

Siehe auch

Methode: `IsWellFormed` (*pData* als `XMLData`, *bWithChildren* als `Boolean`, *strError* als `Variant`, *nErrorPos* als `Variant`, *pBadXMLData* als `Variant`) als `Boolean`

Rückgabewert

True, wenn das Dokument wohlgeformt ist.

Beschreibung

`IsWellFormed` überprüft das Dokument beginnend beim Element `pData` auf Wohlgeformtheit.

Wenn das Dokument nicht wohlgeformt ist, enthält `strError` eine Fehlermeldung, `nErrorPos` die Position der Datei und `pBadXMLData` enthält 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](#).

Name

Siehe auch

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](#).

Fehler

- 1400 Das document-Objekt ist nicht mehr gültig.
- 1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Parent

Siehe auch

Eigenschaft: `Parent` als [Documents](#) (schreibgeschützt)

Beschreibung

Ruft das übergeordnete Objekt des Document-Objekts auf.

Fehler

- 1400 Das document-Objekt ist nicht mehr gültig.
- 1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Eigenschaft: `Parent` als [Application](#) (schreibgeschützt)

Path

Siehe auch

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](#).

Fehler

- 1400 Das document-Objekt ist nicht mehr gültig.
- 1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

RootElement

Siehe auch

Eigenschaft: `RootElement` als [XMLData](#) (schreibgeschützt)

Beschreibung

Über die Eigenschaft `RootElement` haben 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](#). Wenn `CurrentViewMode` nicht `spyViewGrid` oder `spyViewAuthentic` ist, ist möglicherweise ein [UpdateXMLData](#) erforderlich, um Zugriff auf die neuesten [XMLData](#) zu erhalten.

Fehler

- 1400 Das document-Objekt ist nicht mehr gültig.
- 1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Save**Siehe auch**

Methode: [Save](#)()

Beschreibung

Die Methode schreibt alle Änderungen des Dokuments in die dazugehörige Datei. Siehe auch [Document.FullName](#).

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.

SaveAs**Siehe auch**

Methode: [SaveAs](#) ([strFileName](#) als String)

Beschreibung

Speichert das Dokument in der angegebenen Datei. Wenn das Dokument erfolgreich gespeichert werden konnte, wird die Eigenschaft [FullName](#) 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.

Saved**Siehe auch**

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](#) zurück.

Fehler

- 1400 Das document-Objekt ist nicht mehr gültig.
- 1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

SaveInString

Siehe auch

Methode: `SaveInString` (*pData* als [XMLData](#), *bMarked* als Boolean) als String

Parameter

pData

[XMLData](#) Element als Start. Setzen Sie *pData* auf [Document.RootElement](#), 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\(\)](#) wird eventuell indirekt benötigt, da Sie [XMLData](#) 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.

SaveToURL

Siehe auch

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.

SetActiveDocument

Siehe auch

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.

Document.SetEncoding (nicht mehr verwendet)

Wird ersetzt durch [Document.Encoding](#)

```
// ----- javascript sample -----  
// instead of:  
// Application.ActiveDocument.SetEncoding("UTF-16");  
// use now:  
Application.ActiveDocument.Encoding = "UTF-16";
```

Siehe auch

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

SetExternallsValid

Siehe auch

Methode: `SetExternallsValid` (*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](#) Handler abgefragt.

Verfügbar mit Typbibliothek Version 1.5

Fehler

1400 Das Objekt ist nicht mehr gültig.

Document.SetPathName (nicht mehr verwendet)

Wird ersetzt durch [Document.FullName](#)

```
// ----- javascript sample -----  
// instead of:  
// Application.ActiveDocument.SetPathName("C:\\myXMLFiles\\test.xml");  
// use now:  
Application.ActiveDocument.FullName = "C:\\myXMLFiles\\test.xml";
```

Siehe auch

Methode: `SetPathName` (*strPath* als String)

Beschreibung

Die Methode `SetPathName` definiert den Pfad des aktiven Dokuments. `SetPathName` kopiert den String nur und überprüft nicht, ob der Pfad gültig ist. Alle nachfolgenden Speicheroperationen erfolgen in dieser Datei.

StartChanges

Siehe auch

Methode: `StartChanges`()

Beschreibung

Nach Ausführung von `StartChanges` aktualisiert XMLSpy seine Editor-Fenster erst, wenn [Document.EndChanges](#) aufgerufen wird. Dadurch wird die Geschwindigkeit bei der Ausführung komplexer Operationen an der XML-Struktur verbessert.

Fehler

1400 Das Objekt ist nicht mehr gültig.

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](#) Event angezeigt werden.

Fehler

- 1400 Das Objekt ist nicht mehr gültig.
- 1407 Ungültiger Parameter oder es wurde für den Rückgabeparameter eine ungültige Adresse angegeben.

SwitchViewMode

Siehe auch

Methode: [SwitchViewMode](#) (*nMode* als [SPYViewModes](#)) 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](#).

Fehler

- 1400 Das Objekt ist nicht mehr gültig.
- 1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.
- 1417 Es wurde ein ungültiger Ansichtsmodus definiert.

TextView

Siehe auch

Eigenschaft: [TextView](#) als [TextView](#)

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.

Title

Siehe auch

Eigenschaft: [Title](#) als String (schreibgeschützt)

Beschreibung

[Title](#) enthält den Dateinamen des Dokuments. Um den Pfad und Dateinamen der Datei abzurufen, verwenden Sie [FullName](#).

Fehler

- 1400 Das document-Objekt ist nicht mehr gültig.
- 1407 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

TransformXSL**Siehe auch**

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](#).

Fehler

- 1400 Das document-Objekt ist nicht mehr gültig.
- 1411 Fehler bei der Transformation.

TransformXSLEx**Siehe auch**

Method: [TransformXSL\(*nAction* as \[SPYDialogAction\]\(#\)\)](#)

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](#).

Fehler

- 1400 Das document-Objekt ist nicht mehr gültig.
- 1411 Fehler bei der Transformation.

TransformXSLFO**Siehe auch**

Methode: [TransformXSLFO\(\)](#)

Beschreibung

[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](#). 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.

TreatXBRLInconsistencies AsErrors

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.

UpdateViews

Siehe auch

Methode: [UpdateViews\(\)](#)

Beschreibung

Rufen Sie zum Neuzeichnen der Grid-Ansicht und der Strukturansicht `UpdateViews` auf. 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.

UpdateXMLData

Siehe auch

Methode: [UpdateXMLData\(\)](#) als Boolean

Beschreibung

Die [XMLData](#)-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](#) `spyViewGrid` oder `spyViewAuthentic` ist, weil die [XMLData](#) in diesen Ansichten automatisch aktualisiert werden.

Verfügbar mit Typbibliothek Version 1.5

Fehler

- 1400 Das document-Objekt ist nicht mehr gültig.

3.2.11 Documents

Siehe auch

Eigenschaften

[Count](#)
[Item](#)

Methoden

[NewFile](#)
[OpenFile](#)
[OpenURL](#)

[OpenURLDialog](#)
[NewFileFromText](#)

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

```
' -----
' XMLSpy scripting environment - VBScript
' iterate through open documents
' -----

Dim objDocuments
Set objDocuments = Application.Documents

For Each objDoc In objDocuments
    'do something useful with your document
    objDoc.SetActiveDocument()
Next

// -----
// XMLSpy scripting environment - JScript
// close all open documents
// -----
for ( var iter = new Enumerator ( Application.Documents );
      ! iter.atEnd();
      iter.moveNext() )
{
    // MsgBox ("Closing file " + iter.item().Name);
    iter.item().Close ( true );
}
```

Count

Siehe auch

Eigenschaft: [Count](#) als long

Beschreibung

Anzahl der offenen Dokumente.

Item

Siehe auch

Methode: [Item](#) (*n* als long) als [Document](#)

Beschreibung

Ruft das Dokument mit dem Index *n* in dieser Sammlung ab. Der Index ist 1-basiert.

NewAuthenticFile

Siehe auch

Methode: [NewAuthenticFile](#) (*strSPSPath* als String, *strXMLPath* als String) als [Document](#)

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.

NewFile**Siehe auch**

Methode: `NewFile` (`strFile` als String, `strType` als String) als [Document](#)

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

`NewFile` erstellt eine neue Datei vom Typ `strType` (d.h. "xml"). Die neu erstellte Datei ist auch das ActiveDocument.

NewFileFromText**Siehe auch**

Methode: `NewFileFromText` (`strText` als String, `strType` als String) als [Document](#)

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.

OpenAuthenticFile

Siehe auch

Methode: `OpenAuthenticFile` (*strSPSPath* als String, *strXMLPath* als String) als [Document](#)

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.

OpenFile

Siehe auch

Methode: `OpenFile` (*strPath* als String, *bDialog* als Boolean) als [Document](#)

Parameter

strPath

Pfad und Dateiname der zu öffnenden Datei.

bDialog

Zeigt die Dialogfelder für die Benutzereingaben an..

Rückgabewert

Gibt die geöffnete Datei zurück, wenn die Methode erfolgreich war.

Beschreibung

`OpenFile` öffnet die Datei *strPath* . Wenn *bDialog* TRUE ist, wird ein Dateidialogfeld angezeigt.

Beispiel

```
Dim objDoc As Document
Set objDoc = objSpy.Documents.OpenFile(strFile, False)
```

OpenURL

Siehe auch

Methode: `OpenURL` (*strURL* als String, *nURLType* als [SPYURLTypes](#), *nLoading* als [SPYLoading](#), *strUser* als String, *strPassword* als String) als [Document](#)

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`.

OpenURLDialog

Siehe auch

Methode: `OpenURLDialog` (`strURL` als String, `nURLType` als `SPYURLTypes`, `nLoading` als `SPYLoading`, `strUser` als String, `strPassword` als String) als `Document`

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

`OpenURLDialog` zeigt dem Benutzer das Dialogfeld "URL öffnen" an und setzt in die Eingabefelder die angegebenen Parameter ein.

3.2.12 DTDSchemaGeneratorDlg

Siehe auch

Eigenschaften und Methoden

Standard-Automation-Eigenschaften

[Application](#)

[Parent](#)

[DTDSchemaFormat](#)

[ValueList](#)

[TypeDetection](#)

[FrequentElements](#)

[MergeAllEqualNamed](#)

[ResolveEntities](#)

[AttributeDefinition](#)

[GlobalAttributes](#)

[OnlyStringEnums](#)

[MaxEnumLength](#)

[OutputPath](#)

[OutputPathDialogAction](#)

Beschreibung

Mit Hilfe dieses Objekts können Sie die Generierung eines Schemas oder einer DTD konfigurieren. Die Methode [GenerateDTDOrSchemaEx](#) erwartet einen [DTDSchemaGeneratorDlg](#) als Parameter, um die Generierung sowie die damit in Zusammenhang stehenden Interaktionen mit dem Benutzer zu konfigurieren.

Application

Eigenschaft: [Application](#) als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

- 3000 Das Objekt ist nicht mehr gültig.
- 3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

AttributeTypeDefinition

Eigenschaft: [AttributeTypeDefinition](#) als [SPYAttributeTypeDefinition](#)

Beschreibung

Definiert, wie Attributdefinitionen zusammengeführt werden.

Fehler

- 3000 Das Objekt ist nicht mehr gültig.
- 3001 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

DTDSchemaFormat

Eigenschaft: `DTDSchemaFormat` als [SPYDTDSchemaFormat](#)

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.

FrequentElements

Eigenschaft: `FrequentElements` als [SPYFrequentElements](#)

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.

GlobalAttributes

Eigenschaft: `GlobalAttributes` als 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.

MaxEnumLength

Eigenschaft: `MaxEnumLength` als 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.

MergeAllEqualNamed

Eigenschaft: `MergeAllEqualNamed` als 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.

OnlyStringEnums

Eigenschaft: [OnlyStringEnums](#) als 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.

OutputPath

Eigenschaft: [OutputPath](#) als 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.

OutputPathDialogAction

Eigenschaft: [OutputPathDialogAction](#) als [SPYDialogAction](#)

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](#) 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.

Parent

Eigenschaft: [Parent](#) als [Dialogs](#) (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.

ResolveEntities

Eigenschaft: [ResolveEntities](#) als 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.

TypeDetection

Eigenschaft: [TypeDetection](#) als [SPYTypeDetection](#)

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.

ValueList

Eigenschaft: [ValueList](#) als 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.

3.2.13 ElementList

Siehe auch

Eigenschaften

[Count](#)
[Item](#)

Methoden

[RemoveElement](#)

Beschreibung

Elementlisten werden beim Export und Import von Daten für verschiedene Zwecke verwendet. Je nach Verwendungszweck werden unterschiedliche Eigenschaften von [ElementListItem](#) verwendet.

Die Methode kann die folgenden Eigenschaften enthalten

- eine Liste von Tabellennamen, die durch einen Aufruf von [Application.GetDatabaseTables](#) zurückgegeben werden.
- eine Liste von Feldnamen, die durch einen Aufruf von [Application.GetDatabaseImportElementList](#) oder [Application.GetTextImportElementList](#) zurückgegeben werden,
- eine Feldnamenfilterliste, die in [Application.ImportFromDatabase](#) und [Application.ImportFromText](#) verwendet wird,
- eine Liste von Tabellennamen und der Anzahl ihrer Zeilen und Spalten, die durch Aufrufen von [GetExportElementList](#) zurückgegeben werden oder
- eine in [Document.ExportToDatabase](#) und [Document.ExportToText](#) verwendete Feldnamenfilterliste.

Count

Siehe auch

Eigenschaft: [Count](#) als long (schreibgeschützt)

Beschreibung

Anzahl der Elemente in dieser Sammlung.

Item

Siehe auch

Methode: [Item](#)([n](#) als long) als [ElementListItem](#)

Beschreibung

Ruft das Element mit dem Index [n](#) aus dieser Sammlung ab. Das erste Element hat den Index 1.

RemoveElement

Siehe auch

Methoden: [RemoveElement](#)([Index](#) als long)

Beschreibung

[RemoveElement](#) entfernt den Elementindex aus der Sammlung. Das erste Element hat den Index 1.

3.2.14 ElementListItem

Siehe auch

Eigenschaften

[Name](#)

[ElementKind](#)

[FieldCount](#)

[RecordCount](#)

Beschreibung

Ein Element in einer [ElementList](#). Die Verwendung seiner Eigenschaften ist vom Verwendungszweck der Elementliste abhängig. Nähere Informationen dazu siehe [ElementList](#).

ElementKind

Siehe auch

Eigenschaft: [ElementKind](#) als [SPYXMLDataKind](#)

Beschreibung

Definiert, ob ein Feld als XML-Element (Datenwert von `spyXMLDataElement`) oder als Attribut (Datenwert von `spyXMLDataAttr`) importiert werden soll.

FieldCount

Siehe auch

Eigenschaft: [FieldCount](#) als long (schreibgeschützt)

Beschreibung

Anzahl der Felder (d.h. Spalten) in der durch dieses Element beschriebenen Tabelle. Diese Eigenschaft ist nur nach einem Aufruf von [DocumentGetExportElementList](#) gültig.

Name

Siehe auch

Eigenschaft: [Name](#) als String (schreibgeschützt)

Beschreibung

Name des Elements. Dies ist je nach Verwendungszweck der Elementliste entweder der Name einer Tabelle oder eines Felds.

RecordCount

Siehe auch

Eigenschaft: [RecordCount](#) als long (schreibgeschützt)

Beschreibung

Anzahl der Datensätze (d.h. Zeilen) in der durch dieses Element beschriebenen Tabelle. Diese Eigenschaft ist nur nach einem Aufruf von [DocumentGetExportElementList](#) gültig.

3.2.15 ExportSettings

Siehe auch

Eigenschaften

[ElementList](#)

[EntitiesToText](#)

[ExportAllElements](#)
[SubLevelLimit](#)

[FromAttributes](#)
[FromSingleSubElements](#)
[FromTextValues](#)

[CreateKeys](#)
[IndependentPrimaryKey](#)

[Namespace](#)

[ExportCompleteXML](#)
[StartFromElement](#)

Beschreibung

[ExportSettings](#) enthält 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](#).

CreateKeys

Siehe auch

Eigenschaft: [CreateKeys](#) als Boolean

Beschreibung

Diese Eigenschaft aktiviert bzw. deaktiviert die Erstellung von Schlüsseln (d.h. Primär- und Sekundärschlüssel). Die Standardeinstellung ist "True".

ElementList

Siehe auch

Eigenschaft: [ElementList](#) als [ElementList](#)

Beschreibung

Standardeinstellung ist eine leere Liste. In dieser Elementliste wird definiert, welche Felder exportiert werden. Um die Liste der verfügbaren Felder abzurufen, verwenden Sie [DocumentGetExportElementList](#). Sie können Spalten vom Export ausnehmen, indem Sie Elemente mit [ElementList.RemoveElement](#) aus dieser Liste entfernen, bevor Sie sie an [DocumentExportToDatabase](#) oder [DocumentExportToText](#) übergeben.

EntitiesToText

Siehe auch

Eigenschaft: [EntitiesToText](#) als Boolean

Beschreibung

Definiert, ob XML-Entities in Text konvertiert oder unverändert belassen werden sollen. Die Standardeinstellung ist True.

ExportAllElements

Siehe auch

Eigenschaft: [ExportAllElements](#) als Boolean

Beschreibung

Wenn diese Eigenschaft auf `true` gesetzt ist, werden alle Elemente im Dokument exportiert. Wenn sie auf `false` gesetzt ist, wird die Anzahl der zu exportierenden Unterebenen mittels [ExportSettings.SubLevelLimit](#) eingeschränkt. Die Standardeinstellung ist `true`.

ExportCompleteXML

Siehe auch

Eigenschaft: [ExportCompleteXML](#) als Boolean

Beschreibung

Definiert, ob die komplette XML-Datei oder nur das durch [StartFromElement](#) definierte Element und seine untergeordneten Elemente exportiert werden sollen. Standardeinstellung ist True.

FromAttributes

Siehe auch

Eigenschaft: [FromAttributes](#) als Boolean

Beschreibung

Setzen Sie [FromAttributes](#) auf "false", wenn keine Exportdaten anhand von Attributen erstellt werden sollen. Die Standardeinstellung ist True.

FromSingleSubElements

Siehe auch

Eigenschaft: [FromSingleSubElements](#) als Boolean

Beschreibung

Setzen Sie [FromSingleSubElements](#) auf "false", wenn keine Exportdaten von Elementen erstellt werden sollen. Die Standardeinstellung ist True.

FromTextValues

Siehe auch

Eigenschaft: [FromTextValues](#) als Boolean

Beschreibung

Setzen Sie [FromTextValues](#) auf "false", wenn keine Exportdaten anhand von Textwerten erstellt werden sollen. Die Standardeinstellung ist "True".

IndependentPrimaryKey

Siehe auch

Eigenschaft: [IndependentPrimaryKey](#) als Boolean

Beschreibung

Aktiviert bzw. deaktiviert die Erstellung unabhängiger Primärschlüsselzähler für alle Elemente. Wenn [ExportSettings.CreateKeys](#) "False" ist, wird diese Eigenschaft ignoriert. Die Standardeinstellung ist "True".

Namespace

Siehe auch

Eigenschaft: [Namespace](#) als [SPYExportNamespace](#)

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`.

StartFromElement

Siehe auch

Eigenschaft: [StartFromElement](#) als String

Beschreibung

Definiert das Startelement für den Export. Diese Eigenschaft wird nur berücksichtigt, wenn [ExportCompleteXML](#) "false" ist.

SubLevelLimit

Siehe auch

Eigenschaft: [SubLevelLimit](#) als Integer

Beschreibung

Definiert die Anzahl der Unterebenen, die exportiert werden soll. Diese Eigenschaft wird ignoriert, wenn [ExportSettings.ExportAllElements](#) `true` ist.

3.2.16 FileSelectionDlg

Siehe auch

Eigenschaften und Methoden

Standard-Automation-Eigenschaften

[Application](#)

[Parent](#)

Dialogfeldeigenschaften

[FullName](#)

Annahme oder Abbruch der Aktion, die das Event verursacht hat.

[DialogAction](#)

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](#) können Sie den Dateipfad auswählen oder ändern und mit Hilfe der Eigenschaft [FileSelectionDlg.DialogAction](#) können Sie die Aktion, die das Event verursacht hat, abrechnen oder akzeptieren.

Application

Siehe auch

Eigenschaft: [Application](#) als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

- 2400 Das Objekt ist nicht mehr gültig.
- 2401 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

DialogAction

Eigenschaft: [DialogAction](#) als [SPYDialogAction](#)

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.

FullName

Eigenschaft: [FullName](#) als 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.

Parent**Siehe auch**

Eigenschaft: [Parent](#) als [Dialogs](#) (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.

3.2.17 FindInFilesDlg**Siehe auch****Eigenschaften und Methoden**

Standard-Automation-Eigenschaften

[Application](#)

[Parent](#)

[Find](#)

[RegularExpression](#)

[Replace](#)

[DoReplace](#)

[ReplaceOnDisk](#)

[MatchWholeWord](#)

[MatchCase](#)

[SearchLocation](#)

[StartFolder](#)

[IncludeSubfolders](#)

[SearchInProjectFilesDoExternal](#)

[FileExtension](#)

[AdvancedXMLSearch](#)

[XMLElementNames](#)

[XMLElementContents](#)

[XMLAttributeNames](#)

[XMLAttributeContents](#)

[XMLComments](#)

[XMLCDATA](#)

[XMLPI](#)

[XMLRest](#)

[ShowResult](#)

Beschreibung

Mit Hilfe dieses Objekts können Sie die Suche (oder die Ersetzung) von Strings in Dateien konfigurieren. Die Methode [FindInFiles](#) erwartet einen [FindInFilesDlg](#) als Parameter.

AdvancedXMLSearch

Eigenschaft: [AdvancedXMLSearch](#) als Boolean

Beschreibung

Legt fest, ob die XML-Sucheigenschaften ([XMLElementNames](#), [XMLElementContents](#),

[XMLAttributeNames](#), [XMLAttributeContents](#), [XMLComments](#), [XMLCDATA](#), [XMLPI](#) und [XMLRest](#)) berücksichtigt werden. Die Standardeinstellung ist "false".

Fehler

- 3500 Das Objekt ist nicht mehr gültig.
- 3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Application

Eigenschaft: [Application](#) als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

- 3500 Das Objekt ist nicht mehr gültig.
- 3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

DoReplace

Eigenschaft: [DoReplace](#) als Boolean

Beschreibung

Legt fest, ob der gefundene String durch den in [Replace](#) 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.

FileExtension

Eigenschaft: [FileExtension](#) als 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.

Find

Eigenschaft: [Find](#) als String

Beschreibung

Definiert den zu suchenden String.

Fehler

- 3500 Das Objekt ist nicht mehr gültig.
- 3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

IncludeSubfolders

Eigenschaft: [IncludeSubfolders](#) als 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.

MatchCase

Eigenschaft: [MatchCase](#) als 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.

MatchWholeWord

Eigenschaft: [MatchWholeWord](#) als 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.

Parent

Eigenschaft: [Parent](#) als [Dialogs](#) (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.

RegularExpression

Eigenschaft: [RegularExpression](#) als Boolean

Beschreibung

Definiert, ob [Find](#) 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.

Replace

Eigenschaft: [Replace](#) als String

Beschreibung

Definiert den Ersetzungsstring. Der gefundene String wird nur ersetzt, wenn [DoReplace](#) auf "true" gesetzt ist.

Fehler

- 3500 Das Objekt ist nicht mehr gültig.
- 3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

ReplaceOnDisk

Eigenschaft: [ReplaceOnDisk](#) als 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.

SearchInProjectFilesDoExternal

Eigenschaft: [SearchInProjectFilesDoExternal](#) als 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.

SearchLocation

Eigenschaft: [SearchLocation](#) als [SPYFindInFilesSearchLocation](#)

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.

ShowResult

Eigenschaft: [ShowResult](#) als 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.

StartFolder

Eigenschaft: [StartFolder](#) als 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.

XMLAttributeContents

Eigenschaft: [XMLAttributeContents](#) als Boolean

Beschreibung

Legt fest ob Attributinhalt durchsucht werden soll, wenn [AdvancedXMLSearch](#) true ist. Die Standardeinstellung ist "True".

Fehler

- 3500 Das Objekt ist nicht mehr gültig.
- 3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

XMLAttributeNames

Eigenschaft: [XMLAttributeNames](#) als Boolean

Beschreibung

Definiert, ob Attributnamen durchsucht werden sollen, wenn [AdvancedXMLSearch](#) true ist. Die Standardeinstellung ist "True".

Fehler

- 3500 Das Objekt ist nicht mehr gültig.
- 3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

XMLCDATA

Eigenschaft: [XMLCDATA](#) als Boolean

Beschreibung

Legt fest, ob CDATA Tags durchsucht werden sollen, wenn [AdvancedXMLSearch](#) true ist. Die Standardeinstellung ist "True".

Fehler

- 3500 Das Objekt ist nicht mehr gültig.
- 3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

XMLComments

Eigenschaft: [XMLComments](#) als Boolean

Beschreibung

Legt fest, ob Kommentare durchsucht werden sollen, wenn [AdvancedXMLSearch](#) true ist. Die

Standardeinstellung ist "True".

Fehler

- 3500 Das Objekt ist nicht mehr gültig.
- 3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

XMLElementContents

Eigenschaft: [XMLElementContents](#) als Boolean

Beschreibung

Legt fest, ob der Elementinhalt durchsucht werden soll, wenn [AdvancedXMLSearch](#) true ist. Die Standardeinstellung ist "True".

Fehler

- 3500 Das Objekt ist nicht mehr gültig.
- 3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

XMLElementNames

Eigenschaft: [XMLElementNames](#) als Boolean

Beschreibung

Legt fest, ob Elementnamen durchsucht werden sollen, wenn [AdvancedXMLSearch](#) true ist. Die Standardeinstellung ist "True".

Fehler

- 3500 Das Objekt ist nicht mehr gültig.
- 3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

XMLPI

Eigenschaft: [XMLPI](#) als Boolean

Beschreibung

Legt fest, ob XML-Verarbeitungsanweisungen durchsucht werden sollen, wenn [AdvancedXMLSearch](#) true ist. Die Standardeinstellung ist "True".

Fehler

- 3500 Das Objekt ist nicht mehr gültig.
- 3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

XMLRest

Eigenschaft: [XMLRest](#) als Boolean

Beschreibung

Legt fest, ob der Rest der XML-Datei (der nicht durch die anderen XML-Sucheigenschaften berücksichtigt wird) durchsucht werden soll, wenn [AdvancedXMLSearch](#) true ist. Die Standardeinstellung ist "True".

Fehler

- 3500 Das Objekt ist nicht mehr gültig.
- 3501 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

3.2.18 FindInFilesResult

Siehe auch

Eigenschaften und Methoden

Standard-Automation-Eigenschaften

[Application](#)

[Parent](#)

[Count](#)

[Item](#)

[Path](#)

[Document](#)

Beschreibung

Dieses Objekt repräsentiert eine Datei, die den Suchkriterien entsprochen hat. Es enthält eine Liste von [FindInFilesResultMatch](#) Objekten, die die Suchposition beschreiben.

Application

Eigenschaft: [Application](#) als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

- 3700 Das Objekt ist nicht mehr gültig.
- 3701 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Count

Eigenschaft: [Count](#) als long (schreibgeschützt)

Beschreibung

Anzahl der Elemente in dieser Sammlung.

Document

Eigenschaft: [Path](#) als [Document](#) (schreibgeschützt)

Beschreibung

Diese Eigenschaft gibt das [Document](#)-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.

Item

Methode: [Item](#)(n als long) als [FindInFilesResultMatch](#)

Beschreibung

Ruft das Element mit dem Index n aus dieser Sammlung ab. Das erste Element hat den Index 1.

Parent

Eigenschaft: `Parent` als [FindInFilesResults](#) (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.

Path

Eigenschaft: `Path` als 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.

3.2.19 FindInFilesResultMatch

Siehe auch

Eigenschaften und Methoden

Standard-Automation-Eigenschaften

[Application](#)

[Parent](#)

[Line](#)

[Position](#)

[Length](#)

[LineText](#)

[Replaced](#)

Beschreibung

Enthält die genaue Position des gesuchten String in der Datei.

Application

Eigenschaft: `Application` als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

- 3800 Das Objekt ist nicht mehr gültig.

3801 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Length

Eigenschaft: [Length](#) als 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.

Line

Eigenschaft: [Line](#) als 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.

LineText

Eigenschaft: [LineText](#) als 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.

Parent

Eigenschaft: [Parent](#) als [FindInFilesResult](#) (schreibgeschützt)

Beschreibung

Ruft das dem Objekt übergeordnete Objekt auf.

Fehler

- 3800 Das Objekt ist nicht mehr gültig.
- 3801 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Position

Eigenschaft: [Position](#) als 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.

Replaced

Eigenschaft: [Replaced](#) als 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.

3.2.20 FindInFilesResults

Siehe auch

Eigenschaften und Methoden

Standard-Automation-Eigenschaften

[Application](#)

[Parent](#)

[Count](#)

[Item](#)

Beschreibung

Dies ist das Ergebnis der [FindInFiles](#) Methode. Es handelt sich hierbei um eine Liste von [FindInFilesResult](#) Objekten.

Application

Eigenschaft: [Application](#) als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

- 3600 Das Objekt ist nicht mehr gültig.
- 3601 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Count

Eigenschaft: [Count](#) als long (schreibgeschützt)

Beschreibung

Anzahl der Elemente in dieser Sammlung.

Item

Methode: `Item(n` als long) als [FindInFilesResult](#)

Beschreibung

Ruft das Element mit dem Index `n` aus dieser Sammlung ab. Das erste Element hat den Index 1.

Parent

Eigenschaft: `Parent` als [Application](#) (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.

3.2.21 GenerateSampleXMLDlg

Siehe auch

Eigenschaften und Methoden

Standard-Automation-Eigenschaften

[Application](#)

[Parent](#)

[NonMandatoryAttributes](#)

[RepeatCount](#)

[FillAttributesWithSampleData](#)

[FillElementsWithSampleData](#)

[ContentOfNillableElementsIsNonMandatory](#)

[TryToUseNonAbstractTypes](#)

[Optimization](#)

[SchemaOrDTDAssignment](#)

[LocalNameOfRootElement](#)

[NamespaceURIOfRootElement](#)

[OptionsDialogAction](#)

Nicht mehr unterstützte Eigenschaften

[NonMandatoryElements - nicht mehr verwendet](#)

[TakeFirstChoice - nicht mehr verwendet](#)

[FillWithSampleData - nicht mehr verwendet](#)

Beschreibung

Dient zum Setzen der Parameter für die Generierung von XML-Beispielinstanzen auf Basis eines W3C-Schemas oder einer DTD.

Application

Eigenschaft: `Application` als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

- 2200 Das Objekt ist nicht mehr gültig.
- 2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Parent

Eigenschaft: `Parent` als `Dialogs` (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.

NonMandatoryAttributes

Eigenschaft: `NonMandatoryAttributes` als 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.

NonMandatoryElements - obsolete

Eigenschaft: `NonMandatoryElements` als Boolean

Beschreibung

Diese Eigenschaft wird nicht mehr verwendet. Verwenden Sie stattdessen [Optimization](#).

Fehler

- 0001 Die Eigenschaft steht nicht mehr zur Verfügung.

TakeFirstChoice - obsolete

Eigenschaft: `TakeFirstChoice` als Boolean

Beschreibung

Diese Eigenschaft wird nicht mehr verwendet.

Fehler

- 0001 Die Eigenschaft steht nicht mehr zur Verfügung.

RepeatCount

Eigenschaft: `RepeatCount` als 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.

FillWithSampleData - obsolete

Eigenschaft: [FillWithSampleData](#) als Boolean

Beschreibung

Diese Eigenschaft wird nicht mehr verwendet. Verwenden Sie stattdessen [FillAttributesWithSampleData](#) und [FillElementsWithSampleData](#), instead.

Fehler

- 0001 Die Eigenschaft steht nicht mehr zur Verfügung.

FillElementsWithSampleData

Eigenschaft: [FillElementsWithSampleData](#) als Boolean

Beschreibung

Wenn der Wert "true" ist, haben Elemente Beispielinhalt.

Fehler

- 2200 Das Objekt ist nicht mehr gültig.
- 2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

FillAttributesWithSampleData

Eigenschaft: [FillAttributesWithSampleData](#) als 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.

ContentOfNillableElementsIsNonMandatory

Eigenschaft: [ContentOfNillableElementsIsNonMandatory](#) als 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.

TryToUseNonAbstractTypes

Eigenschaft: [TryToUseNonAbstractTypes](#) als 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.

Optimization

Eigenschaft: [Optimization](#) als [SPYSampleXMLGenerationOptimization](#)

Beschreibung

Definiert, welche Elemente generiert werden.

Fehler

- 2200 Das Objekt ist nicht mehr gültig.
- 2201 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

SchemaOrDTDAssignment

Eigenschaft: [SchemaOrDTDAssignment](#) als [SPYSampleXMLGenerationSchemaOrDTDAssignment](#)

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.

LocalNameOfRootElement

Eigenschaft: [LocalNameOfRootElement](#) als 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.

NamespaceURIOfRootElement

Eigenschaft: [NamespaceURIOfRootElement](#) als 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.

OptionsDialogAction

Eigenschaft: `OptionsDialogAction` als [SPYDialogAction](#)

Beschreibung

Damit das 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.

3.2.22 GridView

Siehe auch

Methoden

[Deselect](#)

[Select](#)

[SetFocus](#)

Eigenschaften

[CurrentFocus](#)

[IsVisible](#)

Beschreibung

GridView Class

Events

OnBeforeDrag

Siehe auch

Event: `OnBeforeDrag()` als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_BeforeDrag()
    'On_BeforeStartEditing=False 'to prohibit dragging
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_BeforeDrag()
{
    //to enable 'to prohibit dragging'
}
```

XMLSpy 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.

OnBeforeDrop**Siehe auch**

Event: OnBeforeDrop(*objXMLData* als [XMLData](#)) als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_BeforeDrop(objXMLData)
    'On_BeforeStartEditing=False 'prohibit dropping
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_BeforeDrop(objXMLData)
{
    //return false; 'prohibit dropping
}
```

XMLSpy 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.

OnBeforeStartEditing**Siehe auch**

Event: OnBeforeStartEditing(*objXMLData* als [XMLData](#), *bEditingName* als Boolean) als Boolean

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_BeforeStartEditing(objXMLData, bEditingName)
    'On_BeforeStartEditing=False 'prohibit editing the field
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_BeforeStartEditing(objXMLData, bEditingName)
{
```

```

        //true: bEditingName
    }

```

XMLSpy IDE Plugin:

```
IXMLSpyPlugin.OnEvent (1, ...) //nEvent=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.

OnEditingFinished**Siehe auch**

Event: *OnEditingFinished*(*objXMLData* als [XMLData](#), *bEditingName* als Boolean)

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_EditingFinished(objXMLData, bEditingName)
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_EditingFinished(objXMLData, bEditingName)
{
}

```

XMLSpy IDE Plugin:

```
IXMLSpyPlugin.OnEvent (2, ...) //nEvent=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.

OnFocusChanged**Siehe auch**

Event: *OnFocusChanged*(*objXMLData* als [XMLData](#), *bSetFocus* als Boolean, *bEditingName* als Boolean)

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_FocusChanged(objXMLData, bSetFocus, bEditingName)
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_FocusChanged(objXMLData, bSetFocus, bEditingName)
{
}

```

XMLSpy IDE Plugin:

```
IXMLSpyPlugin.OnEvent (3, ...) //nEvent=3
```

Beschreibung

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.

CurrentFocus

Siehe auch

Eigenschaft: `CurrentFocus` als [XMLData](#)

Beschreibung

Enthält das XML-Element mit dem aktuellen Fokus. Diese Eigenschaft ist schreibgeschützt.

Deselect

Siehe auch

Methode: `Deselect(pData` als [XMLData](#))

Beschreibung

Hebt die Auswahl des Elements `pData` in der Grid-Ansicht auf.

IsVisible

Siehe auch

Eigenschaft: `IsVisible` als Boolean

Beschreibung

True, wenn die Grid-Ansicht die aktive Ansicht des Dokuments ist. Diese Eigenschaft ist schreibgeschützt.

Select

Siehe auch

Methode: `Select (pData` als [XMLData](#))

Beschreibung

Wählt das XML-Element `pData` in der Grid-Ansicht aus.

SetFocus

Siehe auch

Methode: `SetFocus (pFocusData` als [XMLData](#))

Beschreibung

Setzt den Fokus auf das Element `pFocusData` in der Grid-Ansicht.

3.2.23 SchemaDocumentationDlg

Siehe auch

Eigenschaften und Methoden

Standard-Automation-Eigenschaften

[Application](#)

[Parent](#)

Eigenschaften bzgl. Interaktion und Sichtbarkeit

[OutputFile](#)

[OutputFileDialogAction](#)

[OptionsDialogAction](#)

[ShowProgressbar](#)

[ShowResult](#)

Optionen und Methoden für die Dokumentgenerierung

[OutputFormat](#)

[EmbedDiagrams](#)

[DiagramFormat](#)

[MultipleOutputFiles](#)

[EmbedCSSInHTML](#)

[CreateDiagramsFolder](#)

[GenerateRelativeLinks](#)

[IncludeAll](#)

[IncludeIndex](#)

[IncludeGlobalAttributes](#)

[IncludeGlobalElements](#)

[IncludeLocalAttributes](#)

[IncludeLocalElements](#)

[IncludeGroups](#)

[IncludeComplexTypes](#)

[IncludeSimpleTypes](#)

[IncludeAttributeGroups](#)

[IncludeRedefines](#)

[IncludeReferencedSchemas](#)

[AllDetails](#)

[ShowDiagram](#)

[ShowNamespace](#)

[ShowType](#)

[ShowChildren](#)

[ShowUsedBy](#)

[ShowProperties](#)

[ShowSingleFacets](#)

[ShowPatterns](#)

[ShowEnumerations](#)

[ShowAttributes](#)

[ShowIdentityConstraints](#)

[ShowAnnotations](#)

[ShowSourceCode](#)

Beschreibung

Dieses Objekt kombiniert alle Optionen für die Schemadokumentgenerierung, die auch über die Dialogfelder der Benutzeroberfläche von XMLSpy verfügbar 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 Eigenschaft [SetOutputFile](#) auf einen gültigen Dateipfad setzen. Verwenden Sie [OptionsDialogAction](#), [OutputFileDialogAction](#) und [ShowProgressBar](#) um den Grad der gewünschten Benutzerinteraktion festzulegen. Mit Hilfe von [trueAl](#) und [ADet](#) können Sie ganze Optionsgruppen auf einmal setzen. Durch Definition der einzelnen Eigenschaften können Sie die Optionen einzeln definieren.

AllDetails

Siehe auch

Methode: [AllDetails](#) ([i_bDetailsOn](#) als `Boolean`)

Beschreibung

Mit Hilfe dieser Methode können Sie alle Detailoptionen aktivieren bzw. deaktivieren.

Fehler

2900 Das Objekt ist nicht mehr gültig.

Application

Siehe auch

Eigenschaft: [Application](#) als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

2900 Das Objekt ist nicht mehr gültig.

2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

CreateDiagramsFolder

Siehe auch

Eigenschaft: [CreateDiagramsFolder](#) als `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 [DocumentGenerateWSDLDocumentation](#) 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.

DiagramFormat

Siehe auch

Eigenschaft: `DiagramFormat` als [SPYImageKind](#)

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 [DocumentGenerateSchemaDocumentation](#) 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.

EmbedCSSInHTML

Siehe auch

Eigenschaft: `EmbedCSSInHTML` als `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 [DocumentGenerateWSDLDocumentation](#) 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.

EmbedDiagrams

Siehe auch

Eigenschaft: `EmbedDiagrams` als `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 [DocumentGenerateSchemaDocumentation](#) 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.

GenerateRelativeLinks

Siehe auch

Eigenschaft: [GenerateRelativeLinks](#) als `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 [DocumentGenerateSchemaDocumentation](#) 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.

IncludeAll

Siehe auch

Methode: [IncludeAll](#) ([i_bInclude](#) als `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.

IncludeAttributeGroups

Siehe auch

Eigenschaft: [IncludeAttributeGroups](#) als `Boolean`

Beschreibung

Setzen Sie diese Eigenschaft auf `true`, um Attributgruppen in die Schemadokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von [DocumentGenerateSchemaDocumentation](#) 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.

IncludeComplexTypes

Siehe auch

Eigenschaft: [IncludeComplexTypes](#) als `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 [DocumentGenerateSchemaDocumentation](#) 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.

IncludeGlobalAttributes**Siehe auch**

Eigenschaft: [IncludeGlobalAttributes](#) als `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 [DocumentGenerateSchemaDocumentation](#) 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.

IncludeGlobalElements**Siehe auch**

Eigenschaft: [IncludeGlobalElements](#) als `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 [DocumentGenerateSchemaDocumentation](#) 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.

IncludeGroups**Siehe auch**

Eigenschaft: [IncludeGroups](#) als `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 [DocumentGenerateSchemaDocumentation](#) 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.

IncludeIndex

Siehe auch

Eigenschaft: `IncludeIndex` als `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 [DocumentGenerateSchemaDocumentation](#) 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.

IncludeLocalAttributes

Siehe auch

Eigenschaft: `IncludeLocalAttributes` als `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 [DocumentGenerateSchemaDocumentation](#) 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.

IncludeLocalElements

Siehe auch

Eigenschaft: `IncludeLocalElements` als `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 [DocumentGenerateSchemaDocumentation](#) 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.

IncludeRedefines

Siehe auch

Eigenschaft: `IncludeRedefines` als `Boolean`

Beschreibung

Setzen Sie diese Eigenschaft auf `true`, um Redefines in die Schemadokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von [DocumentGenerateSchemaDocumentation](#) 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.

IncludeReferencedSchemas**Siehe auch**

Eigenschaft: [IncludeReferencedSchemas](#) als `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 [DocumentGenerateSchemaDocumentation](#) 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.

IncludeSimpleTypes**Siehe auch**

Eigenschaft: [IncludeSimpleTypes](#) als `Boolean`

Beschreibung

Setzen Sie diese Eigenschaft auf `true`, um `simpleTypes` in die Schemadokumentation zu inkludieren. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von [DocumentGenerateSchemaDocumentation](#) 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.

MultipleOutputFiles**Siehe auch**

Eigenschaft: [MultipleOutputFiles](#) als `Boolean`

Beschreibung

Setzen Sie diese Eigenschaft auf `true`, um die Dokumentationsdateien aufzuteilen. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von [DocumentGenerateSchemaDocumentation](#) 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.

OptionsDialogAction**Siehe auch**

Eigenschaft: `OptionsDialogAction` als [`SPYDialogAction`](#)

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.

OutputFile**Siehe auch**

Eigenschaft: `SetOutputFile` als `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 dieses Objekt in einem Aufruf von [`DocumentGenerateSchemaDocumentation`](#) verwenden.

Fehler

- 2900 Das Objekt ist nicht mehr gültig.
- 2901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

OutputFileDialogAction**Siehe auch**

Eigenschaft: `OutputFileDialogAction` als [`SPYDialogAction`](#)

Beschreibung

Damit der Benutzer die Ausgabedatei in einem Dialogfeld zur Dateiauswahl auswählen kann, setzen Sie diese Eigenschaft auf `spyDialogUserInput(2)`. Wenn der in [`OutputFile`](#) gespeicherte Wert verwendet werden soll und keine 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.

OutputFormat

Siehe auch

Eigenschaft: `OutputFormat` als [SPYSchemaDocumentationFormat](#)

Beschreibung

Definiert, welche Art von Dokumentation generiert werden soll: HTML (Wert=0), MS-Word (Wert=1) oder RTF (Wert=2). Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von [DocumentGenerateSchemaDocumentation](#) 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.

Parent

Siehe auch

Eigenschaft: `Parent` als [Dialogs](#) (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.

ShowAnnotations

Siehe auch

Eigenschaft: `ShowAnnotations` als `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 [DocumentGenerateSchemaDocumentation](#) 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.

ShowAttributes

Siehe auch

Eigenschaft: `ShowAttributes` als `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 [DocumentGenerateSchemaDocumentation](#) 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.

ShowChildren**Siehe auch**

Eigenschaft: [ShowChildren](#) als `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 [DocumentGenerateSchemaDocumentation](#) 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.

ShowDiagram**Siehe auch**

Eigenschaft: [ShowDiagram](#) als `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 [DocumentGenerateSchemaDocumentation](#) 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.

ShowEnumerations**Siehe auch**

Eigenschaft: [ShowEnumerations](#) als `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 [DocumentGenerateSchemaDocumentation](#) 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.

ShowIdentityConstraints**Siehe auch**

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 [DocumentGenerateSchemaDocumentation](#) 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.

ShowNamespace**Siehe auch**

Eigenschaft: `ShowNamespace` als `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 [DocumentGenerateSchemaDocumentation](#) 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.

ShowPatterns**Siehe auch**

Eigenschaft: `ShowPatterns` als `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 [DocumentGenerateSchemaDocumentation](#) 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.

ShowProgressBar

Siehe auch

Eigenschaft: `ShowProgressBar` als `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.

ShowProperties

Siehe auch

Eigenschaft: `ShowProperties` als `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 [DocumentGenerateSchemaDocumentation](#) 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.

ShowResult

Siehe auch

Eigenschaft: `ShowResult` als `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 XMLSpy geöffnet. Um Word-Dokumentation anzuzeigen, wird MS-Word gestartet. Die Eigenschaft wird mit dem Wert initialisiert, der beim letzten Aufruf von [DocumentGenerateSchemaDocumentation](#) 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.

ShowSingleFacets

Siehe auch

Eigenschaft: `ShowSingleFacets` als `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 [DocumentGenerateSchemaDocumentation](#) 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.

ShowSourceCode**Siehe auch**

Eigenschaft: [ShowSourceCode](#) als `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 [DocumentGenerateSchemaDocumentation](#) 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.

ShowType**Siehe auch**

Eigenschaft: [ShowType](#) als `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 [DocumentGenerateSchemaDocumentation](#) 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.

ShowUsedBy**Siehe auch**

Eigenschaft: [ShowUsedBy](#) als `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 [DocumentGenerateSchemaDocumentation](#) 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.

SPSFile**Siehe auch**

Eigenschaft: [SPSFile](#) als 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.

UseFixedDesign**Siehe auch**

Eigenschaft: [UseFixedDesign](#) als Boolean

Beschreibung

Gibt an, ob 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.

3.2.24 SpyProject

Siehe auch**Methoden**

[CloseProject](#)
[SaveProject](#)
[SaveProjectAs](#)

Eigenschaften

[RootItems](#)
[ProjectFile](#)

Beschreibung

[SpyProject](#) Class

CloseProject**Siehe auch**

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

`CloseProject` schließt das aktuelle Projekt.

ProjectFile

Siehe auch

Deklaration: `ProjectFile` als String

Beschreibung

Pfad und Dateiname des Projekts.

RootItems

Siehe auch

Deklaration: `RootItems` als [SpyProjectItems](#)

Beschreibung

Root-Ebene der Sammlung der Projektelemente.

SaveProject

Siehe auch

Deklaration: `SaveProject`

Beschreibung

`SaveProject` speichert das aktuelle Projekt.

SaveProjectAs

Siehe auch

Deklaration: `SaveProjectAs` (`strPath` als String, `bDialog` als Boolean)

Parameter

`strPath`

Der vollständige Pfad mit dem Dateinamen der neuen Projektdatei.

`bDialog`

Wenn `bDialog` `TRUE` ist, wird ein Datei-Dialogfeld angezeigt.

Beschreibung

`SaveProjectAs` speichert die Projektdaten unter einem neuen Pfad.

3.2.25 SpyProjectItem

Siehe auch

Methoden

[Open](#)

Eigenschaften

[ChildItems](#)

[ParentItem](#)

[FileExtensions](#)

[ItemType](#)

[Name](#)

[Path](#)

[ValidateWith](#)

[XMLForXSLTransformation](#)

[XSLForXMLTransformation](#)

[XSLTransformationFileExtension](#)

[XSLTransformationFolder](#)

Beschreibung

`SpyProjectItem` Class

ChildItems

Siehe auch

Deklaration: `ChildItems` als [SpyProjectItems](#)

Beschreibung

Wenn es sich bei dem Element um einen Ordner handelt, ist `ChildItems` die Sammlung des Ordnerinhalts.

FileExtensions

Siehe auch

Deklaration: `FileExtensions` als String

Beschreibung

Dient zum Definieren der Dateierweiterungen, wenn das Projektelement ein Ordner ist.

ItemType

Siehe auch

Deklaration: `ItemType` als [SPYProjectItemTypes](#)

Beschreibung

Die Eigenschaft ist schreibgeschützt.

Name

Siehe auch

Deklaration: `Name` als String

Beschreibung

Name des Projektelements. Die Eigenschaft ist schreibgeschützt.

Open

Siehe auch

Deklaration: `Open` als [Document](#)

Rückgabewert

Das Projektelement wird als Dokument geöffnet.

Beschreibung

Öffnet das Projektelement.

ParentItem

Siehe auch

Deklaration: `ParentItem` als [SpyProjectItem](#)

Beschreibung

Das dem aktuellen Projektelement übergeordnete Element. Kann NULL (nichts) sein, wenn das Projektelement ein Element der obersten Ebene ist.

Path

Siehe auch

Deklaration: `Path` als String

Beschreibung

Pfad des Projektelements. Diese Eigenschaft ist schreibgeschützt.

ValidateWith

Siehe auch

Deklaration: `ValidateWith` als String

Beschreibung

Dient zum Definieren des Schemas/der DTD für die Validierung.

XMLForXSLTransformation

Siehe auch

Deklaration: `XMLForXSLTransformation` als String

Beschreibung

Dient zum Definieren der XML-Datei für die XSL-Transformation.

XSLForXMLTransformation

Siehe auch

Deklaration: [XSLForXMLTransformation](#) als String

Beschreibung

Dient zum Definieren der XSL-Datei für die XML-Transformation.

XSLTransformationFileExtension

Siehe auch

Deklaration: [XSLTransformationFileExtension](#) als String

Beschreibung

Dient zum Definieren der Dateierweiterung für die Ausgabedateien der XSL-Transformation.

XSLTransformationFolder

Siehe auch

Deklaration: [XSLTransformationFolder](#) als String

Beschreibung

Dient zum Definieren des Zielordners für die Ausgabedateien der XSL-Transformation.

3.2.26 SpyProjectItems

Siehe auch

Methoden

[AddFile](#)

[AddFolder](#)

[AddURL](#)

[RemoveItem](#)

Eigenschaften

[Count](#)

[Item](#)

Beschreibung

[SpyProjectItems](#) Class

AddFile

Siehe auch

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.

AddFolder

Siehe auch

Deklaration: `AddFolder` (`strName` als String)

Parameter

`strName`

Name des neuen Ordners

Beschreibung

Die Methode `AddFolder` fügt einen Ordner mit dem Namen `strName` zur Sammlung der Projektelemente hinzu.

AddURL

Siehe auch

Deklaration: `AddURL` (`strURL` als String, `nURLType` als `SPYURLTypes`, `strUser` als String, `strPassword` als String, `bSave` als Boolean)

Beschreibung

`strURL`

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.

Count

Siehe auch

Deklaration: `Count` als long

Beschreibung

Diese Eigenschaft ruft die Anzahl der Projektelemente in der Sammlung ab. Die Eigenschaft ist schreibgeschützt.

Item

Siehe auch

Deklaration: `Item` (*n* als long) als [SpyProjectItem](#)

Beschreibung

Ruft das nte Element der Sammlung der Projektelemente auf. Das erste Element hat den Index 1.

RemoveItem

Siehe auch

Deklaration: `RemoveItem` (*pItem* als [SpyProjectItem](#))

Beschreibung

`RemoveItem` löscht das Element *pItem* aus der Sammlung der Projektelemente.

3.2.27 TextImportExportSettings

Siehe auch

Eigenschaften nur für den Import

[ImportFile](#)

Eigenschaften nur für den Export

[DestinationFolder](#)

[FileExtension](#)

[CommentIncluded](#)

[RemoveDelimiter](#)

[RemoveNewline](#)

Eigenschaften für Import und Export

[HeaderRow](#)

[FieldDelimiter](#)

[EnclosingCharacter](#)

[Encoding](#)

[EncodingByteOrder](#)

Beschreibung

`TextImportExportSettings` enthält Optionen, die Textimport- und -exportfunktionen gemeinsam haben.

CommentIncluded

Siehe auch

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.

DestinationFolder

Siehe auch

Eigenschaft: [DestinationFolder](#) als String

Beschreibung

Die Eigenschaft `DestinationFolder` legt den Ordner fest, in dem die gespeicherten Dateien beim Textexport abgelegt werden.

EnclosingCharacter

Siehe auch

Eigenschaft: [EnclosingCharacter](#) als [SPYTextEnclosing](#)

Beschreibung

Diese Eigenschaft definiert das Zeichen, das alle Feldwerte für den Import und Export umschließt. Der Standardwert ist [spyNoEnclosing](#).

Encoding

Siehe auch

Eigenschaft: [Encoding](#) als String

Beschreibung

Die Eigenschaft `Encoding` definiert die Zeichensatzcodierung für die zu importierenden und exportierenden Textdateien.

EncodingByteOrder

Siehe auch

Eigenschaft: [EncodingByteOrder](#) als [SPYEncodingByteOrder](#)

Beschreibung

Die Eigenschaft `EncodingByteOrder` definiert die Bytefolge für Unicode-Zeichen. Standardeinstellung ist [spyNONE](#).

FieldDelimiter

Siehe auch

Eigenschaft: [FieldDelimiter](#) als [SPYTextDelimiters](#)

Beschreibung

Die Eigenschaft `FieldDelimiter` definiert die Trennzeichen zwischen den Feldern beim Import und Export. Standardeinstellung ist [spyTabulator](#).

FileExtension

Siehe auch

Eigenschaft: [FileExtension](#) als String

Beschreibung

Diese Eigenschaft definiert die Dateierweiterung für beim Textexport erstellte Dateien.

HeaderRow

Siehe auch

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.

ImportFile

Siehe auch

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](#).

RemoveDelimiter

Siehe auch

Eigenschaft: [RemoveDelimiter](#) als Boolean

Beschreibung

Die Eigenschaft [RemoveDelimiter](#) definiert, ob Zeichen im Text, die dem Trennzeichen entsprechen, entfernt werden sollen. Der Standardwert ist false. Diese Eigenschaft wird beim Export in Textdateien verwendet.

RemoveNewline

Siehe auch

Eigenschaft: [RemoveNewline](#) als Boolean

Beschreibung

Die Eigenschaft [RemoveNewline](#) definiert, 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.

3.2.28 TextView

Siehe auch

Eigenschaften und Methoden

[Application](#)

[Parent](#)

[LineFromPosition](#)

[PositionFromLine](#)

[LineLength](#)

[SetText](#)

[GetRangeText](#)

[ReplaceText](#)

[MoveCaret](#)

[GoToLineChar](#)

[SelectText](#)

[SelectionStart](#)

[SelectionEnd](#)

[Text](#)

[LineCount](#)

[Length](#)

Beschreibung

Events

OnBeforeShowSuggestions

Siehe auch

Event: [OnBeforeShowSuggestions\(\)](#) als Boolean

Beschreibung

Dieses Event wird ausgelöst, bevor das Vorschlagsfenster angezeigt wird. Die [Document](#)-Eigenschaft [Suggestions](#) 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](#) ein leerer Array zugewiesen werden. Die beste Stelle dafür ist das [OnDocumentOpened](#) Event. Damit das Vorschlagsfenster nicht angezeigt wird, geben Sie [false](#) zurück. Um es weiter anzuzeigen, verwenden Sie den Wert [true](#).

Beispiele

Unten finden Sie Beispiele dafür, wie dieses Event in Form eines Scripts geschrieben werden kann.

XMLSpy Skripting-Umgebung - VBScript:

```
Function On\_BeforeShowSuggestions\(\)  
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On\_BeforeShowSuggestions\(\)  
{  
}  
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn. OnEvent (33, ...) //nEventId=33
```

OnChar**Siehe auch**

Event: **OnChar**(**nChar** als Long, **bExistSuggestions** als Boolean) als Boolean

Beschreibung

Dieses Event wird bei jedem Tastenanschlag ausgelöst. Der Parameter **nChar** ist die Taste, die gedrückt wurde und **bExistSuggestions** gibt darüber Auskunft ob ein von XMLSpy generiertes Vorschlagsfenster nach dem Drücken dieser Taste angezeigt wird. Die [Document](#) Eigenschaft [Suggestions](#) 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](#) ein leerer Array zugewiesen werden. Die beste Stelle dafür ist das [OnDocumentOpened](#) 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](#) Eigenschaft [Suggestions](#) auf einen String Array mit Ihren Empfehlungen und geben Sie **true** zurück.

Dieses Event wird vor dem [OnBeforeShowSuggestions](#) Event ausgelöst. Wenn Sie festlegen, dass das Vorschlagsfenster nicht angezeigt werden soll, indem Sie den Wert auf **false** setzen, dann wird [OnBeforeShowSuggestions](#) nicht ausgelöst.

Beispiele

Unterhalb finden Sie Beispiele, wie ein Skript für dieses Event erstellt werden kann.

XMLSpy Skripting-Umgebung - VBScript:

```
Function On_Char(nChar, bExistSuggestions )
EndFunction
```

XMLSpy Skripting-Umgebung - JScript:

```
function On_Char(nChar, bExistSuggestions )
{
}
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn. OnEvent (35, ...) //nEventId=35
```

Application

Eigenschaft: [Application](#) als [Application](#) (schreibgeschützt)

Beschreibung

Ruft das XMLSpy application-Objekt auf.

Fehler

- 3900 Das Objekt ist nicht mehr gültig.
- 3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

GetRangeText

Methode: `GetRangeText`(`nStart` als Long, `nEnd` als 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.

GoToLineChar

Methode: `GoToLineChar`(`nLine` als Long, `nChar` als Long)

Beschreibung

Verschiebt den Cursor an die angegebene Zeilen- und Zeichenposition.

Fehler

- 3900 Das Objekt ist nicht mehr gültig.
- 3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Length

Eigenschaft: `Length` als 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.

LineCount

Eigenschaft: `LineCount` als 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.

LineFromPosition

Methode: `LineFromPosition`(`nCharPos` als 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.

LineLength

Methode: `LineLength`(`nLine` als 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.

MoveCaret

Methode: `MoveCaret`(`nDiff` als Long)

Beschreibung

Verschiebt den Cursor um `nDiff` Zeichen.

Fehler

- 3900 Das Objekt ist nicht mehr gültig.
- 3901 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Parent

Eigenschaft: `Parent` als [Document](#) (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.

PositionFromLine

Methode: `PositionFromLine`(`nLine` als 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.

ReplaceText

Methode: `ReplaceText`(`nPosFrom` als Long, `nPosTill` als Long, `sText` als 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.

SelectionEnd

Eigenschaft: [SelectionEnd](#) als 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.

SelectionStart

Eigenschaft: [SelectionStart](#) als 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.

SelectText

Methode: [SelectText](#)([nPosFrom](#) als Long, [nPosTill](#) als 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.

SelText

Eigenschaft: [SelText](#) als 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.

Text

Eigenschaft: [Text](#) als 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.

3.2.29 WSDLDocumentationDlg

Siehe auch

Eigenschaften und Methoden

Standard-Automation-Eigenschaften

Application
Parent

Eigenschaften zur Interaktion und Sichtbarkeit

OptionsDialogAction
OutputFile
OutputFileDialogAction
ShowProgressBar
ShowResult

Optionen und Methoden zur Dokumentgenerierung

OutputFormat
EmbedDiagrams
DiagramFormat
MultipleOutputFiles
EmbedCSSInHTML
CreateDiagramsFolder

IncludeAll
IncludeBinding
IncludeImportedWSDLFiles
IncludeMessages
IncludeOverview
IncludePortType
IncludeService
IncludeTypes

AllDetails
ShowBindingDiagram
ShowExtensibility
ShowMessageParts
ShowPort
ShowPortTypeDiagram
ShowPortTypeOperations
ShowServiceDiagram
ShowSourceCode
ShowTypesDiagram
ShowUsedBy

Beschreibung

Dieses Objekt kombiniert alle Optionen für die WSDL-Dokumentgenerierung, die über Dialogfelder der Benutzeroberfläche von XMLSpy zur 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 die `OutputFile` Eigenschaft auf einen gültigen Dateipfad setzen. Mit Hilfe von `OptionsDialogAction`, `OutputFileDialogAction` und `ShowProgressBar` können Sie den Grad der gewünschten Benutzerinteraktion definieren. Mit Hilfe von `IncludeAll` und `AllDetails` können Sie ganze Optionsgruppen auf einmal definieren. Oder Sie definieren die Eigenschaften einzeln, um die Feineinstellung vorzunehmen.

UseFixedDesign

Siehe auch

Eigenschaft: `UseFixedDesign` als Boolean

Beschreibung

Gibt an, ob 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.

SPSFile

Siehe auch

Eigenschaft: `SPSFile` als 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.

3.2.30 XBRLDocumentationDlg

Siehe auch

Eigenschaften und Methoden

Standard-Automation-Eigenschaften

`Application`
`Parent`

Eigenschaften zur Interaktion und Sichtbarkeit

`OptionsDialogAction`
`OutputFile`
`OutputFileDialogAction`
`ShowProgressBar`
`ShowResult`

Optionen und Methoden zur Dokumentgenerierung

`OutputFormat`
`EmbedDiagrams`
`DiagramFormat`
`EmbedCSSInHTML`
`CreateDiagramsFolder`

```
IncludeAll  
IncludeOverview  
IncludeNamespacePrefixes  
IncludeGlobalElements  
IncludeDefinitionLinkroles  
IncludePresentationLinkroles  
IncludeCalculationLinkroles
```

```
AllDetails  
ShowDiagram  
ShowSubstitutiongroup  
ShowItemtype  
ShowBalance  
ShowPeriod  
ShowAbstract  
ShowNillable  
ShowLabels  
ShowReferences  
ShowLinkbaseReferences
```

```
ShortQualifiedName  
ShowImportedElements
```

Beschreibung

Dieses Objekt kombiniert alle Optionen für die XBRL-Dokumentgenerierung, die über Dialogfelder der Benutzeroberfläche von XMLSpy zur 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 die `OutputFile` Eigenschaft auf einen gültigen Dateipfad setzen. Mit Hilfe von `OptionsDialogAction`, `OutputFileDialogAction` und `ShowProgressBar` können Sie den Grad der gewünschten Benutzerinteraktion definieren. Mit Hilfe von `IncludeAll` und `AllDetails` können Sie ganze Optionsgruppen auf einmal definieren. Oder Sie definieren die Eigenschaften einzeln, um die Feineinstellung vorzunehmen.

SPSFile

Siehe auch

Eigenschaft: `SPSFile` als 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.

UseFixedDesign

Siehe auch

Eigenschaft: [UseFixedDesign](#) als Boolean

Beschreibung

Gibt an, ob 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.

3.2.31 XMLData

Siehe auch

Eigenschaften

[Kind](#)

[Name](#)

[TextValue](#)

[HasChildren](#)

[MayHaveChildren](#)

[Parent](#)

Methoden

[GetFirstChild](#)

[GetNextChild](#)

[GetCurrentChild](#)

[InsertChild](#)

[InsertChildAfter](#)

[InsertChildBefore](#)

[AppendChild](#)

[EraseAllChildren](#)

[EraseChild](#)

[EraseCurrentChild](#)

[IsSameNode](#)

[CountChildren](#)

[CountChildrenKind](#)

[GetChild](#)

[GetChildAttribute](#)

[GetChildElement](#)

[GetChildKind](#)

[GetNamespacePrefixForURI](#)

[HasChildrenKind](#)

[SetTextValueXMLEncoded](#)

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` Eigenschaft auf 'Text' gesetzt ist.
- 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`. Jeder Teil kennt seine untergeordneten Elemente, wodurch eine XMLData-Struktur mit `DocumentRootElement` auf der obersten Ebene gebildet wird. Um das oberste Element des Dokumentinhalts (ohne XML Header) abzurufen, verwenden Sie `DocumentDataRoot`. Beispiele, wie man die XMLData-Struktur durchquert, finden Sie unter `Using XMLData to modify document structure` und `GetNextChild`.

AppendChild

Siehe auch

Deklaration: `AppendChild` (*pNewData* als `XMLData`)

Beschreibung

`AppendChild` hängt *pNewData* als letztes Child-Element an das `XMLData`-Objekt an. Siehe auch `Using XMLData`.

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 ungültige XMLData-Art für diese Position definiert.
- 1900 Das Dokument darf nicht geändert werden.

Beispiel

```
Dim objCurrentParent As XMLData
Dim objNewChild As XMLData
```

```
Set objNewChild = objSpy.ActiveDocument.CreateChild(spyXMLDataElement)
Set objCurrentParent = objSpy.ActiveDocument.RootElement

objCurrentParent.AppendChild objNewChild

Set objNewChild = Nothing
```

CountChildren

Siehe auch

Deklaration: `CountChildren` als long

Beschreibung

`CountChildren` ruft die Anzahl der Child-Elemente ab.

Verfügbar mit Typbibliothek Version 1.5

Fehler

1500 Das XMLData-Objekt ist nicht mehr gültig.

CountChildrenKind

Siehe auch

Deklaration: `CountChildrenKind` (`nKind` als `SPYXMLDataKind`) als long

Beschreibung

`CountChildrenKind` ruft die Anzahl der Child-Elemente dieser bestimmten Art ab.

Verfügbar mit Typbibliothek Version 1.5

Fehler

1500 Das XMLData-Objekt ist nicht mehr gültig.

EraseAllChildren

Siehe auch

Deklaration: `EraseAllChildren`

Beschreibung

`EraseAllChildren` löscht alle zum XMLData Objekt 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.

```
Dim objCurrentParent As XMLData
```

```
Set objCurrentParent = objSpy.ActiveDocument.RootElement
objCurrentParent.EraseAllChildren
```

EraseChild

Methode: `EraseChild` (Child als [XMLData](#))

Beschreibung

Löscht den angegebenen Child Node.

Fehler

- 1500 Ungültiges Objekt.
- 1506 Ungültige XML-Input-Datei.
- 1510 Ungültiger Parameter.

EraseCurrentChild

Siehe auch

Deklaration: `EraseCurrentChild`

Beschreibung

`EraseCurrentChild` löscht das aktuelle `XMLData` Child-Objekt. Bevor Sie `EraseCurrentChild` aufrufen, müssen Sie einen internen Iterator mit [XMLData.GetFirstChild](#) 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 `EraseCurrentChild` gibt 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

```
// -----
// XMLSpy 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); }
}
```

GetChild

Siehe auch

Deklaration: `GetChild` (*position* als long) als [XMLData](#)

Rückgabewert

Gibt ein XML-Element als `XMLData` Objekt 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.

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.

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.

GetChildKind

Siehe auch

Deklaration: `GetChildKind` (*position* als long, *nKind* als [SPYXMLDataKind](#)) als

[XMLData](#)

Rückgabewert

Gibt ein XML-Element als `XMLData` Objekt zurück.

Beschreibung

`GetChild(nKind)` 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.

GetCurrentChild

Siehe auch

Deklaration: `GetCurrentChild` als [XMLData](#)

Rückgabewert

Gibt ein XML-Element als `XMLData` Objekt zurück.

Beschreibung

`GetCurrentChild` ruft das aktuelle Child-Element ab. Bevor Sie `GetCurrentChild` aufrufen, müssen Sie einen internen Iterator mit [XMLData.GetFirstChild](#) 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.

GetFirstChild

Siehe auch

Deklaration: `GetFirstChild` (`nKind` als [SPYXMLDataKind](#)) als [XMLData](#)

Rückgabewert

Gibt ein XML-Element als `XMLData` Objekt zurück.

Beschreibung

`GetFirstChild` initialisiert 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](#), [GetNextChild](#) oder [EraseCurrentChild](#) 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](#).

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.

GetNextChild

Siehe auch

Deklaration: [GetNextChild](#) als [XMLData](#)

Rückgabewert

Gibt ein XML-Element als `XMLData` Objekt zurück.

Beschreibung

[GetNextChild](#) geht zum nächsten Child-Element dieses Elements. Bevor Sie [GetNextChild](#) aufrufen, müssen Sie einen internen Iterator mit [XMLData.GetFirstChild](#) 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
' -----
On Error Resume Next
Set objParent = objSpy.ActiveDocument.RootElement

'get elements of all kinds
Set objCurrentChild = objParent.GetFirstChild(-1)

Do
    'do something useful with the child

    'step to next child
    Set objCurrentChild = objParent.GetNextChild
Loop Until (Err.Number - vbObjectError = 1503)

// -----
// XMLSpy 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
    else if ((err.number & 0xffff) == 1503)
        ; // last child reached
    else
        throw (err);
}
```

GetTextValueXMLDecoded

Methode: `GetTextValueXMLDecoded()` als string

Beschreibung

Ruft den dekodierten Textwert der XML-Datei ab.

Fehler

- 1500 Ungültiges Objekt.
- 1510 Ungültiger Parameter.

HasChildren

Siehe auch

Deklaration: `HasChildren` als Boolean

Beschreibung

Die Eigenschaft ist true, wenn das Objekt das übergeordnete Objekt von anderen `XMLData` Objekten ist. Diese Eigenschaft ist schreibgeschützt.

Fehler

- 1500 Das XMLData-Objekt ist nicht mehr gültig.
- 1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

HasChildrenKind

Siehe auch

Deklaration: `HasChildrenKind` (*nKind* als `SPYXMLDataKind`) als Boolean

Beschreibung

Die Methode gibt true zurück, wenn das Objekt das übergeordnete Objekt von anderen

XMLData Objekten 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.

InsertChild

Siehe auch

Deklaration: `InsertChild` (*pNewData* als [XMLData](#))

Beschreibung

`InsertChild` fügt das neue Child-Element vor dem aktuellen Child-Element ein (Informationen zum Setzen des aktuellen Child-Elements finden Sie auch unter [XMLData.GetFirstChild](#), [XMLData.GetNextChild](#)).

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.

Beispiele

Siehe [Using XMLData to modify document structure](#).

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

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

IsSameNode**Siehe auch**

Deklaration: `IsSameNode` (`pNodeToCompare` als [XMLData](#)) als Boolean

Beschreibung

Gibt true zurück, wenn `pNodeToCompare` denselben Node wie das Objekt selbst referenziert.

Fehler

- 1500 Das XMLData-Objekt ist nicht mehr gültig.
- 1506 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Kind**Siehe auch**

Dellaration: `Kind` als [SPYXMLDataKind](#)

Beschreibung

Art dieses `XMLData` Objekts. Diese Eigenschaft ist schreibgeschützt.

Fehler

- 1500 Das XMLData-Objekt ist nicht mehr gültig.
- 1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

MayHaveChildren**Siehe auch**

Deklaration: `MayHaveChildren` als Boolean

Beschreibung

Gibt an, ob Child-Elemente zu diesem `XMLData` Objekt 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.

Name**Siehe auch**

Deklaration: `Name` als String

Beschreibung

Dient zum Bearbeiten und Abrufen des Namens des `XMLData` Objekts.

Fehler

- 1500 Das XMLData-Objekt ist nicht mehr gültig.
- 1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Parent**Siehe auch**

Deklaration: `Parent` als `XMLData`

Rückgabewert

Übergeordnetes Element als `XMLData` Objekt. 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.

SetTextValueXMLEncoded

Methode: `SetTextValueXMLEncoded` (`strVal` als `String`)

Beschreibung

Definiert den kodierten Textwert der XML-Datei.

Fehler

- 1500 Ungültiges Objekt.
- 1513 Keine Änderung zulässig.

TextValue**Siehe auch**

Deklaration: `TextValue` als String

Beschreibung

Dient zum Bearbeitung und Abrufen des Textwerts dieses `XMLData` Objekts.

Fehler

- 1500 Das XMLData-Objekt ist nicht mehr gültig.
- 1510 Für den Rückgabeparameter wurde eine ungültige Adresse angegeben.

Beispiele

Siehe „ndem derD okum entstruktur iH ie von XM LData .

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.

3.3.1 AuthenticEvent (obsolete)

Wird ersetzt durch [AuthenticView](#) und [AuthenticRange](#)

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](#), [DocEditSelection](#), [DocEditEvent](#) und [DocEditDataTransfer](#) verfügbaren Funktionalitäten stehen nun über [AuthenticView](#), [AuthenticRange](#) und [AuthenticDataTransfer](#) 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.

Siehe auch

Eigenschaften

[altKey](#)
[altLeft](#)
[ctrlKey](#)
[ctrlLeft](#)
[shiftKey](#)
[shiftLeft](#)

[keyCode](#)
[repeat](#)

[button](#)

[clientX](#)
[clientY](#)

[dataTransfer](#)

[srcElement](#)
[fromElement](#)

[propertyName](#)

[cancelBubble](#)
[returnValue](#)

[type](#)

Beschreibung

DocEditEvent Schnittstelle.

altKey (nicht mehr verwendet)**Wird ersetzt durch Parameter für**[AuthenticView.OnKeyboardEvent](#) (On_AuthenticView_KeyPressed)[AuthenticView.OnMouseEvent](#) (On_AuthenticView_MouseEvent)[AuthenticView.OnDragOver](#) (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 erleichtern. 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");  
}
```

Siehe auch**Deklaration:** [altKey](#) als Boolean**Beschreibung**

True, wenn die rechte ALT-Taste gedrückt wird.

altLeft (nicht mehr verwendet)**Wird ersetzt durch Parameter für**[AuthenticView.OnKeyboardEvent](#) (On_AuthenticView_KeyPressed)[AuthenticView.OnMouseEvent](#) (On_AuthenticView_MouseEvent)[AuthenticView.OnDragOver](#) (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 erleichtern. 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, long i_nKeyCode,
SPYVirtualKeyMask i_nVirtualKeyStatus)
{
    if ( i_nVirtualKeyStatus & spyAltKeyMask)
        MsgBox ("alt key is down");
}
```

Siehe auch

Deklaration: `altLeft` als Boolean

Beschreibung

True, wenn die linke ALT-Taste gedrückt wird.

button (nicht mehr verwendet)**Wird ersetzt durch Parameter für****[AuthenticView.OnMouseEvent](#)** (On_AuthenticView_MouseEvent)**[AuthenticView.OnDragOver](#)** (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 erleichtern. 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 (long i_nXPos, long i_nYPos,  
SPYMouseEvent i_eMouseEvent, IAuthenticRange *i_ipRange)  
{  
    if (i_eMouseEvent & spyLeftButtonDownMask)  
        MsgBox ("left mouse button down detected");  
}
```

Siehe auch**Deklaration:** `button` als long**Beschreibung**

Definiert, welche Maustaste gedrückt wird:

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.

cancelBubble (nicht mehr verwendet)

Wird ersetzt durch den Booleschen Rückgabewert der folgenden Event Handler-Funktionen

[AuthenticView.OnKeyboardEvent](#) (On_AuthenticView_KeyPressed)

[AuthenticView.OnMouseEvent](#) (On_AuthenticView_MouseEvent)

[AuthenticView.OnDragOver](#) (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 erleichtern. 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, long
i_nKeyCode, SPYVirtualKeyMask i_nVirtualKeyStatus)
{
    if ( i_nKeyCode == 0x20)
        return true; // cancel key processing, swallow spaces :- )
}
```

Siehe auch

Declaration: [cancelBubble](#) als Boolean

Beschreibung

Setzen Sie [cancelBubble](#) auf TRUE, wenn der Standard-Event Handler nicht aufgerufen werden soll.

clientX (nicht mehr verwendet)

Wird ersetzt durch Parameter für

[AuthenticView.OnMouseEvent](#) (On_AuthenticView_MouseEvent)

[AuthenticView.OnBeforeDrop](#) (On_AuthenticView_BeforeDrop)

[AuthenticView.OnDragOver](#) (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 erleichtern. 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 ( long i_nXPos, long i_nYPos,  
SPYMouseEvent i_eMouseEvent, IAuthenticRange *i_ipRange)  
{  
    if ( i_eMouseEvent & spyMouseMoveMask)  
        MsgBox ("moving over " + i_nXPos + "/" + i_nYPos);  
}
```

Siehe auch

Deklaration: `clientX` als long

Beschreibung

X-Wert der aktuellen Mausposition in den Client-Koordinaten.

clientY (nicht mehr verwendet)

Wird ersetzt durch Parameter für

[AuthenticView.OnMouseEvent](#) (On_AuthenticView_MouseEvent)

[AuthenticView.OnBeforeDrop](#) (On_AuthenticView_BeforeDrop)

[AuthenticView.OnDragOver](#) (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 erleichtern. 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 ( long i_nXPos, long i_nYPos,
SPYMouseEvent i_eMouseEvent, IAuthenticRange *i_ipRange)
{
    if ( i_eMouseEvent & spyMouseMoveMask)
        MsgBox ("moving over " + i_nXPos + "/" + i_nYPos);
}
```

Siehe auch

Deklaration: `clientY` als long

Beschreibung

Y-Wert der aktuellen Mausposition in den Client-Koordinaten.

ctrlKey (nicht mehr verwendet)

Wird ersetzt durch Parameter für

[AuthenticView.OnKeyboardEvent](#) (On_AuthenticView_KeyPressed)

[AuthenticView.OnMouseEvent](#) (On_AuthenticView_MouseEvent)

[AuthenticView.OnDragOver](#) (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 erleichtern. 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");  
}
```

Siehe auch

Deklaration: `ctrlKey` als Boolean

Beschreibung

True, wenn die rechte Strg-Taste gedrückt wird.

ctrlLeft (nicht mehr verwendet)

Wird ersetzt durch parameters to

[AuthenticView.OnKeyboardEvent](#) (On_AuthenticView_KeyPressed)

[AuthenticView.OnMouseEvent](#) (On_AuthenticView_MouseEvent)

[AuthenticView.OnDragOver](#) (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 erleichtern. 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");  
}
```

Siehe auch

Deklaration: `ctrlLeft` als Boolean

Beschreibung

True, wenn die linke Strg-Taste gedrückt wird.

dataTransfer (nicht mehr verwendet)

Wird ersetzt durch Parameter für

[AuthenticView.OnBeforeDrop](#) (On_AuthenticView_BeforeDrop)

[AuthenticView.OnDragOver](#) (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 erleichtern. 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 ( long i_nXPos, long i_nYPos,
IAuthenticRange *i_ipRange,
IAuthenticDataTransfer *i_ipData)
{
    if (i_ipRange != null)
        if (! i_ipRange.ownDrag)
            return true; // cancel key processing, don't drop foreign
objects :-)
    return false;
}
```

Siehe auch

Deklaration: [dataTransfer](#) als Variant

Beschreibung

Eigenschaft [dataTransfer](#)

fromElement (nicht mehr verwendet)**Nicht unterstützt****Siehe auch****Deklaration:** `fromElement` als Variant (nicht unterstützt)**Beschreibung**

Derzeit wird diese Eigenschaft von keinem Event definiert.

keyCode (nicht mehr verwendet)**Wird ersetzt durch einen Parameter für
[AuthenticView.OnKeyboardEvent](#) (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 erleichtern. 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, long  
i_nKeyCode, SPYVirtualKeyMask i_nVirtualKeyStatus)  
{  
    if ( i_nKeyCode == 0x20)  
        return true; // cancel key processing, swallow spaces :-)  
}
```

Siehe auch**Deklaration:** `keyCode` als long**Beschreibung**

Tastaturcode der aktuell gedrückten Taste. Diese Eigenschaft ist read-write.

propertyName (nicht mehr verwendet)

Nicht unterstützt

Siehe auch

Deklaration: `propertyName` als String (nicht unterstützt)

Beschreibung

Derzeit wird diese Eigenschaft von keinem Event definiert.

repeat (nicht mehr verwendet)

Nicht unterstützt

Siehe auch

Deklaration: `repeat` als Boolean (nicht unterstützt)

Beschreibung

True, wenn das `onkeydown` Event wiederholt wird.

returnValue (nicht mehr verwendet)

Nicht mehr unterstützt

Siehe auch

Deklaration: `returnValue` als Variant

Beschreibung

Mit Hilfe von `returnValue` können Sie einen Rückgabewert für Ihren Event Handler setzen.

shiftKey (nicht mehr verwendet)

Wird ersetzt durch Parameter für

[AuthenticView.OnKeyboardEvent](#) (On_AuthenticView_KeyPressed)

[AuthenticView.OnMouseEvent](#) (On_AuthenticView_MouseEvent)

[AuthenticView.OnDragOver](#) (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 erleichtern. 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");
}
```

Siehe auch

Deklaration: `shiftKey` als Boolean

Beschreibung

True, wenn die rechte Umschalttaste gedrückt wird.

shiftLeft (nicht mehr verwendet)

Wird ersetzt durch Parameter für

[AuthenticView.OnKeyboardEvent](#) (On_AuthenticView_KeyPressed)

[AuthenticView.OnMouseEvent](#) (On_AuthenticView_MouseEvent)

[AuthenticView.OnDragOver](#) (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 erleichtern. 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");
}
```

Siehe auch

Deklaration: `shiftLeft` als Boolean

Beschreibung

True, wenn die linke Umschalttaste gedrückt wird.

srcElement (nicht mehr verwendet)

Wird ersetzt durch Parameter für

[AuthenticView.OnMouseEvent](#) (On_AuthenticView_MouseEvent)

[AuthenticView.OnBeforeDrop](#) (On_AuthenticView_BeforeDrop)

[AuthenticView.OnDragOver](#) (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 erleichtern. 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 Mausursors 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 (long i_nXPos, long i_nYPos,
SPYMouseEvent i_eMouseEvent, IAuthenticRange *i_ipRange)
{
    if ((i_eMouseEvent & spyMouseMoveMask) &&
        (i_ipRange != null))
        MsgBox ("moving over " + i_ipRange.FirstXMLData.Parent.Name);
}
```

Siehe auch

Deklaration: `srcElement` als Variant

Beschreibung

Element, welches das aktuelle Event auslöst. Dies ist normalerweise ein [XMLData](#) Objekt.

type (nicht mehr verwendet)

Nicht unterstützt

Siehe auch

Deklaration: `type` als String (nicht unterstützt)

Beschreibung

Derzeit wird diese Eigenschaft von keinem Event definiert.

3.3.2 AuthenticSelection (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange](#)

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](#), [DocEditSelection](#), [DocEditEvent](#) und [DocEditDataTransfer](#) verfügbaren Funktionalitäten stehen nun über [AuthenticView](#), [AuthenticRange](#) und [AuthenticDataTransfer](#) 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.

Siehe auch

Eigenschaften

[Start](#)

[StartTextPosition](#)

[End](#)

[EndTextPosition](#)

End (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.LastXMLData](#)

```
// ----- javascript sample -----  
// instead of:  
// var objXMLData =  
Application.ActiveDocument.DocEditView.CurrentSelection.End;  
// use now:  
var objXMLData =  
Application.ActiveDocument.AuthenticView.Selection.LastXMLData;
```

Siehe auch

Deklaration: End als [XMLData](#)

Beschreibung

XML-Element, an dem die aktuelle Auswahl endet.

EndTextPosition (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.LastXMLDataOffset](#)

```
// ----- javascript sample -----  
// instead of:  
// var nOffset =  
Application.ActiveDocument.DocEditView.CurrentSelection.EndTextPosition;  
// use now:  
var nOffset =  
Application.ActiveDocument.AuthenticView.Selection.LastXMLDataOffset;
```

Siehe auch

Deklaration: EndTextPosition als long

Beschreibung

Positioniert den Cursor in [DocEditSelection.End.TextValue](#), wo die Auswahl endet.

Start (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.FirstXMLData](#)

```
// ----- javascript sample -----  
// instead of:  
// var objXMLData =  
Application.ActiveDocument.DocEditView.CurrentSelection.Start;  
// use now:  
var objXMLData =  
Application.ActiveDocument.AuthenticView.Selection.FirstXMLData;
```

Siehe auch

Deklaration: `Start` als [XMLData](#)

Beschreibung

XML-Element, an dem die Cursorauswahl beginnt.

StartTextPosition (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.FirstXMLDataOffset](#)

```
// ----- javascript sample -----  
// instead of:  
// var nOffset =  
Application.ActiveDocument.DocEditView.CurrentSelection.StartTextPosition;  
// use now:  
var nOffset =  
Application.ActiveDocument.AuthenticView.Selection.FirstXMLDataOffset;
```

Siehe auch

Deklaration: `StartTextPosition` als long

Beschreibung

Positioniert den Cursor in [DocEditSelection.Start.TextValue](#), wo die Auswahl beginnt.

3.3.3 OldAuthenticView (nicht mehr verwendet)

Wird ersetzt durch [AuthenticView](#) und [AuthenticRange](#)

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](#), [DocEditSelection](#), [DocEditEvent](#) und [DocEditDataTransfer](#) verfügbaren Funktionalitäten stehen nun über [AuthenticView](#), [AuthenticRange](#) und [AuthenticDataTransfer](#) 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.

Siehe auch

Methoden

[LoadXML](#)

[SaveXML](#)

[EditClear](#)

[EditCopy](#)

[EditCut](#)

[EditPaste](#)

[EditRedo](#)

[EditSelectAll](#)

[EditUndo](#)

[RowAppend](#)

[RowDelete](#)

[RowDuplicate](#)

[RowInsert](#)

[RowMoveDown](#)

[RowMoveUp](#)

[ApplyTextState](#)

[IsTextStateApplied](#)

[IsTextStateEnabled](#)

[MarkUpView](#)

[SelectionSet](#)

[SelectionMoveTabOrder](#)

[GetNextVisible](#)

[GetPreviousVisible](#)

[GetAllowedElements](#)

Eigenschaften

[CurrentSelection](#)

[event](#)

[XMLRoot](#)

[IsEditClearEnabled](#)

[IsEditCopyEnabled](#)

[IsEditCutEnabled](#)

[IsEditPasteEnabled](#)

[IsEditRedoEnabled](#)

[IsEditUndoEnabled](#)

[IsRowAppendEnabled](#)

[IsRowDeleteEnabled](#)

[IsRowDuplicateEnabled](#)

[IsRowInsertEnabled](#)

[IsRowMoveDownEnabled](#)

[IsRowMoveUpEnabled](#)

Beschreibung

Schnittstelle für die Authentic-Ansicht.

ApplyTextState (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.PerformAction](#)

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");
```

Siehe auch

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 wird das Wortfragment fett formatiert. Dies geschieht jedoch nur, weil der HTML-Parser weiß, dass der Tag 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](#) und [IsTextStateApplied](#).

CurrentSelection (nicht mehr verwendet)**Wird ersetzt durch [AuthenticView.Selection](#)**

Das zurückgegebene Objekt [AuthenticRange](#) 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;
```

Siehe auch

Deklaration: [CurrentSelection](#) als [DocEditSelection](#)

Beschreibung

Die Eigenschaft bietet Zugriff auf die aktuelle Auswahl in der Authentic-Ansicht.

EditClear (nicht mehr verwendet)**Wird ersetzt durch [AuthenticRange.Delete](#)**

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");
```

Siehe auch

Deklaration: [EditClear](#)

Beschreibung

Löscht die aktuelle Auswahl.

EditCopy(nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.Copy](#)

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");
```

Siehe auch

Deklaration: [EditCopy](#)

Beschreibung

Kopiert die aktuelle Auswahl in die Zwischenablage.

EditCut (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.Cut](#)

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");
```

Siehe auch

Deklaration: [EditCut](#)

Beschreibung

Schneidet die aktuelle Auswahl aus dem Dokument aus und kopiert sie in die Zwischenablage.

EditPaste (nicht mehr verwendet)**Wird ersetzt durch [AuthenticRange.Paste](#)**

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");
```

Siehe auch

Deklaration: [EditPaste](#)

Beschreibung

Fügt den Inhalt aus der Zwischenablage in das Dokument ein.

EditRedo (nicht mehr verwendet)**Wird ersetzt durch [AuthenticView.Redo](#)**

```
// ----- javascript sample -----  
// instead of:  
// Application.ActiveDocument.DocEditView.EditRedo();  
// use now:  
if (! Application.ActiveDocument.AuthenticView.Redo())  
    MsgBox ("Error: no redo step available");
```

Siehe auch

Deklaration: [EditRedo](#)

Beschreibung

Wiederholt den zuletzt rückgängig gemachten Schritt.

EditSelectAll (nicht mehr verwendet)

Wird ersetzt durch [AuthenticView.WholeDocument](#) and [AuthenticRange.Select](#)

```
// ----- javascript sample -----  
// instead of:  
// Application.ActiveDocument.DocEditView.EditSelectAll();  
// use now:  
Application.ActiveDocument.AuthenticView.WholeDocument.Select();
```

Siehe auch

Deklaration: [EditSelectAll](#)

Beschreibung

Die Methode wählt das gesamte Dokument aus.

EditUndo (nicht mehr verwendet)

Wird ersetzt durch [AuthenticView.Undo](#)

```
// ----- javascript sample -----  
// instead of:  
// Application.ActiveDocument.DocEditView.EditUndo();  
// use now:  
if (! Application.ActiveDocument.AuthenticView.Undo())  
    MsgBox ("Error: no undo step available");
```

Siehe auch

Deklaration: [EditUndo](#)

Beschreibung

Macht die letzte Aktion rückgängig.

event (nicht mehr verwendet)

Wird ersetzt durch Parameter für [AuthenticView events](#).

Siehe auch

Dkclaration: `event` als [DocEditEvent](#)

Beschreibung

Die Event-Eigenschaft enthält ein `DocEditEvent` Objekt, das Informationen über das aktuelle Event enthält.

GetAllowedElements (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.CanPerformActionWith](#)

`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](#).

```
// ----- 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);
```

Siehe auch

Deklaration: `GetAllowedElements` (*nAction* als [SpyAuthenticElementActions](#), *pStartElement* als [XMLData](#), *pEndElement* als [XMLData](#), *pElements* als Variant)

Beschreibung

`GetAllowedElements()` gibt die zulässigen Elemente für die verschiedenen von *nAction* definierten Aktionen zurück.

JavaScript-Beispiel:

```
Function GetAllowed()  
{  
    var objView = Application.ActiveDocument.DocEditView;
```

```
var arrElements = New Array(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";
}

Return strText;
}

Function ListArray(arrIn)
{
    var strText = "";

    If( TypeOf(arrIn) == "object") {
        For( var i = 0; i <= (arrIn.length - 1); i++)
            strText = strText + arrIn[i] + "\n";
    }

    Return strText;
}
```

VBScript-Beispiel:

```
Sub DisplayAllowed
    Dim objView
    Set objView = Application.ActiveDocument.DocEditView

    Dim arrElements()

    Dim objStart
    Dim objEnd
    Set objStart = objView.CurrentSelection.Start
    Set objEnd = objView.CurrentSelection.End

    Dim strText
    strText = "valid elements at current selection:" & chr(13) & chr(13)

    Dim i

    For i = 1 To 4
        objView.GetAllowedElements i, objStart, objEnd, arrElements
        strText = strText & ListArray(arrElements) & "-----" & chr(13)
    Next

    msgbox strText
End Sub

Function ListArray(arrIn)
    Dim strText

    If IsArray(arrIn) Then
        Dim i

        For i = 0 To UBound(arrIn)
            strText = strText & arrIn(i) & chr(13)
        Next
    End If

    ListArray = strText
End Function
```

End Function

GetNextVisible (nicht mehr verwendet)**Wird ersetzt durch [AuthenticRange.SelectNext](#)**

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);  
}
```

Siehe auch

Deklaration: `GetNextVisible` (*pElement* als [XMLData](#)) als [XMLData](#)

Beschreibung

Die Methode ruft das nächste sichtbare XML-Element im Dokument ab.

GetPreviousVisible (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.SelectPrevious](#)

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);  
}
```

Siehe auch

Deklaration: [GetPreviousVisible](#) ([pElement](#) als [XMLData](#)) als [XMLData](#)

Beschreibung

Die Methode ruft das vorhergehende sichtbare XML-Element im Dokument ab.

IsEditClearEnabled (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.IsDeleteEnabled](#)

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();
```

Siehe auch

Deklaration: [IsEditClearEnabled](#) als Boolean

Beschreibung

True, wenn [EditClear](#) möglich ist. Siehe auch [Bearbeitungsoperationen](#).

IsEditCopyEnabled (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.IsCopyEnabled](#)

Die IsCopyEnabled-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();
```

Siehe auch

Deklaration: [IsEditCopyEnabled](#) als Boolean

Beschreibung

True, wenn das Kopieren in die Zwischenablage möglich ist. Siehe auch [EditCopy](#) und [Bearbeitungsoperationen](#).

IsEditCutEnabled (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.IsCutEnabled](#)

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();
```

Siehe auch

Deklaration: [IsEditCutEnabled](#) als Boolean

Beschreibung

True, wenn [EditCut](#) derzeit möglich ist. Siehe auch [Bearbeitungsoperationen](#).

IsEditPasteEnabled (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.IsPasteEnabled](#)

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();
```

Siehe auch

Deklaration: [IsEditPasteEnabled](#) als Boolean

Beschreibung

True, wenn [EditCut](#) derzeit möglich ist. Siehe auch [Bearbeitungsoperationen](#).

IsEditRedoEnabled (nicht mehr verwendet)

Wird ersetzt durch [AuthenticView.IsRedoEnabled](#)

```
// ----- javascript sample -----  
// instead of:  
// if ( Application.ActiveDocument.DocEditView.IsEditRedoEnabled)  
//     Application.ActiveDocument.DocEditView.EditRedo();  
// use now:  
if ( Application.ActiveDocument.AuthenticView.IsRedoEnabled)  
    Application.ActiveDocument.AuthenticView.Redo();
```

Siehe auch

Deklaration: [IsEditRedoEnabled](#) als Boolean

Beschreibung

True, wenn [EditRedo](#) derzeit möglich ist. Siehe auch [Bearbeitungsoperationen](#).

IsEditUndoEnabled (nicht mehr verwendet)

Wird ersetzt durch [AuthenticView.IsUndoEnabled](#)

```
// ----- javascript sample -----  
// instead of:  
// if ( Application.ActiveDocument.DocEditView.IsEditUndoEnabled)  
//     Application.ActiveDocument.DocEditView.EditUndo();  
// use now:  
if ( Application.ActiveDocument.AuthenticView.IsUndoEnabled)  
    Application.ActiveDocument.AuthenticView.Undo();
```

Siehe auch

Deklaration: [IsEditUndoEnabled](#) als Boolean

Beschreibung

True, wenn [EditUndo](#) derzeit möglich ist. Siehe auch [Bearbeitungsoperationen](#).

IsRowAppendEnabled (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.IsInDynamicTable](#)

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();
```

Siehe auch

Deklaration: [IsRowAppendEnabled](#) als Boolean

Beschreibung

True, wenn [RowAppend](#) möglich ist. Siehe auch [Zeilenoperationen](#).

IsRowDeleteEnabled (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.IsInDynamicTable](#)

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();
```

Siehe auch

Deklaration: [IsRowDeleteEnabled](#) als Boolean

Beschreibung

True, wenn [RowDelete](#) möglich ist. Siehe auch [Zeilenoperationen](#).

IsRowDuplicateEnabled (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.IsInDynamicTable](#)

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();
```

Siehe auch

Deklaration: `IsRowDuplicateEnabled` als Boolean

Beschreibung

True, wenn [RowDuplicate](#) möglich ist. Siehe auch [Zeilenoperationen](#).

IsRowInsertEnabled (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.IsInDynamicTable](#)

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();
```

Siehe auch

Deklaration: `IsRowInsertEnabled` als Boolean

Beschreibung

True, wenn [RowInsert](#) möglich ist. Siehe auch [Zeilenoperationen](#).

IsRowMoveDownEnabled (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.IsLastRow](#)

```
// ----- 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( );
```

Siehe auch

Deklaration: [IsRowMoveDownEnabled](#) als Boolean

Beschreibung

True, wenn [RowMoveDown](#) möglich ist. Siehe auch [Zeilenoperationen](#).

IsRowMoveUpEnabled (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.IsFirstRow](#)

```
// ----- 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( );
```

Siehe auch

Deklaration: [IsRowMoveUpEnabled](#) als Boolean

Beschreibung

True, wenn [RowMoveUp](#) möglich ist. Siehe auch [Zeilenoperationen](#).

IsTextStateApplied (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.IsTextStateApplied](#)

```
// ----- 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");
```

Siehe auch

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".

IsTextStateEnabled (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.CanPerformAction](#)

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
```

Siehe auch

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".

LoadXML (nicht mehr verwendet)**Wird ersetzt durch [AuthenticView.AsXMLString](#)**

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"); }
```

Siehe auch

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>");`

MarkUpView (nicht mehr verwendet)**Wird ersetzt durch [AuthenticView.MarkupVisibility](#)**

```
// ----- javascript sample -----  
// instead of:  
// Application.ActiveDocument.DocEditView.MarkuUpView = 2;  
// use now:  
Application.ActiveDocument.AuthenticView.MarkupVisibility =  
spyAuthenticMarkupLarge;
```

Siehe auch

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

RowAppend (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.AppendRow](#)

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");
```

Siehe auch

Deklaration: [RowAppend](#)

Beschreibung

Hängt eine Zeile an der aktuellen Position an.

Siehe auch [Zeilenoperationen](#).

RowDelete (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.DeleteRow](#)

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");
```

Siehe auch

Deklaration: [RowDelete](#)

Beschreibung

Löscht die aktuell ausgewählte(n) Zeile(n).

Siehe auch [Zeileoperationen](#).

RowDuplicate (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.DuplicateRow](#)

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");
```

Siehe auch

Deklaration: [RowDuplicate](#)

Beschreibung

Die Methode kopiert die aktuell ausgewählten Zeilen.

Siehe auch [Zeilenoperationen](#).

RowInsert (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.InsertRow](#)

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");
```

Siehe auch

Deklaration: [RowInsert](#)

Beschreibung

Fügt eine neue Zeile unmittelbar oberhalb der aktuellen Auswahl ein.

Siehe auch [Zeilenoperationen](#).

RowMoveDown (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.MoveRowDown](#)

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");
```

Siehe auch

Deklaration: [RowMoveDown](#)

Beschreibung

Verschiebt die aktuelle Zeile um eine Position nach unten.

Siehe auch [Zeilenoperationen](#).

RowMoveUp (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.MoveRowUp](#)

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");
```

Siehe auch

Deklaration: [RowMoveUp](#)

Beschreibung

Verschiebt die aktuelle Zeile um eine Position nach oben.

Siehe auch [Zeilenoperationen](#).

SaveXML (nicht mehr verwendet)**Wird ersetzt durch [AuthenticView.AsXMLString](#)**

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"); }
```

Siehe auch

Deklaration: [SaveXML](#) als String

Rückgabewert

XML-Struktur als String.

Beschreibung

Speichert die aktuellen XML-Daten als String, der an den Aufrufenden zurückgegeben wird.

SelectionMoveTabOrder (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.SelectNext](#)

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
```

Siehe auch

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.

SelectionSet (nicht mehr verwendet)

Wird ersetzt durch [AuthenticRange.FirstXMLData](#) 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.

```
// ----- 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
```

Siehe auch

Deklaration: `SelectionSet` (*pStartElement* als [XMLData](#), *nStartPos* als long, *pEndElement* als [XMLData](#), *nEndPos* als long) als Boolean

Beschreibung

Verwenden Sie `SelectionSet()` um in der Authentic-Ansicht eine neue Auswahl zu definieren. Sie können `pEndElement` auf Null (nichts) setzen, wenn die Auswahl nur ein XML-Element (`pStartElement`) umfassen soll.

XMLRoot (nicht mehr verwendet)

Wird ersetzt durch [AuthenticView.XMLDataRoot](#)

```
// ----- javascript sample -----  
// instead of:  
// var objXMLData = Application.ActiveDocument.DocEditView.XMLRoot;  
// use now:  
var objXMLData = Application.ActiveDocument.AuthenticView.XMLDataRoot;
```

Siehe auch

Deklaration: [XMLRoot](#) als [XMLData](#)

Beschreibung

[XMLRoot](#) ist das übergeordnete Element der aktuell angezeigten XML-Struktur. Über die [XMLData](#)-Schnittstelle haben Sie vollen Zugriff auf den kompletten Dateiinhalt.

Nähere Informationen dazu siehe auch Verwenden von [XMLData](#).

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.

3.4.1 ENUMApplicationStatus

Beschreibung

Enumeration zur Definition des aktuellen Applikationsstatus.

Mögliche Werte:

eApplicationRunning	= 0
eApplicationAfterLicenseCheck	= 1
eApplicationBeforeLicenseCheck	= 2
eApplicationConcurrentLicenseCheckFailed	= 3
eApplicationProcessingCommandLine	= 4

3.4.2 SPYAttributeTypeDefinition

Beschreibung

Attributtypdefinition, die zur Generierung einer XML-Beispieldatei ausgewählt werden kann. Dieser Typ wird mit den Methoden [GenerateDTDOrSchema](#) und [GenerateDTDOrSchemaEx](#) verwendet.

Mögliche Werte:

spyMergedGlobal	= 0
spyDistinctGlobal	= 1
spyLocal	= 2

3.4.3 SPYAuthenticActions

Beschreibung

Aktionen, die an [AuthenticRange](#) Objekten durchgeführt werden können.

Mögliche Werte:

spyAuthenticInsertAt	= 0
spyAuthenticApply	= 1
spyAuthenticClearSurr	= 2
spyAuthenticAppend	= 3
spyAuthenticInsertBefore	= 4
spyAuthenticRemove	= 5

3.4.4 SPYAuthenticDocumentPosition

Beschreibung

Relative und absolute Position für die Navigation mit [AuthenticRange](#) Objekten.

Mögliche Werte:

spyAuthenticDocumentBegin	= 0
spyAuthenticDocumentEnd	= 1
spyAuthenticRangeBegin	= 2
spyAuthenticRangeEnd	= 3

3.4.5 SPYAuthenticElementActions

Beschreibung

Aktionen die mit [GetAllowedElements](#) verwendet werden können (Wird ersetzt durch [AuthenticRange.CanPerformActionWith](#)).

Mögliche Werte:

k_ActionInsertAt	= 0
k_ActionApply	= 1
k_ActionClearSurr	= 2
k_ActionAppend	= 3
k_ActionInsertBefore	= 4
k_ActionRemove	= 5

3.4.6 SPYAuthenticElementKind

Beschreibung

Enumeration der verschiedenen Arten von Elementen, die für die Navigation und Auswahl innerhalb der [AuthenticRange](#) und [AuthenticView](#) Objekte verwendet werden.

Mögliche Werte:

spyAuthenticChar	= 0
spyAuthenticWord	= 1
spyAuthenticLine	= 3
spyAuthenticParagraph	= 4
spyAuthenticTag	= 6
spyAuthenticDocument	= 8
spyAuthenticTable	= 9
spyAuthenticTableRow	= 10
spyAuthenticTableColumn	= 11

3.4.7 SPYAuthenticMarkupVisibility

Beschreibung

Enumerationswerte zum Anpassen der Sichtbarkeit von Markup-Tags mit [MarkupVisibility](#).

Mögliche Werte:

spyAuthenticMarkupHidden	= 0
spyAuthenticMarkupSmall	= 1
spyAuthenticMarkupLarge	= 2
spyAuthenticMarkupMixed	= 3

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

3.4.9 SPYDatabaseKind

Beschreibung

Werte zur Auswahl verschiedener Datenbankarten für den Import. Informationen zur Verwendung finden Sie unter [DatabaseConnection.DatabaseKind](#).

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

3.4.10 SPYDialogAction

Beschreibung

Werte zur Simulation verschiedener Interaktionen in Dialogfeldern. Informationen dazu, welche Dialogfelder zur Verfügung stehen, finden Sie unter [Dialogs](#).

Mögliche Werte:

spyDialogOK	= 0	//simulate click on OK button
spyDialogCancel	= 1	//simulate click on Cancel button
spyDialogUserInput	= 2	// show dialog and allow user interaction

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.

3.4.12 SPYDTDSchemaFormat

Beschreibung

Enumeration zum Identifizieren der verschiedenen Schemaformate.

Mögliche Werte:

spyDTD	= 0
spyW3C	= 1

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

3.4.14 SPYExportNamespace

Beschreibung

Enumerationstyp, um die Behandlung von Namespace Identifiers beim Export zu konfigurieren.

Mögliche Werte:

spyNoNamespace	= 0
spyReplaceColonWithUnderscore	= 1

3.4.15 SPYFindInFilesSearchLocation

Beschreibung

Die verschiedenen Optionen, wo eine Suche durchgeführt werden kann. Dieser Typ wird mit dem Dialogfeld [FindInFilesDlg](#) verwendet.

Mögliche Werte:

spyFindInFiles_Documents	= 0
spyFindInFiles_Project	= 1
spyFindInFiles_Folder	= 2

3.4.16 SPYFrequentElements

Beschreibung

Enumerationswert zum Parametrisieren der Schemagenerierung.

Mögliche Werte:

spyGlobalElements	= 0
spyGlobalComplexType	= 1

3.4.17 SPYImageKind

Beschreibung

Enumerationswerte zum Parametrisieren des Bildtyps der generierten Dokumentation. Diese Werte werden in [SchemaDocumentationDiagramFormat](#) und [WSDLDocumentationDiagramFormat](#) verwendet.

Mögliche Werte:

spyImageType_PNG	= 0
spyImageType_EMF	= 1

3.4.18 SPYImportColumnsType

Beschreibung

Enumeration zur Definition der verschiedenen Importspaltenarten.

Mögliche Werte:

spyImportColumns_Element	= 0
spyImportColumns_Attribute	= 1

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

3.4.20 SPYKeyStatus**Beschreibung**

Enumerationstyp zum Identifizieren des Tastenstatus.

Mögliche Werte:

spyLeftShiftKeyMask	= 1
spyRightShiftKeyMask	= 2
spyLeftCtrlKeyMask	= 4
spyRightCtrlKeyMask	= 8
spyLeftAltKeyMask	= 16
spyRightAltKeyMask	= 32

3.4.21 SPYLibType**Beschreibung**

Enumerationswerte zum Parametrisieren der Generierung von C++ Code anhand von Schemadefinitionen.

Mögliche Werte:

spyLibType_static	= 0
spyLibType_dll	= 1

3.4.22 SPYLoading**Beschreibung**

Enumerationswerte zum Definieren des Ladeverhaltens von URL-Dateien.

Mögliche Werte:

spyUseCacheProxy	= 0
spyReload	= 1

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 Or
XMLSpyLib.spyCtrlKeyDownMask)) Then
    ' react on ctrl-leftbutton-down
End If

' to check for double-click with any button in VBScript
If (((i_eMouseEvent And spyDoubleClickMask) <> 0) Then
    ' react on double-click
End If

```

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

3.4.24 SPYNumberDateTimeFormat**Beschreibung**

Enumerationswert zum Konfigurieren von Datenbankverbindungen.

Mögliche Werte:

spySystemLocale	= 0
spySchemaCompatible	= 1

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

3.4.26 SPYProjectItemTypes

Beschreibung

Enumerationswerte zum Identifizieren der verschiedenen Elemente in Projektelementlisten.
Siehe [SpyProjectItem.ItemType](#).

Mögliche Werte:

spyUnknownItem	= 0
spyFileItem	= 1
spyFolderItem	= 2
spyURLItem	= 3

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	
spyVisualStudio2005Project	= 4	Auch für C++-Code
spyVisualStudio2008Project	= 5	Auch für C++-Code
spyVisualStudio2010Project	= 6	Auch für C++-Code

3.4.28 SPYSampleXMLGenerationOptimization

Beschreibung

Definiert die Elemente, die in der XML-Beispieldatei generiert werden.
Diese Enumeration wird in [GenerateSampleXMLDlg](#) verwendet.

Mögliche Werte:

spySampleXMLGen_Optimized	= 0
spySampleXMLGen_NonMandatoryElements	= 1
spySampleXMLGen_Everything	= 2

3.4.29 SPYSampleXMLGenerationSchemaOrDTDAssignment

Beschreibung

Definiert, welche Art von Referenz zum Schema/zur DTD zur generierten XML-Beispieldatei hinzugefügt werden soll.
Diese Enumeration wird in [GenerateSampleXMLDlg](#) verwendet.

Mögliche Werte:

spySampleXMLGen_AssignRelatively	= 0
spySampleXMLGen_AssignAbsolutely	= 1
spySampleXMLGen_DoNotAssign	= 2

3.4.30 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

3.4.31 SPYSchemaDocumentationFormat

Beschreibung

Enumerationswerte zum Parametrisieren der Generierung von Schemadokumentation. Diese

Werte werden in [SchemaDocumentationDiagramOutputFormat](#) und

WSDLDocumentationDiagramOutputFormat verwendet.

Mögliche Werte:

spySchemaDoc_HTML	= 0
spySchemaDoc_MSWord	= 1
spySchemaDoc_RTF	= 2
spxSchemaDoc_PDF	= 3

3.4.32 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

3.4.33 SPYSchemaFormat

Beschreibung

Enumeration zur Definition der unterschiedlichen Schemaformattypen.

Mögliche Werte:

spySchemaFormat_Hierarchical	= 0
spySchemaFormat_Flat	= 1

3.4.34 SPYTextDelimiters

Beschreibung

Enumerationswerte zum Definieren der Texttrennzeichen für den Textexport.

Mögliche Werte:

spyTabulator	= 0
spySemicolon	= 1
spyComma	= 2
spySpace	= 3

3.4.35 SPYTextEnclosing

Beschreibung

Enumerationswert zum Definieren der den Text einschließenden Zeichen beim Textimport und -export.

Mögliche Werte:

spyNoEnclosing	= 0
spySingleQuote	= 1
spyDoubleQuote	= 2

3.4.36 SPYTypeDetection

Beschreibung

Enumeration, um auszuwählen, wie die Typerkennung bei [GenerateDTDOrSchema](#) und [GenerateDTDOrSchemaEx](#) funktioniert.

Mögliche Werte:

spyBestPossible	= 0
spyNumbersOnly	= 1
spyNoDetection	= 2

3.4.37 SPYURLTypes

Beschreibung

Enumeration zum Definieren der verschiedenen URL-Typen.

Mögliche Werte:

spyURLTypeAuto	= -1
spyURLTypeXML	= 0
spyURLTypeDTD	= 1

3.4.38 SPYViewModes

Beschreibung

Enumerationenwerte, 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 *spyViewWSDL* identifiziert einen Modus, der auf der Benutzeroberfläche der Schema-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	

3.4.39 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 And spyCtrlKeyMask) <> 0)) Then
    ' ctrl-key is pressed
End If

' VBScript sample: check if ONLY ctrl-key is pressed
If (i_nVirtualKeyStatus == spyCtrlKeyMask) Then
    ' exactly ctrl-key is pressed
End If

// 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

3.4.40 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

3.5 Applikations-API für Java

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](#).

Die Applikations-API in Java hat eine Benutzeroberfläche, die auf Java-Klassen aufbaut, wobei jede dieser Klassen einem Objekt in der [Applikations-API](#) 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](#)-Objekte selbst finden Sie in der [Dokumentation zur Applikations-API](#). Beachten Sie, dass einige API-Funktionen nur in Skripting-Umgebungen zur Verfügung stehen; diese wurden daher nicht nach Java portiert.

Java-Klassen

- [SpyApplication](#)
- [SpyProject](#)
- [SpyProjectItems](#)
- [SpyProjectItem](#)
- [SpyDocuments](#)
- [SpyDoc](#)
- [SpyAuthenticView](#)
- [SpyAuthenticRange](#)
- [SpyDocEditView](#)
- [SpyDocEditSelection](#)
- [SpyGridView](#)
- [SpyTextView](#)
- [SpyXMLData](#)
- [SpyDialogs](#)
- [SpyCodeGeneratorDlg](#)
- [SpyDTDSchemaGeneratorDlg](#)
- [SpyFileSelectionDlg](#)
- [SpyFindInFilesDlg](#)
- [SpyGenerateSampleXMLDlg](#)
- [SpySchemaDocumentationDlg](#)
- [SpyWSDL20DocumentationDlg](#)
- [SpyWSDLDocumentationDlg](#)
- [SpyXBRLDocumentationDlg](#)
- [SpyDatabaseConnection](#)
- [SpyElementList](#)
- [SpyElementListItem](#)
- [SpyExportSettings](#)
- [SpyFindInFilesResults](#)
- [SpyFindInFilesResult](#)
- [SpyFindInFilesMatch](#)
- [SpyTextImportExportSettings](#)

Implementierung von COM-Eigenschaften in Java

Die Eigenschaften in Java wurden so definiert, dass Sie sowohl eine `set` als auch eine `get`

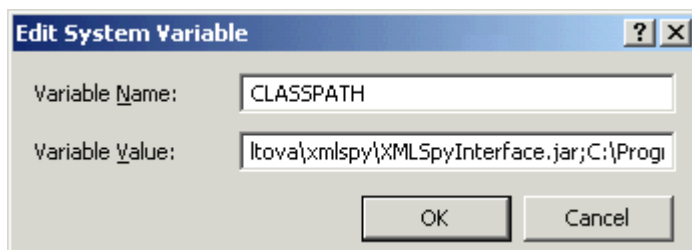
Methode enthalten (`set` wenn dies die COM-Implementierung zulässt). So enthält z.B. die COM-Klasse `Document` die Eigenschaft `GridView`. In Java heißt die Methode `SpyDoc` und die Eigenschaft ist als eine `GetGridView` Methode definiert.

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 Systemvariablen 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).

3.5.1 Beispielquellcode

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\\<Benutzername>\\Eigene Dateien\\Altova\\XMLSpy2012\\. 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
                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
```

3.5.2 SpyApplication

```
public class SpyApplication
{
    public void ReleaseInstance();
    public void ShowApplication( boolean bShow );
    public void Quit();
    public void AddMacroMenuItem( String sMacro, String sDisplayText );
    public void ClearMacroMenu();
    public SpyDoc GetActiveDocument();
    public SpyProject GetCurrentProject();
    public SpyDocuments GetDocuments();
    public SpyElementList GetDatabaseImportElementList( SpyDatabaseConnection
oImportSettings );
    public SpyDatabaseConnection GetDatabaseSettings();
    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 );
}

```

3.5.3 SpyCodeGeneratorDlg

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 GetApplication\(\);
    public SpyDialogs GetParent\(\);
    public long GetProgrammingLanguage\(\);
    public void SetProgrammingLanguage( long i_eVal );
    public String GetTemplateFileName\(\);
    public void SetTemplateFileName( String i_strVal );
    public String GetOutputPath\(\);
    public void SetOutputPath( String i_strVal );
    public long GetOutputPathDialogAction\(\);
    public void SetOutputPathDialogAction( long i_eVal );
    public long GetPropertySheetDialogAction\(\);
    public void SetPropertySheetDialogAction( long i_eVal );
    public long GetOutputResultDialogAction\(\);
    public void SetOutputResultDialogAction( long i_eVal );
    public long GetCPPSettings\_DOMType\(\);
    public void SetCPPSettings\_DOMType( long i_eVal );
    public long GetCPPSettings\_LibraryType\(\);
    public void SetCPPSettings\_LibraryType( long i_eVal );
    public boolean GetCPPSettings\_UseMFC\(\);
    public void SetCPPSettings\_UseMFC( boolean i_bVal );
    public long GetCSharpSettings\_ProjectType\(\);
    public void SetCSharpSettings\_ProjectType( long i_eVal );
}

```

3.5.4 SpyDatabaseConnection

```

public class SpyDatabaseConnection
{
    public void ReleaseInstance\(\);
    public String GetADOConnection\(\);
    public void SetADOConnection( String sValue );
    public boolean GetAsAttributes\(\);
    public void SetAsAttributes( boolean bValue );
    public boolean GetCreateMissingTables\(\);
    public void SetCreateMissingTables( boolean bValue );
    public boolean GetCreateNew\(\);
    public void SetCreateNew( boolean bValue );
    public boolean GetExcludeKeys\(\);
    public void SetExcludeKeys( boolean bValue );
    public String GetFile\(\);
    public void SetFile( String sValue );
}

```



```

public boolean GetIncludeEmptyElements();
public void SetIncludeEmptyElements( boolean bValue );
public long GetNumberDateTimeFormat();
public void SetNumberDateTimeFormat( long nValue );
public String GetODBCConnection();
public void SetODBCConnection( 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 )
}

```

3.5.5 SpyDialogs

```

// 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();
}

```

3.5.6 SpyDoc

```

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 GetApplication();
    public SpyDocuments GetParent();
    public String GetFullName();
    public void SetFullName( String i_strName );
    public String GetName();
    public String GetPath();
    public boolean GetSaved();
    public void SaveAs( String i_strFileNameOrPath );
    public String GetEncoding();
    public SpyXMLData GetDataRoot();
    public void GenerateProgramCode( SpyCodeGeneratorDlg i_dlg );
    public void AssignXSLFO( String i_strFile, boolean i_bUseDialog );
    public void TransformXSLFO();

```

```

public void GenerateSchemaDocumentation( SpySchemaDocumentationDlg i_dlg );

public void ExecuteXQuery( String i_strXMLSourceFile );
public void SetExternalIsValid( boolean bIsValid );
public SpyDoc GenerateSampleXML( SpyGenerateSampleXMLDlg ipGenerateXMLDlg );
public boolean UpdateXMLData();
public String GetAsXMLString();
public void SetAsXMLString( String newVal );
public SpyDoc GenerateDTDOrSchemaEx( SpyDTDSchemaGeneratorDlg
ipDTDSchemaGeneratorDlg );
public SpyDoc ConvertDTDOrSchemaEx( long nFormat, long nFrequentElements,
String sOutputPath, long nOutputPathDialogAction );
public SpyTextView GetTextView();
public String[] GetSuggestions();
public void SetSuggestions( String[] aList );
public void SetSelection( String sVal );

// Since version 2009
public void GenerateWSDLDocumentation( SpyWSDLDocumentationDlg
ipWSDLDocumentationDlg );
public void TransformXSLEx( long nDialogAction );

// Since version 2010
public void GenerateWSDL20Documentation( SpyWSDL20DocumentationDlg
ipWSDL20DocumentationDlg );
public void GenerateXBRLDocumentation( SpyXBRLDocumentationDlg
ipXBRLDocumentationDlg );
public SpyDoc ConvertToWSDL20( String sFilePath, boolean bShowDialogs );

// Since version 2010r3
public String CreateDBStructureFromXMLSchema( SpyDatabaseConnection
oConnection, SpyElementList oTables, boolean bDropTableWithExistingName );
public SpyElementList GetDBStructureList( SpyDatabaseConnection oConnection
);
}

```

3.5.7 SpyDocuments

```

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 strSPSPPath, String strXMLPath );
    public SpyDoc OpenAuthenticFile( String strSPSPPath, String strXMLPath );
}

```

3.5.8 SpyDTDSchemaGeneratorDlg

```

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 );
}

```

3.5.9 SpyElementList

```

public class SpyElementList
{
    public void ReleaseInstance();
    public long GetCount();
    public SpyElementListItem GetItem( long nIndex );
    public void RemoveElement( long nIndex );
}

```

3.5.10 SpyElementListItem

```

public class SpyElementListItem
{
    public void ReleaseInstance();
    public long GetElementKind();
    public void SetElementKind( long nKind );
    public long GetFieldCount();
    public String GetName();
    public long GetRecordCount();
}

```

3.5.11 SpyExportSettings

```

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();
    public void SetSubLevelLimit( int nValue );
}

```

3.5.12 SpyFileSelectionDlg

```

// 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 );
}

```

3.5.13 SpyFindInFilesDlg

```

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 );
}

```

3.5.14 SpyFindInFilesMatch

```

public class SpyFindInFilesMatch
{
    public void ReleaseInstance();
    public long Line();
    public long Position();
    public long Length();
    public String LineText();
    public boolean Replaced();
}

```

3.5.15 SpyFindInFilesResult

```

public class SpyFindInFilesResult
{
    public void ReleaseInstance();
    public long Count();
    public SpyFindInFilesMatch GetItem( long nNo );
    public String GetPath();
    public SpyDoc GetDocument();
}

```

3.5.16 SpyFindInFilesResults

```

public class SpyFindInFilesResults
{
    public void ReleaseInstance();
    public long Count();
    public SpyFindInFilesResult GetItem( long nNo );
}

```

3.5.17 SpyGenerateSampleXMLDlg

```

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 );
}

```

3.5.18 SpyGridView

```

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 );
}

```

3.5.19 SpyProject

```

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 );
}

```

3.5.20 SpyProjectItem

```
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();
    public void SetXSLTransformationFileExtension( String sVal );
    public String GetXSLTransformationFolder();
    public void SetXSLTransformationFolder( String sVal );
}
```

3.5.21 SpyProjectItems

```
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 );
}
```

3.5.22 SpySchemaDocumentationDlg

```
// 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 GetIncludeLocalElements();
public void SetIncludeLocalElements( boolean i_bVal );
public boolean GetIncludeGroups();
public void SetIncludeGroups( boolean i_bVal );
public boolean GetIncludeComplexTypes();
public void SetIncludeComplexTypes( boolean i_bVal );
public boolean GetIncludeSimpleTypes();
public void SetIncludeSimpleTypes( boolean i_bVal );
public boolean GetIncludeAttributeGroups();
public void SetIncludeAttributeGroups( boolean i_bVal );
public boolean GetIncludeRedefines();
public void SetIncludeRedefines( boolean i_bVal );

public void AllDetails( boolean i_bDetailsOn );
public boolean GetShowDiagram();
public void SetShowDiagram( boolean i_bVal );
public boolean GetShowNamespace();
public void SetShowNamespace( boolean i_bVal );
public boolean GetShowType();
public void SetShowType( boolean i_bVal );
public boolean GetShowChildren();
public void SetShowChildren( boolean i_bVal );
public boolean GetShowUsedBy();
public void SetShowUsedBy( boolean i_bVal );
public boolean GetShowProperties();
public void SetShowProperties( boolean i_bVal );
public boolean GetShowSingleFacets();
public void SetShowSingleFacets( boolean i_bVal );
public boolean GetShowPatterns();
public void SetShowPatterns( boolean i_bVal );
public boolean GetShowEnumerations();
public void SetShowEnumerations( boolean i_bVal );
public boolean GetShowAttributes();
public void SetShowAttributes( boolean i_bVal );
public boolean GetShowIdentityConstraints();
public void SetShowIdentityConstraints( boolean i_bVal );
public boolean GetShowAnnotations();
public void SetShowAnnotations( boolean i_bVal );
public boolean GetShowSourceCode();
public void SetShowSourceCode( boolean i_bVal );

// Since version 2009
public boolean GetEmbedDiagrams();
public void SetEmbedDiagrams( boolean i_bVal );
public long GetDiagramFormat();
public void SetDiagramFormat( long i_nVal );
public boolean GetIncludeGlobalAttributes();
public void SetIncludeGlobalAttributes( boolean i_bVal );
public boolean GetIncludeLocalAttributes();
public void SetIncludeLocalAttributes( boolean i_bVal );
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 );
}

```

3.5.23 SpyTextImportExportSettings

```

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 );
}

```

3.5.24 SpyTextView

```

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();
}
```

3.5.25 SpyWSDL20DocumentationDlg

```
// 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 );

    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 );
}

```

3.5.26 SpyWSDLDocumentationDlg

```

// Since version 2008r2spl
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 );

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 );
}

```

3.5.27 SpyXBRLDocumentationDlg

```

// 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 GetShortQualifiedNames();
    public void SetShortQualifiedNames( 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 );
};

```

3.5.28 SpyXMLData

```

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 );
}

```

3.5.29 Authentic

SpyAuthenticRange

```

// 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 GotoPreviousCursorPosition();
public boolean IsEmpty();
public boolean IsEqual( SpyAuthenticRange ipCmp );
public SpyAuthenticRange Clone();
public SpyAuthenticRange SetFromRange( SpyAuthenticRange ipSrc );
public boolean Delete();
public boolean Cut();
public boolean Copy();
public boolean Paste();
public SpyXMLData GetFirstXMLData();
public void SetFirstXMLData( SpyXMLData objXMLDataPtr );
public long GetFirstXMLDataOffset();
public void SetFirstXMLDataOffset( long nOffset );
public SpyXMLData GetLastXMLData();
public void SetLastXMLData( SpyXMLData objXMLDataPtr );
public long GetLastXMLDataOffset();
public void SetLastXMLDataOffset( long nOffset );
public String[] GetElementHierarchy();
public String[] GetElementAttributeNames( String strElementName );
public boolean HasElementAttribute( String strElementName, String
strAttributeName );
public String GetElementAttributeValue( String strElementName, String
strAttributeName );
public void SetElementAttributeValue( String strElementName, String
strAttributeName, String strNewValue );
public String[] GetEntityNames();
public void InsertEntity( String strEntityName );
public boolean IsInDynamicTable();
public boolean AppendRow();
public boolean InsertRow();
public boolean DuplicateRow();
public boolean DeleteRow();
public boolean MoveRowUp();
public boolean MoveRowDown();

// Since version 2004R4
public boolean IsCopyEnabled();
public boolean IsCutEnabled();
public boolean IsPasteEnabled();
public boolean IsDeleteEnabled();
public boolean IsTextStateApplied( String i_strElementName );
public boolean IsFirstRow();
public boolean IsLastRow();
}

```

SpyAuthenticView

```
// Since version 2004R3
public class SpyAuthenticView
{
    public void ReleaseInstance\(\);
    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\(\);
}
```

SpyDocEditSelection

```
public class SpyDocEditSelection
{
    public void ReleaseInstance\(\);
    public SpyXMLData GetEnd\(\);
    public long GetEndTextPosition\(\);
    public SpyXMLData GetStart\(\);
    public long GetStartTextPosition\(\);
}
```

SpyDocEditView

```
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 );
}

```

3.5.30 Vordefinierte Konstanten

In diesem Abschnitt sind alle Klassen aufgelistet, die die von der Java-Schnittstelle verwendeten vordefinierten Konstanten verwenden.

SPYApplicationStatus

```

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 spyApplicationStatus_ConcurrentLicenseCheckFailed = 3;
    public final static long spyApplicationStatus_ProcessingCommandLine = 4;
}

```

SPYAttributeTypeDefinition

```

public class SPYAttributeTypeDefinition
{
    public final static long spyMergedGlobal = 0;
    public final static long spyDistinctGlobal = 1;
    public final static long spyLocal = 2;
}

```

SPYAuthenticActions

```
public class SPYAuthenticActions
{
    public final static long          = 0;
    spyAuthenticInsertAt
    public final static long spyAuthenticApply= 1;
    public final static long          = 2;
    spyAuthenticClearSurr
    public final static long          = 3;
    spyAuthenticAppend
    public final static long          = 4;
    spyAuthenticInsertBefore
    public final static long          = 5;
    spyAuthenticRemove
}
```

SPYAuthenticDocumentPosition

```
public class SPYAuthenticDocumentPosition
{
    public final static long          = 0;
    spyAuthenticDocumentBegin
    public final static long          = 1;
    spyAuthenticDocumentEnd
    public final static long          = 2;
    spyAuthenticRangeBegin
    public final static long          = 3;
    spyAuthenticRangeEnd
}
```

SPYAuthenticElementKind

```
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
}
```

SPYAuthenticMarkupVisibility

```
public class SPYAuthenticMarkupVisibility
{
    public final static long      spyAuthenticMarkupHidden      = 0;
    public final static long      spyAuthenticMarkupSmall       = 1;
    public final static long      spyAuthenticMarkupLarge       = 2;
    public final static long      spyAuthenticMarkupMixed       = 3;
}
```

SPYDatabaseKind

```
public class SPYLoading
{
    public final static long      spyDB_Access                  = 0;
    public final static long      spyDB_SQLServer              = 1;
    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      spyDB_Unspecified            = 7;
}
```

SPYDialogAction

```
public class SPYDialogAction
{
    public final static long      spyDialogOK                   = 0;
    public final static long      spyDialogCancel               = 1;
    public final static long      spyDialogUserInput            = 2;
}
```

SPYDOMType

```
public class SPYDOMType
{
    public final static long      spyDOMType_msxml4             = 0;
}
```

```
        public final static long          = 1;
        spyDOMType xerces
    }
```

SPYDTDSchemaFormat

```
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;
}
```

SPYEncodingByteOrder

```
public class SPYEncodingByteOrder
{
    public final static long spyNONE          = 0;
    public final static long                  = 1;
    spyLITTLE_ENDIAN
    public final static long                  = 2;
    spyBIG_ENDIAN
}
```

SPYExportNamespace

```
public class SPYExportNamespace
{
    public final static long spyNoNamespace          = 0;
    public final static long spyReplaceColonWithUnderscore = 1;
}
```

SPYFindInFilesSearchLocation

```
public class SPYFindInFilesSearchLocation
{
    public final static long          = 0;
    spyFindInFiles Documents
    public final static long          = 1;
    spyFindInFiles_Project
    public final static long          = 2;
    spyFindInFiles Folder
}
```

SPYFrequentElements

```
public class SPYFrequentElements
{
    public final static long          = 0;
    spyGlobalElements
}
```

```
    public final static long          = 1;
    spyGlobalComplexType
}
```

SPYImageKind

```
public class SPYImageKind
{
    public final static long          = 0;
    spyImageType_PNG
    public final static long          = 1;
    spyImageType_EMF
}
```

SPYLibType

```
public class SPYLibType
{
    public final static long          = 0;
    spyLibType_static
    public final static long          = 1;
    spyLibType_dll
}
```

SPYLoading

```
public class SPYLoading
{
    public final static long          = 0;
    spyUseCacheProxy
    public final static long          = 1;
    spyReload
}
```

SPYNumberDateTimeFormat

```
public class SPYNumberDateTimeFormat
{
    public final static long          = 0;
    spySystemLocale
    public final static long          = 1;
    spySchemaCompatible
}
```

SPYProgrammingLanguage

```
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;
}
```

SPYProjectItemTypes

```
public class SPYProjectItemTypes
{
    public final static long      = 0;
    spyUnknownItem
    public final static long      = 1;
    spyFileItem
    public final static long      = 2;
    spyFolderItem
    public final static long      = 3;
    spyURLItem
}
```

SPYProjectType

```
public class SPYProjectType
{
    public final static long      = 0;
    spyVisualStudioProject
    public final static long      = 1;
    spyVisualStudio2003Project
    public final static long      = 2;
    spyBorlandProject
    public final static long      = 3;
    spyMonoMakefile
}
```

SPYSampleXMLGenerationOptimization

```
public class SPYSampleXMLGenerationOptimization
{
    public final static long      = 0;
    spySampleXMLGen Optimized
    public final static long      = 1;
    spySampleXMLGen_NonMandatoryElements
    public final static long      = 2;
    spySampleXMLGen Everything
}
```

SPYSampleXMLGenerationSchemaOrDTDAssignment

```
public class SPYSampleXMLGenerationOptimization
{
    public final static long      = 0;
    spySampleXMLGen_AssignRelatively
    public final static long      = 1;
    spySampleXMLGen_AssignAbsolutely
    public final static long      = 2;
    spySampleXMLGen_DoNotAssign
}
```


SPYSchemaDefKind

```
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 spyKindIdentitySelector = 12;
    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;
}
```

SPYSchemaDocumentationFormat

```
public class SPYSchemaDocumentationFormat
{
    public final static long spySchemaDoc_HTML = 0;
    public final static long spySchemaDoc_MSWord = 1;
    public final static long spySchemaDoc_RTF = 2;
    public final static long spySchemaDoc_PDF = 3;
}
```

SPYSchemaExtensionType

```
public class SPYSchemaExtensionType
{
    public final static long spySchemaExtension_None = 0;
}
```

```
    public final static long                = 1;
    spySchemaExtension SQL XML
    public final static long                = 2;
    spySchemaExtension MS SQL Server
    public final static long                = 3;
    spySchemaExtension_Oracle
}
```

SPYSchemaFormat

```
public class SPYSchemaFormat
{
    public final static long                = 0;
    spySchemaFormat Hierarchical
    public final static long                = 1;
    spySchemaFormat_Flat
}
```

SPYTextDelimiters

```
public class SPYTextDelimiters
{
    public final static long                = 0;
    spyTabulator
    public final static long                = 1;
    spySemicolon
    public final static long spyComma      = 2;
    public final static long spySpace      = 3;
}
```

SPYTextEnclosing

```
public class SPYTextEnclosing
{
    public final static long                = 0;
    spyNoEnclosing
    public final static long                = 1;
    spySingleQuote
    public final static long                = 2;
    spyDoubleQuote
}
```

SPYTypeDetection

```
public class SPYTypeDetection
{
    public final static long                = 0;
    spyBestPossible
    public final static long                = 1;
    spyNumbersOnly
    public final static long                = 2;
    spyNoDetection
}
```

SPYURLTypes

```
public class SPYURLTypes
{
    public final static long      = (-1);
    spyURLTypeAuto
    public final static long      = 0;
    spyURLTypeXML
    public final static long      = 1;
    spyURLTypeDTD
}
```

SpyViewModes

```
public class SPYViewModes
{
    public final static long      = 0;
    spyViewGrid
    public final static long      = 1;
    spyViewText
    public final static long      = 2;
    spyViewBrowser
    public final static long      = 3;
    spyViewSchema
    public final static long      = 4;
    spyViewContent
    public final static long      = 4;
    spyViewAuthentic
    public final static long      = 5;
    spyViewWSDL
    public final static long spyViewZIP = 6;
    public final static long      = 7;
    spyViewEditionInfo
}
```

SPYWhitespaceComparison

```
public class SPYWhitespaceComparison
{
    public final static long      = 0;
    spyCompareAsIs
    public final static long      = 1;
    spyCompareNormalized
    public final static long      = 2;
    spyStripAll
}
```

SPYXMLDataKind

```
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          = 3;
    spyXMLDataXML
    public final static long          = 4;
    spyXMLDataElement
    public final static long          = 5;
    spyXMLDataAttr
    public final static long          = 6;
    spyXMLDataText
    public final static long          = 7;
    spyXMLDataCDATA
    public final static long          = 8;
    spyXMLDataComment
    public final static long          = 9;
    spyXMLDataPI
    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
}
```

4 ActiveX Integration

XMLSpyControl ist ein Control, das ein Mittel zur Integration der XMLSpy Benutzeroberfläche und der in diesem Abschnitt beschriebenen Funktionalitäten in die meisten Applikationsarten darstellt. Es wurde die ActiveX-Technologie gewählt, um die Integration mittels jeder beliebigen der vielen Programmiersprachen zu ermöglichen; dadurch können für die Integration C++, C#, VisualBasic oder HTML verwendet werden. ActiveX-Komponenten funktionieren offiziell nur mit Microsoft Internet Explorer. Alle Komponenten sind vollständige OLE Controls, wodurch die Integration so einfach wie möglich gemacht wird. Es stehen zwei verschiedene Ebenen der Integration zur Verfügung, sodass die Integration an die verschiedensten Bedürfnisse angepasst werden kann.

Um XMLSpy zu integrieren, müssen Sie das XMLSpy Integrationspaket installieren. Stellen Sie sicher, dass Sie XMLSpy zuerst installieren und dann erst das XMLSpy Integrationspaket.

Für eine erfolgreiche Integration müssen beim Design die folgenden Hauptfaktoren berücksichtigt werden:

- Welche Technologie oder Programmiersprache kann die Host-Applikation zur Integration des XMLSpyControl verwenden?
- Soll die integrierte Benutzeroberfläche genau wie XMLSpy mit all seinen Menüs, Symbolleisten und Fenstern aussehen oder wäre eine Untermenge dieser Elemente - z.B. dass nur ein Dokument und eine eingeschränkte Anzahl an Befehlen verwendet werden kann - sinnvoller?
- Wie tief wird die Integration sein? Soll die XMLSpy Benutzeroberfläche, wie sie ist, verwendet werden? Sind Erweiterungen zur Benutzeroberfläche und/oder Einschränkungen erforderlich? Können einige häufig verwendete Aufgaben automatisiert werden?

Die Abschnitte [Integration auf Applikationsebene](#) und [Integration auf Dokumentebene](#), von denen jeder Beispiele in den verschiedenen Programmiersprachen enthält, sollen Ihnen dabei helfen, rasch die richtige Entscheidung zu treffen. Der Abschnitt [Objektreferenz](#) enthält eine Beschreibung aller COM-Objekte mit ihren Eigenschaften und Methoden, die für die Integration verwendet werden können.

Für Automationsaufgaben kann die [XMLSpy Automation-Schnittstelle](#) auch über das XMLSpyControl aufgerufen werden.

Informationen zum Integrieren von XMLSpy in Microsoft Visual Studio finden Sie im Abschnitt [XMLSpy in Visual Studio](#).

4.1 Integration auf Applikationsebene

Die Integration auf Applikationsebene ist einfach. Sie können dadurch die gesamte Benutzeroberfläche von XMLSpy in ein Fenster Ihrer Applikation einbetten. Da Sie die gesamte Benutzeroberfläche von XMLSpy zur Verfügung haben, stehen Ihnen alle Menüs, Symbolleisten, die Statusleiste, die Dokumentfenster und Eingabehilfen zur Verfügung. Die Anpassung der Benutzeroberfläche der Applikation ist auf die Optionen eingeschränkt, die XMLSpy bietet. Dazu gehören die Neuordnung und Anpassung der Größe der Eingabehilfen und die Anpassung von Menüs und Symbolleisten.

Das einzige ActiveX Control, das Sie integrieren müssen, ist [XMLSpyControl](#). Der Wert seiner Eigenschaft [IntegrationLevel](#) ist standardmäßig "application-level". Sie können das Aussehen des Wrapper-Fensters des Control mit Hilfe von [Appearance](#) und [BorderStyle](#) konfigurieren. Bei Integration auf Applikationsebene dürfen [XMLSpyControlDocument](#) oder [XMLSpyControlPlaceholder](#) 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](#) beschriebenen Eigenschaften, Methoden und Events. Um komplexere XMLSpy Funktionsaufrufe auszuführen, sollten Sie eventuell [XMLSpyControl.Application](#) verwenden.

In diesem Abschnitt finden Sie ein Beispiel ([Beispiel: HTML](#)), in dem gezeigt wird, wie die XMLSpy Applikation in eine HTML-Seite eingebettet werden kann. Eine Beschreibung zur Verwendung mit anderen Programmiersprachen oder kompliziertere Aufrufe finden Sie unter den [Beispielen](#) zur Integration auf Dokumentebene.

4.1.1 Beispiel: HTML

In diesem Beispiel wird eine einfache Integration des XMLSpy Control auf Applikationsebene in eine HTML-Seite beschrieben. Die Integration wird in den folgenden Abschnitten beschrieben:

- Instantiieren Sie ein XMLSpyControl in HTML-Code.
- Implementieren Sie Schaltflächen zum Laden von Dokumenten und automatisieren Sie Codegenerierungsaufgaben.
- Definieren Sie Aktionen für einige Applikationsevents.

Den Code für dieses Beispiel finden Sie in Ihrer XMLSpy Installation im folgenden Ordner:
`Examples\ActiveX\HTML\XMLSpyActiveX_ApplicationLevel.htm`

Anmerkung: Dieses Beispiel funktioniert nur in Internet Explorer.

Instantiieren des Control

Der HTML `Object` Tag dient zum Erstellen einer Instanz des XMLSpyControl. Die `Classid` ist die von XMLSpyControl. Mit "width" und "height" wird die Fenstergröße definiert. Es sind keine weiteren Parameter erforderlich, da die Standardeinstellung "application-level" ist.

```
<OBJECT id="objXMLSpyControl"
  Classid="clsid: a258bba2-3835-4c16-8590-72b44f52c471"
  width="1000"
  height="700"
  VIEWASTEXT>
</OBJECT>
```

Hinzufügen einer Schaltfläche zum Öffnen eines Standarddokuments

Als einfaches Beispiel, wie man manche Aufgaben automatisiert, fügen wir eine Schaltfläche zur Seite hinzu:

```
<input type="button" value="Open Marketing Expenses"
onclick="BtnOpenMEFile()">
```

Wenn Sie darauf klicken, wird ein vordefiniertes Dokument im XMLSpyControl geöffnet. Wir verwenden eine Methode, um die Datei relativ zum XMLSpyControl anzugeben, daher kann das Beispiel in unterschiedlichen Installationen ausgeführt werden.

```
<SCRIPT ID=Javahandlers LANGUAGE=javascript>
// -----
// open a pre-defined document
function BtnOpenMEFile()
{
    objXMLSpyControl.Open("C:\Documents and Settings\username\My
Documents\Altova\XMLSpy2012\Examples\OrgChart.xml");
}
</SCRIPT>
```

Hinzufügen von Schaltflächen für die Codegenerierung

Für den direkten Zugriff möchten wir eine Schaltfläche hinzufügen, mit dem das aktuelle Dokument validiert wird. Dazu verwenden wir eine ähnliche Methode wie im vorherigen Abschnitt.

Zuerst kommt die Schaltfläche:

```
<input type="button" value="Validate" onclick="BtnValidate()">
```

Anschließend kommt das Skript, das das aktuelle Dokument validiert.

```
<SCRIPT ID=Javahandlers LANGUAGE=javascript>
// -----
// check validity of current document.
// if validation fails, show validation result in alert box .
function BtnValidate()
{
    // get top-level object of automation interface
    var objApp = objXMLSpyControl.Application;

    // get the active document
    var objDocument = objApp.ActiveDocument;

    if ( objDocument == null )
        alert( "no active document found" );
    else
    {
        // define as arrays to support their usage as return parameters
        var errorText = new Array(1);
        var errorPos = new Array(1);
        var badData = new Array(1);

        var valid = objDocument.IsValid(errorText, errorPos, badData);

        if (! valid)
        {

```

```

        // compose the error description
        var text = errorText;

        // access that XMLData object only if filled in
        if (badData[0] != null)
            text += "(" + badData[0].Name + "/" +
badData[0].TextValue + ")";

        alert("Validation error[" + errorPos + "]: " + text);
    }
    else
        alert("Docuent is valid"); }
}
</SCRIPT>

```

Verbindung zu benutzerdefinierten Events herstellen

In diesem Beispiel werden zur Veranschaulichung des Prinzips zwei Event Callbacks für benutzerdefinierte XMLSpyControl Events implementiert:

```

<!-- ----- -->
<!-- custom event 'OnDocumentOpened' of XMLSpyControl object -->
<SCRIPT LANGUAGE="javascript">
    function objXMLSpyControl::OnDocumentOpened( objDocument )
    {
        // alert("Document '" + objDocument.Name + "' opened!");
    }
</SCRIPT>

<!-- ----- -->
<!-- custom event 'OnDocumentClosed' of XMLSpyControl object -->
<SCRIPT LANGUAGE="javascript">
    function objXMLSpyControl::OnDocumentClosed( objDocument )
    {
        // alert("Document '" + objDocument.Name + "' closed!");
    }
</SCRIPT>

```


4.2 Integration auf Dokumentebene

Bei der Integration auf Dokumentebene haben Sie Freiheit bei der Instantiierung und Platzierung der folgenden Teile der XMLSpy Benutzeroberfläche:

- Bearbeitungsfenster für XMLSpy
- XMLSpy Eingabehilfenfenster
- XMLSpy Validator-Ausgabefenster
- XMLSpy Projektfenster
- XMLSpy XPath Profiler-Fenster
- XMLSpy XPath-Dialogfeld
- XMLSpy XSLT/XQuery Debugger-Fenster
- XMLSpy SOAP Debugger-Fenster

Falls erforderlich, muss in Ihrer Applikation ein Ersatz für die Menüs und Symbolleisten von XMLSpy zur Verfügung gestellt werden.

Je nachdem, welche Teile der Benutzeroberfläche Sie wiederverwenden möchten, müssen Sie mehrere ActiveX Controls instantiieren und aufrufen. Alle diese Controls sind in der XMLSpyControl OCX-Datei enthalten.

- [Mit Hilfe von XMLSpyControl](#) können Sie die Integrationsebene definieren und auf applikationsweite Funktionalitäten zugreifen.
- [Mit Hilfe von XMLSpyControlDocument](#) können Sie beliebig viele Editor-Fenster erstellen. Eventuell genügt es, nur ein Fenster zu erstellen und dieses wiederzuverwenden.
- Verwenden Sie optional [XMLSpyControlPlaceholder](#) um XMLSpy Eingabehilfenfenster, Validierungsausgabefenster oder andere oben erwähnte Fenster einzubetten.
- Rufen Sie Laufzeitinformationen über in XMLSpyControl verfügbare Befehle, Menüs und Symbolleisten auf, um diese Befehle nahtlos in Ihre Applikationsmenüs und Symbolleisten zu integrieren. Nähere Informationen dazu finden Sie unter [Abfrage von XMLSpy Befehlen](#).

Wenn Sie ein bestimmtes Verhalten von XMLSpy automatisieren möchten, verwenden Sie die für [XMLSpyControl](#), [XMLSpyControlDocument](#) und [XMLSpyControlPlaceholder](#) beschriebenen Eigenschaften, Methoden und Events. Bei Aufruf komplexerer XMLSpy Funktionalitäten sollten Sie eventuell [XMLSpyControl.Application](#), [XMLSpyControlDocument.Document](#) und [XMLSpyControlPlaceholder.Project](#) verwenden. Verwenden Sie allerdings zum Öffnen eines Dokuments immer [XMLSpyControlDocument.Open](#) oder [XMLSpyControlDocument.New](#) am entsprechenden Document Control. Verwenden Sie zum Öffnen eines Projekts immer [XMLSpyControlPlaceholder.OpenProject](#) an einem Placeholder Control, das ein XMLSpy Projektfenster einbettet.

Beschreibungen zum Instantiieren und Aufrufen der erforderlichen Controls in verschiedenen Programmierungsumgebungen finden Sie unter [Beispiele](#).

4.2.1 Verwendung von XMLSpyControl

Zur Integration auf Dokumentebene instantiieren Sie zuerst ein [XMLSpyControl](#). Setzen Sie die Eigenschaft [IntegrationLevel](#) auf `ICActiveXIntegrationOnDocumentLevel(=1)`. Setzen Sie die Fenstergröße des einbettenden Fensters auf `0x0`, damit eine andere Benutzeroberfläche gegebenenfalls vom Control verdeckt wird. Sie können das Aussehen des Wrapper-Fensters für das Control mittels [Appearance](#) und [BorderStyle](#) konfigurieren.

Vermeiden Sie nach Möglichkeit die Verwendung der Methode [Open](#), da dies zu unerwarteten Ergebnissen führen könnte. Verwenden Sie stattdessen die entsprechenden

Öffnungsmethoden von [XMLSpyControlDocument](#) und [XMLSpyControlPlaceholder](#).

Eine Beschreibung zur Integration von XMLSpy Befehlen in Ihre Applikation finden Sie unter [Abfragen von XMLSpy Befehlen](#). Senden Sie Befehle über die Methode [Exec](#) an XMLSpy. Führen Sie mit Hilfe der Methode [QueryStatus](#) eine Abfrage durch, um zu sehen, ob ein Befehl derzeit aktiviert oder deaktiviert ist.

4.2.2 Verwendung von XMLSpyControlDocument

Mit Hilfe einer Instanz des `XMLSpyControlDocument` ActiveX Control können Sie ein XMLSpy Mapping-Bearbeitungsfenster in Ihre Applikation einbetten. Sie können so viele Instanzen verwenden, so viele Sie benötigen.

Verwenden Sie die Methode [Open](#), um eine beliebige andere bereits vorhandene Datei zu laden.

Das Control unterstützt keinen schreibgeschützten Modus. Der Wert der Eigenschaft [ReadOnly](#) wird ignoriert.

Verwenden Sie zum Aufrufen von Dokumentfunktionalitäten [Path](#) und [Save](#) oder Methoden und Eigenschaften, die über die Eigenschaft [Document](#) aufgerufen werden.

4.2.3 Verwendung von XMLSpyControlPlaceholder

Mit Hilfe von Instanzen des `XMLSpyControlPlaceholder` ActiveX Control können Sie selektiv weitere Eingabehilfefenster von XMLSpy in Ihre Applikation einbetten. Die Eigenschaft [PlaceholderWindowID](#) wählt das XMLSpy Eingabehilfefenster aus, um es einzubetten. Verwenden Sie für jeden Fenster-Identifizier nur einen `XMLSpyControlPlaceholder`. Eine Beschreibung gültiger Fenster-Identifizier finden Sie unter [Enumerations.XMLSpyControlPlaceholderWindow](#).

Für Platzhalter-Controls, die das XMLSpy Projektfenster auswählen, stehen zusätzliche Methoden zur Verfügung. Verwenden Sie [OpenProject](#), um ein XMLSpy Projekt zu laden. Verwenden Sie die Eigenschaft [Project](#) und die Methoden und Eigenschaften aus dem XMLSpy Automation Interface, um andere projektbezogene Operationen auszuführen.

4.2.4 Abfragen von XMLSpy Befehlen

Bei der Integration auf Dokumentebene stehen Ihrer Applikation kein Menü und keine Symbolleiste aus XMLSpy zur Verfügung. Stattdessen können Sie alle Befehle und die Struktur des Applikationsmenüs zur Laufzeit abfragen. Bei professionellen Applikationen muss dieses Menü auf raffinierte Art in die eigene Menüstruktur integriert werden. In Ihrer Installation von XMLSpy stehen Ihnen sogar Bilder für in XMLSpy verwendete Befehle zur Verfügung. Symbole im GIF-Format finden Sie in Ihrem XMLSpy Installationsordner unter `Examples\ActiveX\Images` of your XMLSpy. Die Dateinamen entsprechen den [labels](#) von Befehlen.

4.2.5 Beispiele

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 `Examples\ActiveX` Ihrer XMLSpy Installation zur Verfügung.

C#

Im C#-Beispiel wird gezeigt, wie man das XMLSpyControl in eine allgemeine Desktop-Applikation integriert, die mit C# über Visual Studio 2008 erstellt wurde. Es werden die folgenden Themen behandelt:

- Integration eines XMLSpyControl Document Control zum Einbetten eines XMLSpy Dokumentbearbeitungsfensters.
- Verwendung eines XMLSpyControlPlaceholder Control zum Einbetten des XMLSpy XPath Dialogfensters.
- Öffnen und Schließen eines XMLSpy Dokuments über das Hauptmenü.
-

Bitte beachten Sie, dass die Beispielapplikation bereits fertig ist. Sie müssen keine Änderungen vornehmen, wenn Sie sie ausführen und sehen möchten, wie sie funktioniert. In den folgenden Schritten wird beschrieben, welche Aktionen durchgeführt und welche Dinge berücksichtigt werden müssen, um ein Projekt wie dieses zu erstellen.

Einführung

Hinzufügen der XMLSpy -Komponenten zur Toolbox

Bevor Sie einen Blick in das Beispielprojekt werfen, fügen Sie bitte die Assemblies zur .NET IDE Toolbox hinzu. Der XMLSpy Installer hat die Assemblies bereits im .NET Global Assembly Cache (GAC) installiert. Wenn Sie das Toolbox-Dialogfeld unter **Tools | Add/Remove Toolbox Items** öffnen, werden die Controls als `AxXMLSpyControl`, `AxXMLSpyControlDocument` und `AxXMLSpyControlPlaceholder` auf dem Register ".NET Framework Components" angezeigt. Aktivieren Sie alle, damit Sie in der IDE zur Verfügung stehen.

Nun können Sie die Datei `XPathDialog.sln` im Ordner `ActiveX\C#\XPathDialog` öffnen, um das Projekt zu laden.

Platzieren des XMLSpyControl

Sie benötigen eine XMLSpyControl Instanz, um die Integrationsebene zu definieren und die Dokument- und Platzhalter-Controls der XMLSpy Bibliothek zu verwalten. Das Control kann über den Abschnitt "General" des Toolbox-Eingabehilfenfensters in der IDE aufgerufen werden. Um es hinzuzufügen, müssen Sie die Komponente im Fenster "Toolbox" auswählen und ein Rechteck an der Stelle aufziehen, an der Sie es im Zielfenster haben möchten. Bei einer Applikation, die beim Start kein Fenster öffnet, können Sie ein einfaches unsichtbares Formular mit dem Control verwenden, an dem es im Code manuell erstellt wird.

Im Beispielprojekt wird diese Instanz zum Haupt-MdiContainer MDIMain hinzugefügt. Wenn Sie MDIMain in der Design-Ansicht über den Solution Explorer öffnen, sehen Sie im Client-Bereich des Frame-Fensters an der linken oberen Seite ein hellblaues Rechteck. Wenn Sie das Rechteck auswählen, sehen Sie die Eigenschaften des XMLSpyControl. Die Eigenschaft `IntegrationLevel` muss unbedingt auf `ICActiveXIntegrationOnDocumentLevel` gesetzt werden, um die Unterstützung der XMLSpy Bibliothek für Dokumente und Platzhalter zu aktivieren. Um Verwirrung zu vermeiden, können Sie das Sichtbarkeitsflag auf "False" setzen.

Hinzufügen des Document Control

Dokumentbearbeitungsfenster im MDI-Frame

Im Beispielprojekt wird ein Document Control verwendet um ein XMLSpy Dokument im

MDI-Hauptframe anzuzeigen und zu bearbeiten. Es wird über das Toolbox-Fenster durch Aufziehen eines Rechtecks am Zielformular hinzugefügt. Für das Document Control wurden auch die Eigenschaften "Anchor" und "Dock" gesetzt, um auf die Anpassung der Größe des Frame-Fensters zu reagieren.

Hinzufügen des Placeholder Control

Placeholders im MDI-Frame

Im Beispielprojekt wird ein Placeholder Control verwendet, um das XPath-Dialogfeldfenster in den MDI-Haupt-Frame einzubetten. Es wird über das Toolbox-Fenster durch Aufziehen eines Rechtecks am Zielformular hinzugefügt. Zum Setzen des Typs des anzuzeigenden Placeholder, müssen Sie die Eigenschaft `PlaceholderWindowID` definieren. Diese Eigenschaft kann auch zur Laufzeit im Code der Applikation geändert werden. Das Placeholder Control würde seinen Inhalt sofort ändern. Für den Placeholder wurden auch die Anchor-Eigenschaften definiert, um auf eine Anpassung der Größe des Frame-Fensters zu reagieren.

Abrufen von Befehlsinformationen

Über das XMLSpyControl haben Sie über die Eigenschaft `CommandsStructure` Zugriff auf alle Befehle von XMLSpy. Im Beispielprojekt werden die Schnittstellen [XMLSpyCommands](#) und [XMLSpyCommand](#) verwendet, um im MDI-Frame-Fenster dynamisch ein Menü zu erstellen.

Der Code zum Hinzufügen der Befehle wird in die Datei `MDIMain.cs` in die `MDIMain` Methode der `XMLSpyApplication` Klasse platziert:

```
public MDIMain()
{
    .
    .
    .
    XMLSpyControlLib.XMLSpyCommands objCommands;
    objCommands = axXMLSpyControl.CommandsStructure;

    long nCount = objCommands.Count;

    for(long idx = 0; idx < nCount; idx++)
    {
        XMLSpyControlLib.XMLSpyCommand objCommand;
        objCommand = objCommands[(int)idx];

        // We are looking for the Menu with the name IDR_XMLSPY. This menu
        // contains
        // the complete main menu of XMLSpy.

        if(objCommand.Label == "IDR_XMLSPY")
        {
            InsertMenuStructure(mainMenu.MenuItems, 1, objCommand, 0, 0,
                               false);
        }
    }
    .
    .
    .
}
```

`mainMenu` ist der Name des Menüobjekts des in der Visual Studio IDE erstellten MDI Frame-Fensters. `InsertMenuStructure` nimmt das XMLSpy Menü aus dem `IDR_XMLSPY` Befehlsobjekt und fügt die XMLSpy Menüstruktur zum bestehenden Menü des Beispielprojekts

hinzu. Es werden keine Befehle aus den Menüs **Datei**, **Projekt** oder **Fenster** hinzugefügt.

Die neuen Befehle sind Instanzen der Klasse `CustomMenuItem`, die in `CustomMenuItem.cs` definiert ist. Diese Klasse hat ein zusätzliches Member zum Speichern der XMLSpy Befehls-ID, die verwendet wird, um den Befehl bei Auswahl des Menübefehls mit [Exec](#) auszuführen. Dieser Code aus `InsertMenuStructure` erstellt den neuen Befehl:

```
CustomMenuItem newMenuItem = new CustomMenuItem();

if(objCommand.IsSeparator)
    newMenuItem.Text = "-";
else
{
    newMenuItem.Text = strLabel;
    newMenuItem.m_XMLSpyCmdID = (int)objCommand.ID;
    newMenuItem.Click += new EventHandler(AltovaMenuItem_Click);
}
```

You can see that all commands get the same event handler `AltovaMenuItem_Click` which does the processing of the command:

```
private void AltovaMenuItem_Click(object sender, EventArgs e)
{
    if(sender.GetType() == System.Type.GetType("XMLSpy
Application.CustomMenuItem"))
    {
        CustomMenuItem customItem = (CustomMenuItem) sender;

        ProcessCommand(customItem.m_XMLSpyCmdID);
    }
}

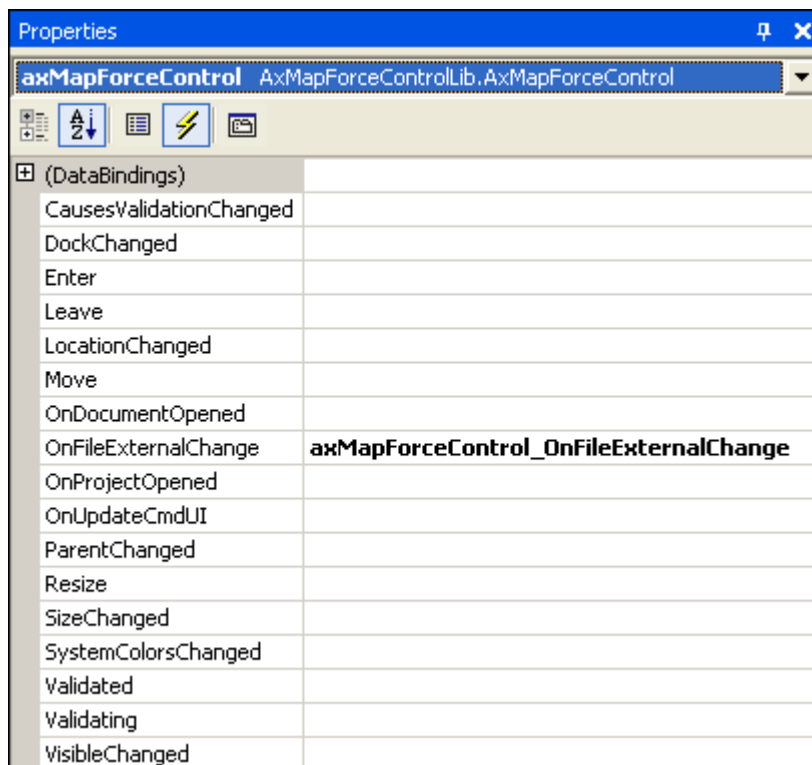
private void ProcessCommand(int nID)
{
    XMLSpyDoc docXMLSpy = GetCurrentXMLSpyDoc();

    if(docXMLSpy != null)
        docXMLSpy.axXMLSpyControlDoc.Exec(nID);
    else
        axXMLSpyControl.Exec(nID);
}
```

`ProcessCommand` delegiert die Ausführung entweder an das XMLSpyControl selbst oder an ein aktives XMLSpy Dokument, das in einem `XMLSpyControlDocument` Control geladen ist. Dies ist notwendig, da das XMLSpyControl nicht wissen kann, welches Dokument gerade in der Host-Applikation aktiv ist.

Behandlung von Events

Da alle Events in der XMLSpy Bibliothek auf Connection Points basieren, können Sie mit Hilfe der C# Delegatsfunktionalität benutzerdefinierte Event Handler bereitstellen. Auf der Seite "Eigenschaften" der einzelnen Controls der XMLSpy Bibliothek finden Sie jeweils eine vollständige Liste von Events. In der Abbildung unten sehen Sie die Events des Haupt-XMLSpyControl:



Wie Sie sehen, wird in dem Beispiel nur das `OnFileExternalChange` Event überschrieben. Die Erstellung des C#-Delegaten wird vom C# Framework erledigt. Sie müssen nur den leeren Event Handler ausfüllen. Die Handler-Implementierung deaktiviert das Neuladen von Dateien und zeigt ein Meldungsfeld an, um den Benutzer zu informieren, dass eine vom XMLSpyControl geladene Datei von außerhalb geändert wurde:

```
private void axMapForceControl_OnFileExternalChange(object sender,
AxMapForceControlLib._DMapForceControlEvents_OnFileExternalChangeEvent e)
{
    MessageBox.Show("Attention: The file " + e.strPath + " has been changed
from outside\nbut reloading is turned off in the sample application!");

    // This turns off any file reloading:
    e.varRet = false;
}
```

Testen des Beispiels

Nachdem Sie die Assemblies zur Toolbox hinzugefügt haben (siehe [Einführung](#)), können Sie das Beispielprojekt mit F5 ausführen, ohne irgendwelche Änderungen im Code vornehmen zu müssen. Das MDI-Haupt-Frame-Fenster wird erstellt und enthält ein Bearbeitungsfenster mit einem leeren XML-Dokument und einem XPath-Dialogfeldfenster von XMLSpy am unteren Rand des Fensters. Mit Hilfe von **Datei | Öffnen** können Sie die Datei `OrgChart.xml`, die sich im XMLSpy Beispielordner befindet, öffnen. Die Datei wird geladen und angezeigt.

Nachdem Sie das Dokument geladen haben, können Sie versuchen, das XPath-Dialogfeld zu verwenden.

HTML

In diesem Beispiel wird eine Integration des XMLSpy Control auf Dokumentenebene in eine HTML-Seite gezeigt. Es werden die folgenden Themen behandelt:

- Instantiieren eines XMLSpyControl ActiveX Control-Objekts in HTML-Code
- Instantiieren eines XMLSpyControlDocument ActiveX Control für die Bearbeitung eines XMLSpy Mappings
- Instantiieren eines XMLSpyControlPlaceholder für ein XMLSpy-Projektfenster
- Instantiieren eines XMLSpyControlPlaceholder ActiveX Control für die Aufnahme eines der XMLSpy-Eingabehilfefenster
- Instantiieren eines XMLSpyControlPlaceholder ActiveX Control zur Anzeige des XMLSpy-Fensters "Validierungsausgabe"
- Erstellen einer benutzerdefinierten Symbolleiste für einige häufig gebrauchte XMLSpy-Befehle
- Hinzufügen von weiteren Schaltflächen, die das COM Automation Interface von XMLSpy verwenden
- Verwenden von Event Handlern zum Aktualisieren von Befehlsschaltflächen

Dieses Beispiel ist vollständig in der Datei XMLSpyActiveX_ApplicationLevel.htm im Ordner C:\Program Files\Altova\XMLSpy2012\Examples\ActiveX\HTML\ Ihrer XMLSpy Installation enthalten.

Anmerkung: Dieses Beispiel funktioniert nur in Internet Explorer.

Instantiieren des XMLSpyControl

Zum Erstellen einer Instanz des XMLSpyControl wird der HTML OBJECT Tag verwendet. Die Classid ist die von XMLSpyControl. "width" und "height" werden auf 0 gesetzt, da wir dieses Control als Manager Control ohne Verwendung für seine Benutzeroberfläche verwenden. Die Integrationsebene wird in Form eines Parameters im OBJECT Tag definiert.

```
<OBJECT id="objXMLSpyXXMLSpyControl"
        Classid="clsid: a258bba2-3835-4c16-8590-72b44f52c471"
        width="0"
        height="0"
        VIEWASTEXT>
    <PARAM NAME="IntegrationLevel" VALUE="1">
</OBJECT>
```

Erstellen eines Bearbeitungsfensters

Der HTML OBJECT Tag wird zum Einbetten eines Dokumentenbearbeitungsfensters verwendet.

```
<OBJECT id="objDoc1"
        Classid="clsid: DFBB0871-DAFE-4502-BB66-08CEB7DF5255"
        width="600"
        height="500"
        VIEWASTEXT>
</OBJECT>
```

Erstellen eines Projektfensters

Der HTML `OBJECT` Tag wird zum Erstellen eines `XMLSpyControlPlaceholder`-Fensters verwendet. Der erste zusätzliche benutzerdefinierte Parameter definiert, welcher Platzhalter das XMLSpy-Projektfenster anzeigen soll. Der zweite Parameter lädt eines der Beispielprojekte aus Ihrer XMLSpy Installation (aus dem Ordner `<yourusername>/MyDocuments`).

```
<OBJECT id="objProjectWindow"
  Classid="clsid: FDEC3B04-05F2-427d-988C-F03A85DE53C2"
  width="200"
  height="200"
  VIEWASTEXT>
  <PARAM name="PlaceholderWindowID" value="3">
  <PARAM name="FileName" value="Examples/Examples.spp">
</OBJECT>
```

Erstellen eines Placeholder für Eingabehilfenfenster

Der HTML `OBJECT` Tag wird zum Instantiieren eines `XMLSpyControlPlaceholder` ActiveX Control verwendet, das die verschiedenen XMLSpy Hilfsfenster aufnehmen kann. Anfangs wird kein Hilfsfenster angezeigt. Siehe Beispieldatei:

```
<OBJECT id="objEHWindow"
  Classid="clsid: 135DEEF4-6DF0-47c2-8F8C-F145F5F3F672"
  width="200"
  height="200"
  VIEWASTEXT>
  <PARAM name="PlaceholderWindowID" value="0">
</OBJECT>
```

Mit Hilfe von drei Schaltflächen kann zwischen dem tatsächlichen Fenster, das angezeigt werden soll, gewechselt werden. Mit der JavaScript-Anweisung die bei Klick auf eine Schaltfläche ausgeführt wird, wird die Eigenschaft `PlaceholderWindowID` auf den entsprechenden in [XMLSpyControlPlaceholderWindow](#) definierten Wert gesetzt.

```
<input type="button" value="EH - Elements" onclick="BtnHelperWindow(0)">
<input type="button" value="EH - Attributes" onclick="BtnHelperWindow(1)">
<input type="button" value="EH - Entities" onclick="BtnHelperWindow(2)">
```

```
<SCRIPT ID="Javahandlers" LANGUAGE="javascript">
//
-----
// specify which of the windows shall be shown in the placeholder control.
function BtnHelperWindow(i_ePlaceholderWindowID)
{
  objEHWindow.PlaceholderWindowID = i_ePlaceholderWindowID;
}
</SCRIPT>
```

Erstellen einer benutzerdefinierten Symbolleiste

Die benutzerdefinierte Symbolleiste besteht aus Schaltflächen mit Symbolen für XMLSpy Befehle. Sie finden die Befehls-ID-Nummern in den Schaltflächen-Elementen unter [Befehlstabelle](#)

```
<button id="btnWellFormed" title="Check Well-formedness"
onclick="BtnDoCommand( 34049) ">
```



```

    
</button>
<button id="btnValidate" title="Validate" onclick="BtnDoCommand( 34174) ">
    
</button>

```

Wenn Sie auf eine dieser Schaltflächen klicken, wird die entsprechende Befehls-ID an das Manager Control gesendet.

```

<SCRIPT ID="Javahandlers" LANGUAGE="javascript">
// -----
// perform any command specified by cmdID.
// command routing includes application, active document and view.
function BtnDoCommand( cmdID)
{
    objXMLSpyX.Exec( cmdID );
    msgtxt.innerText = "Command " + cmdID + " performed.";
}
</SCRIPT>

```

Erstellen weiterer Schaltflächen

In unserem Beispiel fügen wir einige weitere Schaltflächen hinzu, um Beispiele für Automatisierungscode zu zeigen.

```

<p>
    <input type="button" value="New File" onclick="BtnNewFile( objDoc1) ">
    <input type="button" value="Save File" onclick="BtnSaveFile( objDoc1) ">
    <input type="text" title="Path" id="strPath" width="150">
    <input type="button" value="Open OrgChart.xml"
onclick="BtnOpenFile( objDoc1, ' OrgChart.xml') ">
    <input type="button" value="Open OrgChart.xsd"
onclick="BtnOpenFile( objDoc1, ' OrgChart.xsd') ">
</p>

```

Das entsprechende JavaScript sieht folgendermaßen aus:

```

<SCRIPT ID="Javahandlers" LANGUAGE="javascript">
// -----
// open a document in the specified document control window.
function BtnOpenFile( objDocCtrl, strFileName)
{
    // do not use XMLSpyX.Application.OpenDocument(...) to open a document,
    // since then XMLSpyControl wouldn't know a control window to show
    // the document in. Instead:

    objDocCtrl.Open( "C:\Documents and Settings\username\My
Documents\Altova\XMLSpy2012\Examples/" + strFileName);
    objDocCtrl.setActive();
}

// -----
// open a new empty document in the specified document control window.
function BtnNewFile( objDocCtrl)
{
    objDocCtrl.Open( "" );
    objDocCtrl.setActive();
}

// -----
// Saves the current file in the specified document control window.
function BtnSaveFile( objDocCtrl)

```

```

{
    if(objDocCtrl.Path.length > 0)
        objDocCtrl.SaveDocument();
    else
    {
        if(strPath.value.length > 0)
        {
            objDocCtrl.Path = strPath.value;
            objDocCtrl.Save();
        }
        else
        {
            alert("Please set path for the document first!");
            strPath.focus();
        }
    }

    objDocCtrl.setActive();
}

// -----
// check validity of current document.
// if validation fails, show validation result in alert box .
function BtnValidate()
{
    // get top-level object of automation interface
    var objApp = objXMLSpyControl.Application;

    // get the active document
    var objDocument = objApp.ActiveDocument;

    if ( objDocument == null )
        alert( "no active document found" );
    else
    {
        // define as arrays to support their usage as return parameters
        var errorText = new Array(1);
        var errorPos = new Array(1);
        var badData = new Array(1);

        var valid = objDocument.IsValid(errorText, errorPos, badData);

        if (! valid)
        {
            // compose the error description
            var text = errorText;

            // access that XMLData object only if filled in
            if (badData[0] != null)
                text += "(" + badData[0].Name + "/" +
badData[0].TextValue + ")";

            alert("Validation error[" + errorPos + "]: " + text);
        }
        else
            alert("Docuent is valid");
    }
}
</SCRIPT>

```

Erstellen eines Event Handler zum Aktualisieren des Schaltflächenstatus

Die Verfügbarkeit eines Befehls kann je nach Mausklick oder Tastendruck variieren. Das benutzerdefinierte Event `OnUpdateCmdUI` von `XMLSpyControl` gibt uns die Gelegenheit, den Schaltflächenstatus "aktiviert/deaktiviert", der mit XMLSpy Befehlen verknüpft ist, zu aktualisieren. Um abzufragen, ob ein Befehl aktiv oder deaktiviert ist, wird die Methode [`XMLSpyControl.QueryStatus`](#) verwendet.

```
<SCRIPT LANGUAGE="javascript">

    function objXMLSpyX: OnUpdateCmdUI()
    {
        if ( document.readyState == "complete" )           // 'complete'
        {
            // update status of buttons
            btnWellFormed.disabled = ! (objDoc1.QueryStatus(34049) & 0x02);
            // not enabled
            btnValidate.disabled = ! (objDoc1.QueryStatus(34174) & 0x02);
            // not enabled
        }
    }

    // set activity status of simulated toolbar
</SCRIPT>
```

4.3 Befehlstabelle für XMLSpy

In den Tabellen in diesem Abschnitt werden die Namen und Identifier aller Befehle aufgelistet, die in XMLSpy zur Verfügung stehen. In den einzelnen Unterabschnitten werden jeweils die Befehle aus dem entsprechenden Menü von XMLSpy aufgelistet. In der Spalte ganz links wird der Text des Menübefehls angezeigt, damit Sie sofort wissen, um welche Funktionalität es sich hierbei handelt. Im letzten Unterabschnitt finden Sie all jene Befehle, die über das Hauptmenü nicht aufgerufen werden können.

Einige dieser Befehle werden je nach installierter XMLSpy Edition eventuell nicht unterstützt. Eine Beschreibung, wie man die aktuelle Ressourcenstruktur und die Verfügbarkeit von Befehlen abfragt, finden Sie unter [Abfragen von XMLSpy Befehlen](#).

Verwenden Sie die Befehls-Identifier mit [XMLSpyControl.QueryStatus](#) oder [XMLSpyControlDocument.QueryStatus](#) um den aktuellen Status eines Befehls zu überprüfen. Verwenden Sie [XMLSpyControl.Exec](#) oder [XMLSpyControlDocument.Exec](#) um einen Befehl auszuführen.

[Menü "Datei"](#)
[Menü "Bearbeiten"](#)
[Menü "Projekt"](#)
[Menü "XML"](#)
[Menü "DTD/Schema"](#)
[Menü "Schema-Design"](#)
[Menü "XSL/XQuery"](#)
[Menü "Authentic"](#)
[Menü "Konvertieren"](#)
[Menü "Ansicht"](#)
[Menü "Browser"](#)
[Menü "WSDL"](#)
[Menü "SOAP"](#)
[Menü "Extras"](#)
[Menü "Fenster"](#)
[Menü "Hilfe"](#)

4.3.1 Menü "Datei"

Befehle aus dem Menü "Datei":

Menütext	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
Alles schließen	IDC_CLOSE_ALL	34050
Speichern	ID_FILE_SAVE	57603
Speichern unter...	ID_FILE_SAVE_AS	57604
Speichern unter URL...	ID_FILE_SAVE_TO_URL	34209
Alles speichern	ID_FILE_SAVE_ALL	34208
Als Mail senden...	ID_FILE_SEND_MAIL	57612
Drucken...	IDC_PRINT	34103

Druckvorschau	IDC_PRINT_PREVIEW	34104
Druckereinrichtung...	ID_FILE_PRINT_SETUP	57606
Letzte Datei	ID_FILE_MRU_FILE1	57616
Beenden	ID_APP_EXIT	57665

4.3.2 Menü "Bearbeiten"

Befehle aus dem Menü "Bearbeiten":

Menütext	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
Pretty-Print	IDC_PRETTY_PRINT	34101
Alles markieren	ID_EDIT_SELECT_ALL	57642
Suchen...	ID_EDIT_FIND	57636
Weitersuchen	ID_EDIT_REPEAT	57640
Ersetzen...	ID_EDIT_REPLACE	57641
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

4.3.3 Menü "Projekt"

Befehle aus dem Menü "Projekt":

Menütext	Befehlsname	ID
Neues Projekt	IDC_PROJECT_NEW	34122
Projekt öffnen...	IDC_PROJECT_OPEN	34123
Projekt neu laden	IDC_PROJECT_RELOAD	34126
Projekt schließen	IDC_PROJECT_CLOSE	34113
Projekt speichern	IDC_PROJECT_SAVE	34127

Versionskontrolle/Projekt öffnen...	IDC_SCC_OPEN_PROJECT	34140
Versionskontrolle/Versionskontrolle aktivieren	IDC_SCC_ENABLE	34137
Versionskontrolle/Aktuellste Version holen...	IDC_SCC_GET	34138
Versionskontrolle/Auschecken...	IDC_SCC_CHECK_OUT	34135
Versionskontrolle/Einchecken...	IDC_SCC_CHECK_IN	34134
Versionskontrolle/Auschecken rückgängig...	IDC_SCC_UNDO_CHECK_OUT	34145
Versionskontrolle/Zu Versionskontrolle hinzufügen...	IDC_SCC_ADD	34133
Versionskontrolle/Von Versionskontrolle ausgliedern...	IDC_SCC_REMOVE	34143
Versionskontrolle/Verlauf anzeigen...	IDC_SCC_HISTORY	34139
Versionskontrolle/Unterschiede anzeigen...	IDC_SCC_DIFF	34136
Versionskontrolle/Eigenschaften...	IDC_SCC_PROPERTIES	34141
Versionskontrolle/Status aktualisieren...	IDC_SCC_REFRESH	34142
Versionskontrolle/Native Oberfläche ausführen...	IDC_SCC_RUN	34144
Dateien zu Projekt hinzufügen...	IDC_PROJECT_ADD_FILES	34109
URL zu Projekt hinzufügen...	IDC_PROJECT_ADD_URL	34112
Aktive Datei zu Projekt hinzufügen	IDC_PROJECT_ADD_ACTIVE	34106
Aktive und verwandte Dateien zu Projekt hinzufügen	IDC_PROJECT_ADD_RELATED	34111
Projektordner zu Projekt hinzufügen...	IDC_PROJECT_ADD_FOLDER	34110
Externes Verzeichnis zum Projekt hinzufügen...	IDC_PROJECT_ADD_EXT_FOLDER	34107
Externes Webverzeichnis zum Projekt hinzufügen..	IDC_PROJECT_ADD_EXT_URL_FOLDER	34108
Projekteigenschaften...	IDC_PROJECT_PROPERTIES	34124
Letztes Projekt	IDC_RECENT_PROJECT_ID	34265

4.3.4 Menü "XML"

Befehle aus dem Menü "XML":

Menütext	Befehlsname	ID
Einfügen/Attribute	IDC_INSERT_ATTRIBUTE	33449
Einfügen/Element	IDC_INSERT_STRUCT	33459
Einfügen/Text	IDC_INSERT_TEXT	33460
Einfügen/CDATA	IDC_INSERT_CDATA	33450
Einfügen/Comment	IDC_INSERT_COMMENT	33451
Einfügen/XML	IDC_INSERT_XML	33461
Einfügen/Processing Instruction	IDC_INSERT_PI	33458
Einfügen/DOCTYPE	IDC_INSERT_DEF_DOCTYPE	33453
Einfügen/ExternalID	IDC_INSERT_DEF_EXTERNAL_ID	33456
Einfügen/ELEMENT	IDC_INSERT_DEF_ELEMENT	33454
Einfügen/ATTLIST	IDC_INSERT_DEF_ATTLIST	33452

Einfügen/ENTITY	IDC_INSERT_DEF_ENTITY	33455
Einfügen/NOTATION	IDC_INSERT_DEF_NOTATION	33457
Anhängen/Attribute	IDC_APPEND_ATTRIBUTE	33415
Anhängen/Element	IDC_APPEND_STRUCT	33425
Anhängen/Text	IDC_APPEND_TEXT	33426
Anhängen/CData	IDC_APPEND_CDATA	33416
Anhängen/Comment	IDC_APPEND_COMMENT	33417
Anhängen/XML	IDC_APPEND_XML	33427
Anhängen/Processing Instruction	IDC_APPEND_PI	33424
Anhängen/DOCTYPE	IDC_APPEND_DEF_DOCTYPE	33419
Anhängen/ExternalID	IDC_APPEND_DEF_EXTERNAL_ID	33422
Anhängen/ELEMENT	IDC_APPEND_DEF_ELEMENT	33420
Anhängen/ATTLIST	IDC_APPEND_DEF_ATTLIST	33418
Anhängen/ENTITY	IDC_APPEND_DEF_ENTITY	33421
Anhängen/NOTATION	IDC_APPEND_DEF_NOTATION	33423
Child hinzufügen/Attribute	IDC_ADD_CHILD_ATTRIBUTE	33402
Child hinzufügen/Element	IDC_ADD_CHILD_STRUCT	33412
Child hinzufügen/Text	IDC_ADD_CHILD_TEXT	33413
Child hinzufügen/CData	IDC_ADD_CHILD_CDATA	33403
Child hinzufügen/Comment	IDC_ADD_CHILD_COMMENT	33404
Child hinzufügen/XML	IDC_ADD_CHILD_XML	33414
Child hinzufügen/Processing Instruction	IDC_ADD_CHILD_PI	33411
Child hinzufügen/DOCTYPE	IDC_ADD_CHILD_DEF_DOCTYPE	33406
Child hinzufügen/ExternalID	IDC_ADD_CHILD_DEF_EXTERNAL_ID	33409
Child hinzufügen/ELEMENT	IDC_ADD_CHILD_DEF_ELEMENT	33407
Child hinzufügen/ATTLIST	IDC_ADD_CHILD_DEF_ATTLIST	33405
Child hinzufügen/ENTITY	IDC_ADD_CHILD_DEF_ENTITY	33408
Child hinzufügen/NOTATION	IDC_ADD_CHILD_DEF_NOTATION	33410
Konvertieren in/Attribute	IDC_CONVERT_TO_ATTRIBUTE	33429
Konvertieren in/Element	IDC_CONVERT_TO_STRUCT	33439
Konvertieren in/Text	IDC_CONVERT_TO_TEXT	33440
Konvertieren in/CData	IDC_CONVERT_TO_CDATA	33430
Konvertieren in/Comment	IDC_CONVERT_TO_COMMENT	33431
Konvertieren in/XML	IDC_CONVERT_TO_XML	33441
Konvertieren in/Processing Instruction	IDC_CONVERT_TO_PI	33438
Konvertieren in/DOCTYPE	IDC_CONVERT_TO_DEF_DOCTYPE	33433
Konvertieren in/ExternalID	IDC_CONVERT_TO_DEF_EXTERNAL_ID	33436
Konvertieren in/ELEMENT	IDC_CONVERT_TO_DEF_ELEMENT	33434
Konvertieren in/ATTLIST	IDC_CONVERT_TO_DEF_ATTLIST	33432
Konvertieren in/ENTITY	IDC_CONVERT_TO_DEF_ENTITY	33435

Konvertieren in/NOTATION	IDC_CONVERT_TO_DEF_NOTATION	33437
Tabelle/Als Tabelle anzeigen	IDC_GRID_VIEW_AS_TABLE	34075
Tabelle/Zeile einfügen	IDC_TABLE_INSERT_ROW	34158
Tabelle/Zeile anhängen	IDC_TABLE_APPEND_ROW	34157
Tabelle/Aufsteigend sortieren	IDC_TABLE_SORT_ASC	33464
Tabelle/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
Validieren	IDC_VALIDATE	34174
Eingabehilfen aktualisieren	IDC_UPDATE_ELEMENT_CHOICE	34173
Namespace-Präfix...	IDC_NAMESPACE	33462

4.3.5 Menü "DTD/Schema"

Befehle aus dem Menü "DTD/Schema":

Menütext	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/Schema konvertieren...	IDC_CONVERT_DTD_SCHEMA	34052
In UML konvertieren...	IDC_CONVERT_SCHEMA_TO_UML	34008
Im MapForce auf andere(s) DTD/Schema oder DB mappen...	IDC_DTD_OPENIN_MAPFORCE	34056
HTML/PDF-Ausgabe in StyleVision erstellen...	IDC_DTD_OPENIN_STYLEVISION	34057
XML-Beispieldatei erzeugen...	IDC_GENERATE_XML_FROM_SCHEMA	34069
Programmcode generieren...	IDC_GENERATE_CODE_FROM_SCHEMA	34067
Cash-Speicher leeren	IDC_FLUSH_CACHED_FILES	34066

4.3.6 Menü "Schema-Design"

Befehle aus dem Menü "Schema-Design":

Menütext	Befehlsname	ID
Schema-Einstellungen	IDC_SCHEMA_NAMESPACES	3357 1

Diagramm speichern...	IDC_SCHEMA_SAVE_DIAGRAM	3358 1
Dokumentation generieren	IDC_SCHEMA_DOCUMENTATION	3414 6
Ansicht konfigurieren	IDC_SCHEMA_VIEW_CONFIG	3359 3
Vergrößern/Verkleinern	IDC_SCHEMA_ZOOM	3415 0
Globale Elemente anzeigen	IDC_SCHEMA_MODE_GLOBALS	3414 7
Diagramm anzeigen	IDC_SCHEMA_MODE_DIAGRAM	3357 0
Oracle Schema Extensions aktivieren	IDC_SCHEMA_ORACLE_EXTENSIONS	3357 7
Oracle Schema-Einstellungen...	IDC_SCHEMA_ORACLE_SCHEMA_SETTING S	3357 8
Microsoft SQL Server Schema Extensions aktivieren	IDC_SCHEMA_SQLSERVER_EXTENSIONS	3358 8
Named Schema Relationships...	IDC_SCHEMA_SQLSERVER_GLOBAL_RELA TIONSHPIS	3358 9
Unnamed Element Relationships...	IDC_SCHEMA_SQLSERVER_LOCAL_RELATI ONSHIPS	3359 0
Tamino Schema Extensions aktivieren	IDC_SCHEMA_TAMINO_EXTENSIONS	3359 1
Tamino Schema-Eigenschaften...	IDC_SCHEMA_TAMINO_GLOBAL_SETTINGS	3414 9
Verbindung zu SchemaAgent Server herstellen...	IDC_SCHEMA_SCHEMAAGENT_SERVER_C ONNECT	3358 2
Verbindung mit SchemaAgent Server trennen	IDC_SCHEMA_SCHEMAAGENT_SERVER_DI SCONNECT	3358 3
In SchemaAgent anzeigen/Nur Schema	IDC_SCHEMAAGENT_SHOW	3350 4
In SchemaAgent anzeigen/Schema und alle direkt referenzierten	IDC_SCHEMAAGENT_SHOW_RELATED	3350 7
In SchemaAgent anzeigen/Schema und alle referenzierten	IDC_SCHEMAAGENT_SHOW_ALL	3350 5
In SchemaAgent anzeigen/Schema and alle verknüpften	IDC_SCHEMAAGENT_SHOW_ALL_REACHA BLE	3350 6
Erweiterte Validierung	IDC_SCHEMA_EXTVALID_MENU	3353 9

4.3.7 Menü "XSL/XQuery"

Befehle aus dem Menü "XSL/XQuery":

Menütext	Befehlsname	ID
XSL-Transformation	IDC_TRANSFORM_XSL	3300 6
XSL:FO-Transformation	IDC_TRANSFORM_XSLFO	3300 7
XSL-Parameter/XQuery-Variablen. ..	IDC_TRANSFORM_XSL_PARAMS	3300 8
XQuery-Ausführung	IDC_TRANSFORM_XQUERY	3417 0

XSLT 2 / XQuery Profilerstellung aktivieren....	IDC_PROFILING_OPTIONS	3410 5
XSL zuweisen...	IDC_ASSIGN_XSL	3300 1
XSL:FO zuweisen...	IDC_ASSIGN_XSLFO	3300 2
XML-Beispieldatei zuweisen...	IDC_ASSIGN_SAMPLE_XML	3300 0
Gehe zu XSL	IDC_GOTO_XSL	3300 4
Debugger starten	ID_PROCESS_XSL	3421 2
Debugger anhalten	ID_XSLT_DEBUGGER_STOP	3301 7
Debugger neu starten	ID_XSLT_DEBUGGER_RESTART	3301 3
Debugger-Sitzung beenden	ID_XSLT_DEBUGGER_END_SESSION	3301 1
Einsteigen	ID_XSLT_DEBUGGER_STEP	3301 4
Aussteigen	ID_XSLT_DEBUGGER_STEP_OUT	3301 5
Überspringen	ID_XSLT_DEBUGGER_STEP_OVER	3301 6
Aktuellen Execution Node einblenden	ID_XSLT_DEBUGGER_GO_TO_CURRENT_EXECUTION_NODES	3301 2
Breakpoint einfügen/löschen	IDC_TOGGLE_BREAKPOINT	3424 6
Tracepoint einfügen/löschen	IDC_TOGGLE_TRACEPOINT	3424 8
Breakpoint aktivieren/deaktivieren	IDC_ENABLE_BREAKPOINT	3424 5
Tracepoint aktivieren/deaktivieren	IDC_ENABLE_TRACEPOINT	3424 7
Breakpoints/Tracepoints...	ID_XSLTDEBUGGER_BREAKPOINTS	3300 9
Debug-Fenster/Call Stack	ID_XSL_DEBUGWINDOWS_CALLSTACK	3423 8
Debug-Fenster/XPath-Watch	ID_XSL_DEBUGWINDOWS_WATCH	3424 4
Debug-Fenster/Kontext	ID_XSL_DEBUGWINDOWS_CONTEXT	3423 9
Debug-Fenster/Variablen	ID_XSL_DEBUGWINDOWS_VARIABLE	3424 3
Debug-Fenster/Meldungen	ID_XSL_DEBUGWINDOWS_MESSAGES	3424 0
Debug-Fenster/Vorlagen	ID_XSL_DEBUGWINDOWS_TEMPLATES	3424 1
Debug-Fenster/Info	ID_XSLXQUERY_DEBUGWINDOWS_INFO	3423 7
Debug-Fenster/Ablaufverfolgung	ID_XSL_DEBUGWINDOWS_TRACES	3424 2

XSLT/XQuery-Einstellungen...	ID_XSLTDEBUGGER_SETTINGS	3301 0

4.3.8 Menü "Authentic"

Befehle aus dem Menü "Authentic":

Menütext	Befehlsname	ID
Neues Dokument...	IDC_AUTHENTIC_NEW_FILE	34036
Datenbankdaten bearbeiten...	IDC_AUTHENTIC_EDIT_DB	34035
StyleVision Stylesheet zuweisen...	IDC_ASSIGN_SPS	34034
StyleVision Stylesheet bearbeiten	IDC_EDIT_SPS	34060
XML Entities definieren...	IDC_DEFINE_ENTITIES	32805
Markup-Symbole ausblenden	IDC_MARKUP_HIDE	34087
Kleine Markup-Symbole einblenden	IDC_MARKUP_SMALL	34090
Große Markup-Symbole einblenden	IDC_MARKUP_LARGE	34088
Gemischte Markup-Symbole einblenden	IDC_MARKUP_MIX	34089
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

4.3.9 Menü "Konvertieren"

Befehle aus dem Menü "Konvertieren":

Menütext	Befehlsname	ID
Textdatei importieren...	IDC_IMPORT_TEXT	34082
Datenbankdaten importieren...	IDC_IMPORT_DATABASE	34080
Microsoft 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
Tamino/Suchen ...	IDC_TAMINO_XQUERY	34161
Tamino/Schema auflisten ...	ID_TAMINO_GETSCHEMA	34229
Tamino/XML-Beispieldatei generieren...	IDC_GENERATE_XML_FROM_SCHEMA	34069
Tamino/Dokument hinzufügen	IDC_TAMINO_ADDFILE	34159
Tamino/Dokument löschen	ID_TAMINO_DELETE_FILE	34228

Tamino/aktives Dokument abrufen	IDC_TAMINO_GETFILE	34160
Tamino/Aktualisieren	ID_TAMINO_REFRESH_DOC	34230
Tamino/In Tamino speichern ...	IDC_SCHEMA_TAMINO_DEFINE	34148
Tamino/Tamino Schema-Eigenschaften...	IDC_SCHEMA_TAMINO_GLOBAL_SETTING S	34149
Oracle XML DB/Suchen...	ID_CONVERT_ORACLEXMLDB_QUERY	34207
Oracle XML DB/Schemas auflisten...	ID_ORACLEXMLDB_LISTSCHEMAS	34211
Oracle XML DB/Schema hinzufügen...	ID_CONVERT_ORACLEXMLDB_ADDSCHE MA	34204
Oracle XML DB/Oracle XML-Dokumente durchsuchen...	ID_CONVERT_ORACLEXMLDB_BROWSE	34205
Oracle XML DB/Eigenschaften...	ID_CONVERT_ORACLEXMLDB_PROPERTI ES	34206

4.3.10 Menü "Ansicht"

Befehle aus dem Menü "Ansicht":

Menütext	Befehlsname	ID
Text -Ansicht	IDC_VIEW_TEXT	34180
Grid-Ansicht	IDC_VIEW_GRID	34178
Schema/WSDL Design-Ansicht	IDC_VIEW_SCHEMA	34179
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
Zeilennummernleiste	IDC_TOGGLE_NUMLINEMARGIN	34168
Lesezeichenleiste	IDC_TOGGLE_BOOKMARKMARGIN	34163
Klappleiste	IDC_TOGGLE_FOLDINGMARGIN	34166
Einrücklinien	IDC_TOGGLE_INDENTGUIDES	34167

4.3.11 Menü "Browser"

Befehle aus dem Menü "Browser":

Menütext	Befehlsname	ID
Zurück	IDC_BROWSER_BACK	34039
Vorwärts	IDC_BROWSER_FORWARD	34045
Abbrechen	IDC_BROWSER_STOP	34047

Aktualisieren	IDC_BROWSER_REFRESH	34046
Schriftgröße/Maximale Größe	IDC_BROWSER_FONT_LARGEST	34041
Schriftgröße/Größer	IDC_BROWSER_FONT_LARGE	34040
Schriftgröße/Mittel	IDC_BROWSER_FONT_MEDIUM	34042
Schriftgröße/Kleiner	IDC_BROWSER_FONT_SMALL	34043
Schriftgröße/Minimale Größe	IDC_BROWSER_FONT_SMALLEST	34044
Neues Fenster	IDC_BROWSER_USE_OWN_FRAME	34048

4.3.12 Menü "WSDL"

Befehle aus dem Menü "WSDL":

Menütext	Befehlsname	ID
Messages/Message einfügen	ID_WSDL_MESSAGES_ADDNEWMESSAGE	33715
Messages/Message löschen	ID_WSDL_MESSAGES_DELETESELECTEDMESSAGE	33717
Messages/Message Part (Parameter) hinzufügen	ID_WSDL_MESSAGES_ADDMESSAGEPART	33714
Messages/Message Part (Parameter) löschen	ID_WSDL_MESSAGES_DELETEMESSAGEPART	33716
Operations/Operation einfügen	ID_WSDL_OPERATIONS_INSERTOPERATION	33725
Operations/Operation löschen	ID_WSDL_OPERATIONS_DELETEOPERATION	33724
Operations/Input-Element hinzufügen	ID_WSDL_OPERATIONS_ADDINPUTFUNCTION	33719
Operations/Output-Element hinzufügen	ID_WSDL_OPERATIONS_ADDOUTPUTFUNCTION	33721
Operations/Fault-Element hinzufügen	ID_WSDL_OPERATIONS_ADDFAULTFUNCTION	33718
Operations/Input/Output/Fault-Element löschen	ID_WSDL_OPERATIONS_DELETEINPUTOUTPUTFUNCTION	33723
Operations/Neue Message zu Input/Output/Fault-Element hinzufügen	ID_WSDL_OPERATIONS_ADDNEWMESSAGETOTHISELEMENT	33720
Operations/leere Operation	ID_WSDL_OPERATIONS_APPENDAOOPERATIONTOTHISPORTTYPE	33722
PortType/PortType einfügen	ID_WSDL_PORTTYPE_INSERTAPORTTYPE	33727
PortType/PortType löschen	ID_WSDL_PORTTYPE_DELETETHISPORTTYPE	33726
Binding/Binding einfügen	ID_WSDL_BINDING_NEWBINDING	33713
Binding/Binding löschen	ID_WSDL_BINDING_DELETEBINDING	33711
Binding/Child anhängen/soap:body	ID_WSDL_BINDING_APPENDEXTENSIBILITY_SOAPBODY	33706
BindingChild anhängen/soap:header	ID_WSDL_BINDING_APPENDEXTENSIBILITY_SOAPHEADER	33708
Binding/Child anhängen/soap:headerfault	ID_WSDL_BINDING_APPENDEXTENSIBILITY_SOAPHEADERFAULT	33709

Binding/Child anhängen/soap:fault	ID_WSDL_BINDING_APPENDEXTENSIBILITY_SOAPFAULT	33707
Binding/Child anhängen/mime:content	ID_WSDL_BINDING_APPENDEXTENSIBILITY_MIMECONTENT	33702
Binding/Child anhängen/mime:multipartrelated	ID_WSDL_BINDING_APPENDEXTENSIBILITY_MIMEMULTIPARTRELATED	33704
Binding/Child anhängen/mime:part	ID_WSDL_BINDING_APPENDEXTENSIBILITY_MIMEPART	33705
Binding/Child anhängen/mime:mimeXml	ID_WSDL_BINDING_APPENDEXTENSIBILITY_MIMEMIMEXML	33703
Binding/Child anhängen/http:urlencoded	ID_WSDL_BINDING_APPENDEXTENSIBILITY_HTTPURL_ENCODED	33700
Binding/Child anhängen/http:urlreplacement	ID_WSDL_BINDING_APPENDEXTENSIBILITY_HTTPURL_REPLACEMENT	33701
Binding/Extensibility Element löschen	ID_WSDL_BINDING_DELETEEXTENSIBILITY	33712
Service/Service einfügen	ID_WSDL_SERVICE_INSERTSERVICE	33731
Service/Service löschen	ID_WSDL_SERVICE_DELETETHISSERVICE	33729
Service/Port einfügen	ID_WSDL_SERVICE_INSERTNEWPORT	33730
Service/Port löschen	ID_WSDL_SERVICE_DELETETHISPORT	33728
Types/Neues Schema	ID_WSDL_TYPES_NEWSHEMA	33733
Types/Schema(s) in Schema-Ansicht bearbeiten	ID_WSDL_TYPES_EDITTHISSHEMA	33732
Diagramm speichern...	IDC_WSDL_SAVE_DIAGRAM	34001
Dokumentation generieren...	ID_WSDL_GENERATEDOCUMENTATION	34233
WSDL-Dokument neu parsen	IDC_WSDL_REPARSE	33774

4.3.13 Menü "SOAP"

Befehle aus dem Menü "SOAP":

Menütext	Befehlsname	ID
Neue SOAP Request erzeugen	ID_SOAP_GENERATESOAPMESSAGE	34224
Request an Server senden	ID_SOAP_SENDREQUESTTOSERVER	34225
SOAP Request Parameter ändern	ID_SOAP_SOAPREQUESTSETTINGS	34227
Soap Debugger-Sitzung	ID_SOAP_SOAPDEBUGGER	34226
Start	ID_SOAPDEBUGGER_BUTTONPLAY	34221
Einzel schritt	ID_SOAPDEBUGGER_SINGLESTEP	34222
Beim nächsten 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

4.3.14 Menü "Extras"

Befehle aus dem Menü "Extras":

Menütext	Befehlsname	ID
Rechtschreibung...	IDC_SPELL_CHECK	34154
Rechtschreibung - Optionen...	IDC_SPELL_OPTIONS	34155
Skripting-Umgebung...	ID_WINDOW_SWITCHTOVBA	34231
Makros anzeigen...	ID_WINDOW_VBAMACROS	34232
Projekt/Skripts zuweisen	ID_SCRIPTINGPRJ_ASSIGN	34214
Projekt/Projektskripts entkoppeln	ID_SCRIPTINGPRJ_UNASSIGN	34215
Projekt/Projektskripts aktivieren	ID_SCRIPTINGPRJ_ACTIVE	34213
Geöffnete Datei vergleichen mit...	ID_XMLDIFF_CHOOSE_FILES	34235
Verzeichnisse vergleichen...	ID_XMLDIFF_CHOOSE_DIRECTORIES	34234
Vergleichsoptionen...	ID_XMLDIFF_SETTINGS	34236
Anpassen...	IDC_CUSTOMIZE	34055
Optionen...	IDC_SETTINGS	33300
<placeholder>	ID_SCRIPTING_MACROITEMS	34249

4.3.15 Menü "Fenster"

Befehle aus dem Menü "Fenster":

Menütext	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

4.3.16 Menü "Hilfe"

Befehle aus dem Menü "Hilfe":

Menütext	Befehlsname	ID
Inhaltsverzeichnis...	IDC_HELP_CONTENTS	34076
Index...	IDC_HELP_INDEX	34077
Suchen...	IDC_HELP_SEARCH	34079
Tastaturzuordnungen...	IDC_HELP_KEYMAPDLG	34078
Software-Aktivierung...	IDC_ACTIVATION	34005

Bestellformular...	IDC_OPEN_ORDER_PAGE	34094
Registrieren...	IDC_REGISTRATION	34131
Auf Updates überprüfen...	IDC_CHECK_FOR_UPDATES	34275
Support Center...	IDC_OPEN_SUPPORT_PAGE	34096
Fragen und Antworten im Web...	IDC_SHOW_FAQ	34153
Komponenten und Gratistools downloaden...	IDC_OPEN_COMPONENTS_PAGE	34093
XMLSpy im Internet..	IDC_OPEN_XML_SPY_HOME	34098
XMLSpy Training...	ID_HELP_XMLSPYTRAINING	34210
Über XMLSpy...	ID_APP_ABOUT	57664

4.4 Aufrufen von XMLSpyAPI

Diese Dokumentation beschäftigt sich hauptsächlich mit den ActiveX Controls und Schnittstellen, die für die Integration der XMLSpy Benutzeroberfläche in Ihre Applikation benötigt werden. Über die folgenden Eigenschaften haben Sie Zugriff auf das XMLSpy Automation Interface (XMLSpyAPI), um die Funktionalitäten der integrierten Komponenten zu automatisieren oder zu definieren.

[XMLSpyControl.Application](#)

[XMLSpyControlDocument.Document](#)

[XMLSpyControlPlaceholder.Project](#)

Für die Verwendung des XMLSpy Automation Interface gelten beim Integrieren von XMLSpyControl auf Dokumentebene einige Einschränkungen. Nähere Informationen dazu finden Sie unter [Integration auf Dokumentebene](#).

4.5 Objektreferenz

Objekte:

[XMLSpyCommand](#)
[XMLSpyCommands](#)
[XMLSpyControl](#)
[XMLSpyControlDocument](#)
[XMLSpyControlPlaceHolder](#)

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](#), [XMLSpyControlDocument.Document](#) und [XMLSpyControlPlaceHolder.Project](#).

4.5.1 XMLSpyCommand

Eigenschaften:

[ID](#)
[Label](#)
[IsSeparator](#)
[ToolTip](#)
[StatusText](#)
[Accelerator](#)
[SubCommands](#)

Beschreibung:

Jedes `MapForceCommand` Objekt kann einer von drei möglichen Typen sein:

- **Befehl:** `ID` ist auf einen Wert größer als 0 gesetzt und `Label` ist auf "Command Name" gesetzt. `IsSeparator` ist `false` und die `SubCommands` Collection ist leer.
- **Trennzeichen:** `IsSeparator` ist `true`. `ID` ist 0 und `Label` ist nicht definiert. Die `SubCommands` Collection ist leer.
- **(Unter)-Menü:** Die `SubCommands` Collection enthält [Command](#) Objekte und `Label` ist der Name des Menüs. `ID` ist auf 0 gesetzt und `IsSeparator` ist `false`.

Accelerator

Eigenschaft: `Label` als `string`

Beschreibung:

Dies ist bei Befehlsobjekten, die Children der `ALL_COMMANDS` Collection sind, die für den Befehl definierte Zugriffstaste. 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**.

ID

Eigenschaft: `ID` als `long`

Beschreibung:

ID ist 0 für Trennzeichen und Menüs.

Bei Befehlen ist dies die ID, die mit [Exec](#) und [QueryStatus](#) verwendet werden kann.

IsSeparator

Eigenschaft: `IsSeparator` als [boolean](#)

Beschreibung:

True, wenn der Befehl ein Trennzeichen ist.

Label**Beschreibung:**

Label ist bei Trennzeichen leer.

Bei Befehlsobjekten, die Children der ALL_COMMANDS Collection sind, ist dies ein eindeutiger Name. Die Symbole für die Befehlsschaltflächen sind in Dateien mit diesem Namen gespeichert. Nähere Informationen dazu siehe [Abfragen von Befehlen](#).

Bei Befehlsobjekten, die Children von Menüs sind, enthält die Eigenschaft "Label" den Menütext des Befehls. Bei Untermenüs enthält diese Eigenschaft den Menütext.

StatusText

Eigenschaft: `Label` als [string](#)

Beschreibung:

Bei Befehlsobjekten, die Children der ALL_COMMANDS Collection sind, ist dies der Text, der in der Statusleiste angezeigt wird, wenn der Befehl ausgewählt ist.

SubCommands

Eigenschaft: `SubCommands` als [Commands](#)

Beschreibung:

Die `SubCommands` Collection enthält alle untergeordneten Befehle, wenn dieser Befehl tatsächlich ein Menü oder Untermenü ist.

ToolTip

Eigenschaft: `ToolTip` als [string](#)

Beschreibung:

Bei Befehlsobjekten, die Children der ALL_COMMANDS Collection sind, ist dies der Text, der als Tooltipp angezeigt wird.

4.5.2 XMLSpyCommands

Eigenschaften:

[Count](#)

[Item](#)

Beschreibung:

Sammlung von [Command](#)-Objekten für den Zugriff auf Befehlsbezeichnungen und IDs des

XMLSpyControl. Diese Befehle können mit der [Exec](#) Methode ausgeführt werden und ihr Status kann mittels [QueryStatus](#) abgefragt werden.

Count

Eigenschaft: `Count` als [long](#)

Beschreibung:

Anzahl der [Command](#) Objekte auf dieser Ebene der Collection

Item

Eigenschaft: `Item (n als long)` als [Command](#)

Beschreibung:

Ruft den Befehl mit den Index `n` in dieser Sammlung auf. Der Index basiert auf 1.

4.5.3 XMLSpyControl

Eigenschaften:

[IntegrationLevel](#)

[Appearance](#)

[Application](#)

[BorderStyle](#)

[CommandsList](#)

`CommandsStructure` (deprecated)

[EnableUserPrompts](#)

[MainMenu](#)

[Toolbars](#)

Methoden:

[Open](#)

[Exec](#)

[QueryStatus](#)

Events:

[OnUpdateCmdUI](#)

[OnOpenedOrFocused](#)

[OnCloseEditingWindow](#)

[OnFileChangedAlert](#)

[OnContextChanged](#)

[OnDocumentOpened](#)

[OnValidationWindowUpdated](#)

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

Eigenschaften

Es sind die folgenden Eigenschaften definiert:

[IntegrationLevel](#)

[EnableUserPrompts](#)

[Appearance](#)

[BorderStyle](#)

Befehlsbezogene Eigenschaften:

[CommandsList](#)

[MainMenu](#)

[Toolbars](#)

CommandsStructure (nicht mehr verwendet)

Zugriff auf die XMLSpyAPI:

[Application](#)

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.

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.

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.

CommandsList

Eigenschaft: CommandList als [Commands](#) (schreibgeschützt)

Dispatch Id: 1004

Beschreibung:

Diese Eigenschaft gibt eine flache Liste aller Befehle zurück, die mit XMLSpyControl verfügbar sind. Nähere Informationen dazu siehe [C#-Beispiel](#).

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`.

IntegrationLevel

Eigenschaft: IntegrationLevel als [ICActiveXIntegrationLevel](#)

Dispatch Id: 1000

Beschreibung:

Die Eigenschaft `IntegrationLevel` bestimmt den Operationsmodus des Control. Nähere Informationen dazu siehe auch [Integration auf Applikationsebene](#) und [Integration auf Dokumentenebene](#).

Anmerkung: Diese Eigenschaft muss unbedingt sofort nach Erstellung des `XMLSpyControl` Objekts definiert werden.

MainMenu

Eigenschaft: MainMenu als [Command](#)(schreibgeschützt)

Dispatch Id: 1003

Beschreibung:

Mit dieser Eigenschaft erhalten Sie Zugriff auf die Beschreibung des `XMLSpyControl` Hauptmenüs. Nähere Informationen dazu siehe [C#-Beispiel](#).

Toolbars

Eigenschaft: Toolbars als [Commands](#)(schreibgeschützt)

Dispatch Id: 1005

Beschreibung:

Diese Eigenschaft gibt eine Liste aller Symbolleistenbeschreibungen zurück, die alle mit `XMLSpyControl` verfügbaren Symbolleisten beschreiben. Nähere Informationen dazu siehe [C#-Beispiel](#).

Methoden

Es sind die folgenden Methoden definiert:

[Open](#)

[Exec](#)

[QueryStatus](#)

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 `CommandsStructure` und eine Beschreibung, wie man den Status eines Befehls abrufen, finden Sie unter [QueryStatus](#).

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 Dokumentenebene zu laden. Verwenden Sie statt dessen [XMLSpyControlDocument.Open](#) und [XMLSpyControlPlaceholder.OpenProject](#).

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.

Events

Das XMLSpyControl ActiveX Control stellt die folgenden Verbindungspunkt-Events bereit:

[OnUpdateCmdUI](#)

[OnOpenedOrFocused](#)
[OnCloseEditingWindow](#)
[OnFileChangedAlert](#)
[OnContextChanged](#)

[OnDocumentOpened](#)
[OnValidationWindowUpdated](#)

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 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.

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

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](#) zu verwenden.

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.

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).

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.

OnToolWindowUpdated

Event: OnToolWindowUpdated (pToolWnd als long)

Dispatch Id: 1006

Beschreibung:

Dieses Event wird ausgelöst, wenn das Fenster "Extras" aktualisiert wird.

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](#) zu überprüfen. Führen Sie an diesem Callback keine langen Operationen durch.

OnValidationWindowUpdated

Event: OnValidationWindowUpdated ()

Dispatch Id: 3

Beschreibung:

Dieses Event wird ausgelöst, wenn das Fenster "Validierungsausgabe" mit neuen Informationen aktualisiert wird.

4.5.4 XMLSpyControlDocument

Eigenschaften:

[Appearance](#)
[BorderStyle](#)
[Document](#)
[IsModified](#)
[Path](#)
[ReadOnly](#)

Methoden:

[Exec](#)
[New](#)
[Open](#)
[QueryStatus](#)
[Reload](#)
[Save](#)
[SaveAs](#)

Events:

[OnDocumentOpened](#)
[OnDocumentClosed](#)
[OnModifiedFlagChanged](#)
[OnContextChanged](#)
[OnFileChangedAlert](#)
[OnActivate](#)

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

Eigenschaften

Es sind die folgenden Eigenschaften definiert:

[ReadOnly](#)
[IsModified](#)
[Path](#)
[Appearance](#)
[BorderStyle](#)

Zugriff auf die XMLSpyAPI:

[Document](#)

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.

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.

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.

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*.

Path

Eigenschaft: `Path` als [String](#)

Dispatch Id: 1005

Beschreibung:

Definiert den vollständigen Pfadnamen des im Control geladenen Dokuments bzw. ruft diesen ab.

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.

Methoden

Es sind die folgenden Methoden definiert:

Behandlung von Dokumenten:

[New](#)

[Open](#)

[Reload](#)

[Save](#)

[SaveAs](#)

Behandlung von Befehlen:

[Exec](#)

[QueryStatus](#)

Exec

Methode: `Exec` (`nCmdID` als [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. Der Client sollte die `Exec` Methode des Dokuments aufrufen, wenn in der Applikation gerade ein aktives Dokument verfügbar ist.

Eine Liste aller verfügbaren Befehle finden Sie unter `CommandsStructure` und eine Beschreibung, wie man den Status eines Befehls abrufen, finden Sie unter [QueryStatus](#).

New

Methode: `New ()` als `boolean`

Dispatch Id: 1000

Beschreibung:

Diese Methode initialisiert ein neues Mapping innerhalb des Control.

Open

Methode: `Open (strFilePath als string)` als `boolean`

Dispatch Id: 1001

Beschreibung:

`Open` lädt die Datei `strFileName` als das neue Dokument in das Control.

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
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.

Reload

Methode: `Reload ()` als `boolean`

Dispatch Id: 1002

Beschreibung:

`Reload` aktualisiert den Dokumentinhalt aus dem Dateisystem.

Save

Methode: `Save ()` als `boolean`

Dispatch Id: 1003**Beschreibung:**

Save speichert das aktuelle Dokument unter dem Pfad [Path](#).

SaveAs

Method: SaveAs (strFileName als [string](#)) als [boolean](#)

Dispatch Id: 1004**Beschreibung:**

SaveAs setzt [Path](#) auf strFileName und speichert das Dokument anschließend unter diesem Pfad.

Events

Das XMLSpyControlDocument ActiveX Control stellt die folgenden Verbindungspunkt-Events zur Verfügung:

[OnDocumentOpened](#)
[OnDocumentClosed](#)
[OnModifiedFlagChanged](#)
[OnContextChanged](#)
[OnFileChangedAlert](#)
[OnActivate](#)
[OnSetEditorTitle](#)

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.

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.

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.

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.

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.

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.

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_bIsModified` ist `true`, wenn sich der Inhalt des Dokuments vom ursprünglichen Inhalt unterscheidet und ist andernfalls `false`.

OnSetEditorTitle**Event:** OnSetEditorTitle ()**Dispatch Id:** 1006**Beschreibung:**

Dieses Event wird ausgelöst, wenn das enthaltene Dokument intern umbenannt wird.

4.5.5 XMLSpyControlPlaceholder

Eigenschaften für alle Arten von Platzhalterfenstern:[PlaceholderWindowID](#)**Eigenschaften für das Projekt-Platzhalterfenster:**[Project](#)**Methoden für das Projekt-Platzhalterfenster:**[OpenProject](#)[CloseProject](#)

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

Eigenschaften

Es sind die folgenden Eigenschaften definiert:

[PlaceholderWindowID](#)

Zugriff auf die XMLSpyAPI:

[Project](#)**Label****Eigenschaft:** Label als [String](#) (schreibgeschützt)**Dispatch Id:** 1001**Beschreibung:**

Mit dieser Eigenschaft erhalten Sie Zugriff auf den Titel des Platzhalters. Diese Eigenschaft ist schreibgeschützt.

PlaceholderWindowID

Eigenschaft: PlaceholderWindowID als [XMLSpyControlPlaceholderWindow](#)

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](#) gesetzt werden. Das Control ändert seinen Status sofort und zeigt das neue XMLSpy Fenster an.

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](#) mit einem Wert von XMLSpyXProjectWindow (=3) hat. Die Eigenschaft ist schreibgeschützt.

Methoden

Es sind die folgenden Methoden definiert:

[OpenProject](#)

[CloseProject](#)

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](#) hat, die nicht mit XMLSpyXProjectWindow (=3) übereinstimmt, schlägt die Methode fehl.

CloseProject

Methode: CloseProject ()

Dispatch Id: 4

Beschreibung:

CloseProject schließt das in das Control geladene Projekt. Wenn das Platzhalterfenster eine [PlaceholderWindowID](#) hat, die nicht mit XMLSpyXProjectWindow (=3) übereinstimmt, schlägt die Methode fehl.

Events

Das XMLSpyControlPlaceholder ActiveX Control stellt die folgenden Verbindungspunkt-Events zur Verfügung:

[OnModifiedFlagChanged](#)

OnModifiedFlagChanged

Event: OnModifiedFlagChanged (i_bIsModified als [boolean](#))

Dispatch Id: 1

Beschreibung:

Dieses Event wird nur bei Platzhalter-Controls mit einer [PlaceholderWindowID](#) von XMLSpy XProjectWindow (=3) ausgelöst. Das Event wird immer dann ausgelöst, wenn der Projektinhalt sich zwischen dem Status "geändert" und "nicht geändert" ändert. Der Parameter *i_bIsModified* ist *true*, wenn sich der Projektinhalt vom ursprünglichen Inhalt unterscheidet und *false*, wenn dies nicht der Fall ist.

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.

4.5.6 Enumerationen

Es sind die folgenden Enumerationen definiert:

[ICActiveXIntegrationLevel](#)
[XMLSpyControlPlaceholderWindow](#)

ICActiveXIntegrationLevel

Mögliche Werte für die Eigenschaft [IntegrationLevel](#) für das XMLSpyControl.

```
ICActiveXIntegrationOnApplicationLevel = 0
ICActiveXIntegrationOnDocumentLevel   = 1
```

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

Altova XMLSpy 2012

Anhänge

Anhänge

Diese Anhänge enthalten technische Informationen über XMLSpy und wichtige Lizenzinformationen. Die einzelnen Anhänge sind in die folgenden Unterabschnitte gegliedert:

Technische Daten

- [Altova XSLT 1.0-Prozessor](#)
- [Altova XSLT 2.0-Prozessor](#)
- [Altova XQuery 1.0-Prozessor](#)
- [XPath 2.0- und XQuery 1.0-Funktionen](#)
- [Erweiterungen](#)

Datentypkonvertierungen zwischen Datenbanken und XML-Schemas

- [Datenbanken in XML-Schemas](#)
- [XML-Schemas in Datenbanken](#)

Technische Daten

- [Betriebssystem und Arbeitsspeicheranforderungen](#)
- [AltovaXML Parser](#)
- [Altova XSLT- und XQuery-Prozessor](#)
- [Unicode-Unterstützung](#)
- [Internet-Verwendung](#)
-

Lizenzinformationen

- [Electronic Software Distribution](#)
- [Software-Aktivierung und Lizenzüberwachung](#)
- [Rechte am geistigen Eigentum](#)
- [Endbenutzer-Lizenzvereinbarung](#)

1 Prozessor-Informationen

Dieser Abschnitt enthält Informationen über implementierungsspezifische Funktionen des [Altova XSLT 1.0-Prozessors](#), [Altova XSLT 2.0-Prozessors](#) und des [Altova XQuery 1.0-Prozessors](#).

1.1 XSLT 1.0-Prozessor: Implementierungsinformationen

Der Altova XSLT 1.0-Prozessor ist in die Altova XML-Produkte XMLSpy, StyleVision, Authentic und MapForce integriert. Außerdem steht er im kostenlosen AltovaXML-Paket zur Verfügung. Im Altova XSLT 1.0-Prozessor sind die W3C-Spezifikationen implementiert und entsprechen der [XSLT 1.0 Recommendation vom 16. November 1999](#) und der [XPath 1.0 Recommendation vom 16. November 1999](#). Im Folgenden sind Einschränkungen und implementierungsspezifisches Verhalten aufgelistet.

Einschränkungen

- Die Elemente `xsl:preserve-space` und `xsl:strip-space` werden nicht unterstützt.
- Wenn das Attribut `method` von `xsl:output` auf HTML gesetzt ist, oder wenn standardmäßig HTML output ausgewählt ist, werden Sonderzeichen in der XML- oder XSLT-Datei direkt als Sonderzeichen in die HTML-Dokument eingefügt und nicht als HTML-Zeichenreferenzen in der Ausgabe. So wird z.B. das Zeichen ` ` (die Dezimalzeichenreferenz für ein geschütztes Leerzeichen) nicht als ` ` in den HTML-Code eingefügt sondern direkt als geschütztes Leerzeichen.

Behandlung der Implementierung von Nur-Whitespace Nodes in XML-Quelldokumenten

Aus den XML-Daten (und somit dem XML-Infoset), die an den Altova XSLT 1.0-Prozessor übergeben werden, werden boundary-whitespace-only Text Nodes entfernt. (Ein boundary-whitespace-only Text Node ist ein Textnode, der nur Whitespaces enthält und der zwischen zwei Elementen innerhalb eines mixed-content-Elements vorkommt). Dies kann sich auf den Wert auswirken, der von den Funktionen `fn:position()`, `fn:last()` und `fn:count()` zurückgegeben wird.

In jeder Node-Auswahl, bei der auch Text Nodes ausgewählt werden, sind normalerweise auch Boundary-whitespace-only Text Nodes enthalten. Da jedoch beim XML Infoset, das von den Altova-Prozessoren verwendet wird, boundary-whitespace-only Text Nodes entfernt werden, sind diese Nodes im XML Infoset nicht vorhanden. Folglich unterscheidet sich die Größe der Auswahl und die Nummerierung der Nodes von einer, in der diese Text Nodes enthalten sind. Aus diesem Grund können sich die Ergebnisse der Funktionen `fn:position()`, `fn:last()` und `fn:count()` von denen anderer Prozessoren unterscheiden.

Am häufigsten tritt die Situation, dass boundary-whitespace-only Text Nodes als gleichrangige Elemente anderer Elemente überprüft werden, dann auf, wenn `xsl:apply-templates` verwendet wird, um Templates anzuwenden. Wenn die Funktionen `fn:position()`, `fn:last()` und `fn:count()` in Patterns mit einer Namensüberprüfung verwendet werden (z.B. `para[3]`, kurz für `para[position()=3]`), sind boundary-whitespace-only Nodes irrelevant, da nur die benannten Elemente (`para` im obigen Beispiel) ausgewählt werden. (Beachten Sie jedoch, dass boundary-whitespace-only Nodes in Patterns, in denen ein Platzhalterzeichen wie z.B. `*[10]` verwendet wird, sehr wohl relevant sind).

Hinweis: Wenn der boundary-whitespace-only Text Node in der Ausgabe benötigt wird, fügen Sie das erforderliche Whitespace-Zeichen in eines der benachbarten Child-Elemente ein. So erzeugt z.B. das XML-Fragment:

```
<para>This is <b>bold</b> <i>italic</i>.</para>
```

bei Verarbeitung mit dem XSLT Template

```
<xsl:template match="para">
  <xsl:apply-templates/>
</xsl:template>
```


folgendes Resultat:

This is **bolditalic**.

Um in der Ausgabe ein Leerzeichen zwischen `bold` und `italic` zu erhalten, fügen Sie entweder in das Element `<b>` oder `<i>` in der XML-Quelle ein Leerzeichen ein. z.B.:

```
<para>This is <b>bold</b> <i> italic</i>. </para> or  
<para>This is <b>bold&#x20;</b> <i>italic</i>. </para> or  
<para>This is <b>bold</b><i>&#x20; italic</i>. </para>
```

Wenn eines der oben genannten `para`-Elemente mit demselben XSLT Template verarbeitet wird, erhalten Sie folgendes Ergebnis:

This is bold italic.

1.2 XSLT 2.0-Prozessor: Implementierungsinformationen

Der Altova XSLT 2.0-Prozessor ist in die Altova XML-Produkte XMLSpy, StyleVision, Authentic und MapForce integriert. Außerdem steht er im kostenlosen AltovaXML-Paket zur Verfügung. In diesem Abschnitt werden die implementierungsspezifischen Aspekte des Prozessors beschrieben. Im ersten Teil des Abschnitts finden Sie allgemeine Informationen über den Prozessor. Im Anschluss daran finden Sie eine Liste der implementierungsspezifischen Aspekte der XSLT 2.0-Funktionen.

Informationen über das implementierungsspezifische Verhalten von XPath 2.0-Funktionen finden Sie im Abschnitt [XPath 2.0- und XQuery 1.0-Funktionen](#).

1.2.1 Allgemeine Informationen

Der Altova XSLT 2.0-Prozessor entspricht der [XSLT 2.0 Recommendation](#) vom 23 January 2007. Beachten Sie bitte die folgenden allgemeinen Informationen über den Prozessor.

Rückwärtskompatibilität

Der Altova 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 von Altova XML 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.

Bei allen anderen Altova-Produkten wird Rückwärtskompatibilität nie benötigt, da automatisch der entsprechende Prozessor für die Transformation ausgewählt wird. Angenommen, Sie legen fest, dass ein bestimmtes XML-Dokument in XMLSpy mit einem XSLT 1.0 Stylesheet verarbeitet werden soll. Bei Aufruf des Transformationsbefehls wählt XMLSpy zur Transformation automatisch den XSLT 1.0-Prozessor aus.

Anmerkung:

Diese Auswahl basiert auf dem Wert im Attribut `version` des Stylesheet-Elements `stylesheet` oder `transform`.

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 bitte die folgenden Punkte:

- Der Altova XSLT 2.0-Prozessor verwendet als Default Functions 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-Konstrukturen und Typen aus dem XML Schema-Namespaces, muss bei Aufruf des Typ-Konstruktors (z.B. `xs:date`) das in der jeweiligen Namespace-Deklaration verwendete Präfix verwendet werden.
- In den Candidate Recommendations vom 23 January 2007 wurden die Datentypen `untypedAtomic` und `duration` (`dayTimeDuration` und `yearMonthDuration`), die sich zuvor im XPath Datatypes Namespace befanden (normalerweise mit dem Präfix `xdt:`) in den XML-Schema-Namespaces verschoben.
- 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')`.

Schemafähigkeit

Der Altova XSLT 2.0-Prozessor ist schemafähig.

Whitespaces im XML-Dokument

Standardmäßig entfernt der Altova XSLT 2.0-Prozessor alle boundary whitespaces aus boundary-whitespace-only Nodes im XML-Quelldokument. Dies wirkt sich auf die Werte aus, die die Funktionen `fn:position()`, `fn:last()`, `fn:count()` und `fn:deep-equal()` zurückgeben. Nähere Informationen dazu finden Sie in den Abschnitten über XPath 2.0- und XQuery 1.0-Funktionen unter [Nur-Whitespace Nodes im XML-Quelldokument](#).

Hinweis: Wenn in der Ausgabe ein boundary-whitespace-only-Text Node erforderlich ist, fügen Sie das gewünschte Whitespace-Zeichen in eines der beiden benachbarten Child-Elemente ein. So erzeugt z.B. das XML-Fragment:

```
<para>This is <b>bold</b> <i>italic</i>.</para>
```

bei Verarbeitung mit dem XSLT Template

```
<xsl:template match="para">
  <xsl:apply-templates/>
</xsl:template>
```

folgendes Resultat:

```
This is bolditalic.
```

Um in der Ausgabe ein Leerzeichen zwischen `bold` und `italic` zu erhalten, fügen Sie entweder in das Element `<b>` oder `<i>` in der XML-Quelle ein Leerzeichen ein. z.B.:

```
<para>This is <b>bold</b> <i> italic</i>.</para> oder
<para>This is <b>bold<#x20;</b> <i>italic</i>.</para> oder
<para>This is <b>bold</b><i>&#x20;italic</i>.</para>
```

Wenn nun ein solches XML-Fragment mit demselben XSLT Template verarbeitet wird, erhalten Sie folgendes Ergebnis:

This is bold italic.

XSLT 2.0-Elemente und -Funktionen

Einschränkungen und implementierungsspezifisches Verhalten von XSLT 2.0-Elementen und -Funktionen werden im Abschnitt [XSLT 2.0-Elemente und -Funktionen](#) aufgelistet.

XPath 2.0-Funktionen

Implementierungsspezifisches Verhalten von XPath 2.0-Funktionen wird im Abschnitt [XPath 2.0- und XQuery 1.0-Funktionen](#) aufgelistet.

1.2.2 XSLT 2.0-Elemente und -Funktionen

Einschränkungen

Die Elemente `xsl:preserve-space` und `xsl:strip-space` werden nicht unterstützt.

Implementierungsspezifisches Verhalten

Im Folgenden finden Sie eine Beschreibung, wie der Altova 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 2.0, XPath 2.0 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 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 Kodierungen `x-binarytobase16` und `x-binarytobase64` unterstützt.

Anmerkung: Die folgenden Kodierungswerte, die in früheren Versionen von AltovaXML implementiert waren, werden nun nicht mehr verwendet: `base16tobinary`, `base64tobinary`, `binarytobase16` und `binarytobase64`.

1.3 XQuery 1.0-Prozessor: Implementierungsinformationen

Der Altova XQuery 1.0-Prozessor ist in die Altova XML-Produkte XMLSpy und MapForce integriert. Er steht auch in Form des kostenlosen Pakets AltovaXML zur Verfügung. Dieser Abschnitt enthält Informationen über implementierungsspezifische Aspekte seines Verhaltens.

Standardkonformität

Der Altova XQuery 1.0-Prozessor entspricht den [XQuery 1.0 CR](#) vom 23 January 2007. Der Query-Standard stellt frei, wie viele Funktionen 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**.

Codierung

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
Built-in 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 Built-in Funktionen der Default Functions Namespace in XQuery ist, muss beim Aufruf von built-in-Funktionen das Präfix `fn:` nicht verwendet werden (`string("Hello")` ruft z.B. die Funktion `fn:string` auf). Das Präfix `fn:` kann jedoch verwendet werden, um eine built-in-Funktion aufzurufen, ohne die Namespace im Abfrage-Prolog deklarieren zu müssen (z.B.: `fn:string("Hello")`).
- Sie können den Default Functions Namespace durch Deklaration des `default function namespace`-Ausdrucks im Abfrageprolog ändern.
- Bei Verwendung von Typen aus dem XML Schema- und dem XPath-Datentypen-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-Konstrukturen, 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 Überprüfung 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. In der statischen Analysephase wird keine Typ-Überprüfung durchgeführt. 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. Derzeit werden keine anderen Collations unterstützt. Vergleiche, wie u.a. die Funktion `fn:max` basieren auf dieser Collation.

Zeichennormalisierung

Es wird keine Zeichennormalisierungsform unterstützt.

Präzision von numerischen Typen

- Der Datentyp `xs:integer` hat eine beliebige Präzision, dh. 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.

Unterstützung für XQuery-Funktionen

Informationen zu implementierungsspezifischem Verhalten von XQuery 1.0-Funktionen finden Sie im Abschnitt [XPath 2.0- und XQuery 1.0-Funktionen](#).

1.4 XPath 2.0- und XQuery 1.0-Funktionen

Die XPath 2.0- und XQuery 1.0-Funktionen werden von folgenden Prozessoren überprüft:

- dem **Altova XPath 2.0-Prozessor**, der (i) eine Komponente des Altova XSLT 2.0-Prozessors ist und (ii) im XPath Evaluator von Altova XMLSpy verwendet wird, um XPath-Ausdrücke hinsichtlich des in XMLSpy aktiven XML-Dokuments auszuwerten.
- dem **Altova XQuery 1.0-Prozessor**.

In diesem Abschnitt wird beschrieben, wie XPath 1.0- und XQuery 1.0-Funktionen vom Altova XPath 2.0-Prozessor und dem Altova XQuery 1.0-Prozessor behandelt werden. Es sind nur die Funktionen aufgelistet, die implementierungsspezifisch behandelt werden oder bei denen sich eine einzelne Funktion in einer der drei Umgebungen, in denen sie verwendet wird (also in XSLT 2.0, in XQuery 1.0 und im XPath Evaluator von XMLSpy), anders verhält. Beachten Sie, dass in diesem Abschnitt nicht beschrieben wird, wie diese Funktionen verwendet werden. Nähere Informationen über die Verwendung dieser Funktionen finden Sie im [XQuery 1.0 and XPath 2.0 Functions and Operators PR des W3C](#) vom 23 January 2007.

1.4.1 Allgemeine Informationen

Standardkonformität

- Der Altova XPath 2.0-Prozessor implementiert die [XPath 2.0 Recommendation](#) vom 23 January 2007. Der Altova XQuery 1.0-Prozessor implementiert die [XQuery 1.0 Recommendation](#) vom 23 January 2007. Die Unterstützung der XPath 2.0- und XQuery 1.0-Funktionen in diesen beiden Prozessoren entspricht den Vorgaben in den [XQuery 1.0 and XPath 2.0 Functions and Operators Recommendation](#) vom 23 January 2007.
- Der Altova XPath 2.0-Prozessor entspricht den Vorgaben für [XML 1.0 \(Fourth Edition\)](#) und [XML Namespaces \(1.0\)](#).

Default Functions Namespace

Der Default Functions Namespace wurde dem Standard entsprechend eingestellt. Funktionen können daher ohne Präfix aufgerufen werden.

Boundary-whitespace-only-Nodes in XML-Quelldokumenten

Aus den XML-Daten (und somit dem XML-InfoSet), die an den Altova XPath 2.0- und den Altova XQuery 1.0-Prozessor übergeben werden, werden boundary-whitespace-only Text Nodes entfernt. (Ein boundary-whitespace-only Text Node ist ein Child-Textnode, der nur Whitespaces enthält und der zwischen zwei Elementen innerhalb eines mixed-content-Elements vorkommt). Dies kann sich auf den Wert auswirken, der von den Funktionen `fn: position()`, `fn: last()`, `fn: count()` und `fn: deep-equal()` zurückgegeben wird.

In jeder Node-Auswahl, bei der auch Text Nodes ausgewählt werden, sind normalerweise auch Boundary-whitespace-only Text Nodes enthalten. Da jedoch beim XML InfoSet, das von den Altova-Prozessoren verwendet wird, boundary-whitespace-only Text Nodes entfernt werden, sind diese Nodes im XML InfoSet nicht vorhanden. Folglich unterscheidet sich die Größe der Auswahl und die Nummerierung der Nodes von einer, in der diese Text Nodes enthalten sind. Aus diesem Grund können sich die Ergebnisse der Funktionen `fn: position()`, `fn: last()`, `fn: count()` und `fn: deep-equal()` von denen anderer Prozessoren unterscheiden.

Am häufigsten tritt die Situation, dass boundary-whitespace-only Text Nodes als gleichrangige Elemente anderer Elemente überprüft werden, dann auf, wenn `xsl: apply-templates` verwendet wird, um Templates anzuwenden. Wenn die Funktionen `fn: position()`, `fn: last()` und `fn: count()` in Patterns mit einer Namensüberprüfung verwendet werden

(z.B. `para[3]`, kurz für `para[position()=3]`), sind boundary-whitespace-only Nodes irrelevant, da nur die benannten Elemente (`para` im obigen Beispiel) ausgewählt werden. (Beachten Sie jedoch, dass boundary-whitespace-only Nodes in Patterns, in denen ein Platzhalterzeichen wie z.B. `*[10]` verwendet wird, sehr wohl relevant sind).

Numerische Notation

Wenn bei der Ausgabe ein `xs:double` in einen String konvertiert wird, wird die logarithmische Darstellung (z.B. `1.0E12`) verwendet, wenn der absolute Wert kleiner als 0,000001 oder größer als 1.000.000 ist. Andernfalls wird der Wert als Dezimalwert oder Integer angegeben.

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 muss die Zeitzone 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 `fn:implicit-timezone()` überprüft werden.

Collations

Es wird nur die Unicode Codepoint Collation unterstützt. Es können keine anderen Collations verwendet werden. Der Vergleich von Strings, u.a. auch für die Funktionen `fn:max` und `fn:min` basiert auf dieser Collation.

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 `fn:in-scope-prefixes()`, `fn:namespace-uri()` und `fn:namespace-uri-for-prefix()`.

Static Type Extensions

Die optionale Funktion zum Überprüfen von statischen Typen wird nicht unterstützt.

1.4.2 Unterstützung von Funktionen

In der nachfolgenden Tabelle wird das implementierungsspezifische Verhalten bestimmter Funktionen (in alphabetischer Reihenfolge) aufgelistet. Beachten Sie bitte die folgenden allgemeinen Punkte:

- Wenn bei einer Funktion eine Sequenz von einem Datenelement als Argument erwartet wird und eine Sequenz von mehr als einem Datenelement gesendet wird, wird im allgemeinen 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.

Funktionsname	Anmerkungen
base-uri	• Wenn im XML-Quelldokument externe Entities verwendet werden und ein Node in der externen Entity als Input-Node-Argument der Funktion base-uri() definiert ist, wird trotzdem die Basis-URI des inkludierenden XML-Dokuments verwendet und nicht die Basis-URI der externen Entity. • Die Basis-URI eines Node im XML-Dokument kann mit Hilfe des Attributs <code>xml:base</code> geändert werden.
collection	• Das Argument ist eine relative URI, die gegen die aktuelle Basis-URI aufgelöst wird. • Wenn mit der aufgelösten URI eine XML-Datei identifiziert wird, so wird diese XML-Datei als Katalog behandelt, der eine Sammlung von Dateien referenziert. Diese Datei muss die folgende Form haben: <pre> <collection> <doc href="uri-1" /> <doc href="uri-2" /> <doc href="uri-3" /> </collection> </pre> Die von den <code>href</code> Attributen referenzierten Dateien werden geladen und ihre Dokument-Nodes werden als Sequenz zurückgegeben. • Wenn mit der aufgelösten URI keine XML-Datei mit der oben beschriebenen Katalogstruktur identifiziert wird, dann wird der Argument-String (in dem die Platzhalterzeichen <code>?</code> und <code>*</code> zulässig sind) als Suchstring verwendet. XML-Dateien, deren Namen dem Suchausdruck entsprechen, werden geladen und ihre Dokument-Nodes werden als Sequenz zurückgegeben. Siehe Beispiel unten. • XSLT-Beispiel: Der Ausdruck <code>collection("c:\MyDocs*.xml")//Title</code> gibt eine Sequenz aller <code>DocTitle</code> Elemente in den <code>.xml</code> Dateien im Ordner <code>MyDocs</code> zurück. • XQuery-Beispiel: Der Ausdruck <code>{for \$i in collection(c:\MyDocs*.xml) return element doc{base-uri(\$i)}}</code> gibt die Basis-URIs aller <code>.xml</code> Dateien im Ordner <code>MyDocs</code> zurück, wobei jede URI sich innerhalb eines <code>doc</code> Elements befindet. • Die Standard-Collection ist leer.
count	• Siehe Anmerkungen zu Whitespaces im Abschnitt Allgemeine Informationen.

current-date, current-dateTime, current-time	• The argument is a relative URI that is resolved against the current base URI. • If the resolved URI identifies an XML file, then this XML file is treated as a catalog which references a collection of files. This file must have the form: <pre> <collection> <doc href="uri-1" /> <doc href="uri-2" /> <doc href="uri-3" /> </collection> </pre> The files referenced by the href attributes are loaded, and their document nodes are returned as a sequence. • If the resolved URI does not identify an XML file with the catalog structure described above, then the argument string is used as a file-search expression (in which wildcards such as ? and *) are allowed. The specified directory, as identified by the base URI and search string, is searched. XML files with names that matches the search expression are loaded, and their document nodes are returned as a sequence. • The default collection is empty.
deep-equal	• Siehe Anmerkung zu Whitespace im Abschnitt Allgemeine Informationen.
doc	• Es wird nur dann ein Fehler ausgegeben, wenn unter dem angegebenen Pfad keine XML-Datei vorhanden ist oder wenn die Datei nicht wohlgeformt ist. Die Datei wird validiert, wenn ein Schema vorhanden ist. Wenn die Datei nicht gültig ist, so wird die ungültige Datei ohne Schema-Informationen geladen.
id	• In einem wohlgeformten aber ungültigen Dokument, das zwei oder mehr Elemente mit demselben ID-Wert enthält, wird das erste Element in der Dokumentenreihenfolge zurückgegeben.
in-scope-prefixes	• Im XML-Dokument dürfen nur Default Namespaces entdeclariert werden. Doch selbst wenn ein Default Namespace an einem Element-Node entdeclariert wird, wird das Präfix für den Default Namespace, welches der Nulllängenstring ist, für diesen Node zurückgegeben.
last	• Siehe Anmerkung zu Whitespace im Abschnitt Allgemeine Informationen.
lower-case	• Es wird der Unicode-Zeichensatz unterstützt.
normalize-unicode	• Es werden die Normalisierungsformulare NFC, NFD, NFKC und NFKD unterstützt.
position	• Siehe Anmerkung zu Whitespace im Abschnitt Allgemeine Informationen.

<code>resolve-uri</code>	• Wenn das zweite, optionale Argument weggelassen wird, wird die aufzulösende URI (das erste Argument) gegen die Basis-URI aus dem statischen Kontext aufgelöst. Dies ist die URI des XSLT Stylesheets oder die im Prolog des XQuery-Dokuments angegebene Basis-URI. • Die relative URI (das erste Argument) wird nach dem letzten "/" in der Pfadnotation der Basis-URI-Notation angehängt. • Wenn der Wert des ersten Arguments der Nulllängenstring ist, wird die Basis-URI aus dem statischen Kontext zurückgegeben und diese URI inkludiert den Dateinamen des Dokuments, von dem die Basis-URI des statischen Kontexts abgeleitet wird (z.B. die XSLT- oder XML-Datei).
<code>static-base-uri</code>	• Die Basis-URI aus dem statischen Kontext ist die Basis-URI des XSLT Stylesheets oder die im Prolog des XQuery-Dokuments angegebene Basis-URI. • Bei Verwendung des XPath Evaluators in der XMLSpy IDE ist die Basis-URI aus dem statischen Kontext die URI des aktiven XML-Dokuments.
<code>upper-case</code>	• Der Unicode-Zeichensatz wird unterstützt.

1.5 Erweiterungen

Es gibt in Programmiersprachen wie Java und C# eine Reihe von fertigen Funktionen, die nicht als XPath 2.0 / XQuery 1.0-Funktionen oder XSLT 2.0-Funktionen zur Verfügung stehen. Ein gutes Beispiel für solche Funktionen 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.

Die Altova-Prozessoren (XSLT 1.0, XSLT 2.0 und XQuery 1.0), die in einer Reihe von Altova-Produkten verwendet werden, unterstützen die Verwendung von Erweiterungsfunktionen in Java und .NET. Zusätzlich unterstützen die Altova XSLT-Prozessoren MSXSL Scripts für XSLT 1.0 und 2.0.

Beachten Sie, dass Erweiterungsfunktionen immer von XPath-Ausdrücken aus aufgerufen werden. In diesem Abschnitt wird beschrieben, wie Sie Erweiterungsfunktionen und MSXSL Scripts in Ihren XSLT Stylesheets und XQuery-Abfragen verwenden können. Diese Beschreibungen finden Sie in den folgenden Abschnitten:

- [Java-Erweiterungsfunktionen](#)
- [.NET-Erweiterungsfunktionen](#)
- [MSXSL Scripts für XSLT](#)
- [Altova Erweiterungsfunktionen](#)

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 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) installiert sein oder es muss Zugriff auf eine solche bestehen.

1.5.1 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](#)
- [Java: Statische Methoden und statische Felder](#)
- [Java: Instanzmethoden und Instanzfelder](#)
- [Datentypen: XSLT/XQuery in Java](#)
- [Datentypen: Java in XSLT/XQuery](#)

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 `präfix:` 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 `präfix:` 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](#) und [Benutzerdefinierte Jar-Dateien](#) beschrieben. Pfade zu Klassendateien, die sich nicht im aktuellen Verzeichnis befinden, und Pfade zu allen JAR-Dateien müssen jedoch angegeben werden.

Anmerkung: Wenn Sie einen Namespace zu einem XSLT-Stylesheet hinzufügen möchten, das anhand eines in SPS in StyleVision generiert wurde, muss der Namespace zum `schema`-Element der obersten Ebene des XML-Schemas, auf dem das SPS basiert,

hinzugefügt werden. Beachten Sie, dass die folgende Namespace Deklaration `xmlns: java="java"` standardmäßig automatisch in jedem SPS erstellt wird, das in StyleVision entworfen wurde.

Benutzerdefinierte Klassendateien

Wenn der Zugriff über eine Klassendatei erfolgt, gibt es zwei Möglichkeiten:

- Die Klassendatei befindet sich in einem Paket. Die XSLT-oder XQuery-Datei befindet sich im selben Ordner wie das Java-Paket.
- Die Klassendatei befindet sich nicht in einem Paket. Die XSLT-oder XQuery-Datei befindet sich im selben Ordner wie die Klassendatei.
- Die Klassendatei befindet sich in einem Paket. Die XSLT-oder XQuery-Datei befindet sich in irgendeinem beliebig gewählten Ordner.
- Die Klassendatei befindet sich nicht in einem Paket. Die XSLT-oder XQuery-Datei befindet sich in irgendeinem beliebig gewählten Ordner.

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 Methodenaufwurf 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 befindet sich nicht in einem Paket, die XSLT/XQuery-Datei befindet sich im selben Ordner wie die Klassendatei

Im Beispiel unten wird die Methode `getVehicleType()` der Klasse `Car` class des Pakets `com.altova.extfunc` aufgerufen. Die Klassendatei `Car` class befindet sich im folgenden Ordner: `JavaProject/com/altova/extfunc`. 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` class 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/" >

  <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 befindet sich nicht in einem Paket, die XSLT/XQuery-Datei befindet sich in einem beliebigen Ordner

Im Beispiel unten wird die Methode `getCarColor()` der Klasse `Car` class 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. Der Pfad der Klassendatei wird in der Namespace-URI als Abfragestring definiert. 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 Methodenaufwurf 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.

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 Methodenaufwurf 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/Carl.jar!/" >
<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
  <xsl:variable name="myCar" select="car:Carl.new('red')"/>
  <a><xsl:value-of select="car:Carl.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.

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](#), dann gibt die Java-Erweiterungsfunktion eine Sequenz zurück, bei der es sich um einen 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](#)):

```
<xsl:value-of select="date:toString(date:new())" xmlns:date="
java:java.util.Date"/>
```

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. `cos(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>
```

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 `java.lang.Object.getClass` übergeben.
3. Das mit der Methode `getClass` abgerufene Objekt wird als das Argument an die Methode `java.lang.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 enthalten, der von einem Java-Objekt zurückgegeben wird. So erhält z.B. der Parameter `currentTime` 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.

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.

<code>xs:string</code>	<code>java.lang.String</code>
<code>xs:boolean</code>	<code>boolean (primitive)</code> , <code>java.lang.Boolean</code>
<code>xs:integer</code>	<code>int</code> , <code>long</code> , <code>short</code> , <code>byte</code> , <code>float</code> , <code>double</code> und die Wrapper-Klassen davon wie z.B. <code>java.lang.Integer</code>
<code>xs:float</code>	<code>float (primitive)</code> , <code>java.lang.Float</code> , <code>double (primitive)</code>
<code>xs:double</code>	<code>double (primitive)</code> , <code>java.lang.Double</code>

<code>xs: decimal</code>	<code>float (primitive), java.lang.Float, double (primitive), java.lang.Double</code>

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 einige 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 Datentypes 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.

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 `java.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.

1.5.2 .NET-Erweiterungsfunktionen

Wenn Sie 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](#)
- [.NET: Statische Methoden und statische Felder](#)
- [.NET: Instanzmethoden und Instanzfelder](#)
- [Datentypen: XSLT/XQuery in .NET](#)
- [Datentypen: .NET in XSLT/XQuery](#)

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 `präfix: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:

<code>asm</code>	Der Name der zu ladenden Assembly
<code>ver</code>	Die Versionsnummer: eine Maximalzahl von vier Ganzzahlen, die durch Punkte getrennt sind
<code>sn</code>	Das Key Token des Strong Name der Assembly (16 Hex-Stellen).
<code>from</code>	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.
<code>partialname</code>	Der partielle Name der Assembly. Er wird für <code>Assembly.LoadWith.PartialName()</code> bereitgestellt, welches versucht wird, die Assembly zu laden. Wenn <code>partialname</code> vorhanden ist, werden alle anderen Parameter ignoriert.
<code>loc</code>	Die Locale, z.B. <code>en-US</code> . Die Standardeinstellung ist <code>neutral</code>

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>
```

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.

.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](#), 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) auf den erforderlichen XPath-Datentyp gecastet 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](#)):

```
<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( data:get_Month( date:new(2008, 4, 29)) )"
xmlns:date="clitype:System.DateTime" />
```

.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>
```

.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)"
      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. Dieses 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.

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.

<code>xs:string</code>	<code>StringValue</code> , <code>string</code>
<code>xs:boolean</code>	<code>BooleanValue</code> , <code>bool</code>
<code>xs:integer</code>	<code>IntegerValue</code> , <code>decimal</code> , <code>long</code> , <code>integer</code> , <code>short</code> , <code>byte</code> , <code>double</code> , <code>float</code>
<code>xs:float</code>	<code>FloatValue</code> , <code>float</code> , <code>double</code>
<code>xs:double</code>	<code>DoubleValue</code> , <code>double</code>
<code>xs:decimal</code>	<code>DecimalValue</code> , <code>decimal</code> , <code>double</code> , <code>float</code>

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 einige 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 Datentypes 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.

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.

1.5.3 MSXSL Scripts 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.:

```
"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.

1.5.4 Altova Erweiterungsfunktionen

Altova Erweiterungsfunktionen befinden sich im Namespace `http://www.altova.com/xslt-extension` und werden in diesem Abschnitt mit dem Präfix `altova:` gekennzeichnet, das an den oben angegebenen Namespace gebunden ist.

In der aktuellen Version Ihres Altova-Produkts werden die folgenden Erweiterungsfunktionen 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.

Allgemeine Funktionen

- [altova: evaluate\(\)](#)
- [altova: distinct-nodes\(\)](#)
- [altova: encode-for-rtf\(\)](#)
- [altova: xbrl-labels\(\)](#)
- [altova: xbrl-footnotes\(\)](#)
- [altova: generate-auto-number\(\)](#)
- [altova: reset-auto-number\(\)](#)
- [altova: get-temp-folder\(\)](#)

Diagrammfunktionen (Nur Enterprise und Reporting Edition)

Die Altova-Erweiterungsfunktionen für Diagramme werden nur in der Enterprise und der Reporting Edition von Altova Produkten unterstützt und gestatten die Erstellung von Diagrammen anhand von XML-Daten. Die Diagrammfunktionen wurden in zwei Gruppen unterteilt:

- Funktionen zum Generieren und Speichern von Diagrammen
- Funktionen zum Erstellen von Diagrammen

Ein dritter Abschnitt enthält eine Liste der Diagrammdaten-XML-Struktur, anhand der Diagramme generiert werden können und zum Schluss finden Sie im XSLT-Beispieldokument ein Beispiel, wie Diagrammfunktionen verwendet werden können, um anhand von XML-Daten Diagramme zu generieren.

Allgemeine Funktionen

In der aktuellen Version Ihres Altova-Produkts werden die folgenden Erweiterungsfunktionen 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.

- [altova: evaluate\(\)](#)
- [altova: distinct-nodes\(\)](#)
- [altova: encode-for-rtf\(\)](#)
- [altova: xbrl-labels\(\)](#)
- [altova: xbrl-footnotes\(\)](#)
- [altova: generate-auto-number\(\)](#)
- [altova: reset-auto-number\(\)](#)
- [altova: get-temp-folder\(\)](#)

altova: evaluate()

Die `altova: evaluate()` Funktion erhält einen XPath-Ausdruck als obligatorisches Argument, der als String übergeben wird, und gibt das Resultat des ausgewerteten Ausdrucks zurück.

```
altova: evaluate( XPathExp as xs: string)
```

Beispiel:

```
altova: evaluate( ' //Name[ 1] ' )
```

Beachten Sie im obigen Beispiel, dass der Ausdruck `//Name[1]` durch Einschließen in einfache Anführungszeichen als String übergeben wird. Die Funktion `altova:evaluate` gibt den Inhalt des ersten `Name`-Elements im Dokument zurück.

Die Funktion `altova:evaluate` kann zusätzliche (optionale) Argumente erhalten. Diese Argumente sind die Werte der einzelnen Variablen und haben die Namen `p1`, `p2`, `p3`... `pN`, die im XPath-Ausdruck verwendet werden können.

```
altova:evaluate( XPathExp as xs:string [, p1value ... pNvalue])
```

wobei

- die Variablennamen die Form `px` haben müssen, wobei `x` eine Ganzzahl ist
- die Reihenfolge der Argumente der Funktion vom zweiten Argument an der Reihenfolge der Variablen mit den Namen `p1` bis `pN` entspricht. Daher wird das zweite Argument der Wert der Variablen `p1`, das dritte Argument der der Variablen `p2`, usw.
- die Werte der Variablen den Typ `item*` haben müssen

Beispiel:

```
<xsl:variable name="xpath" select="$p3, $p2, $p1" />
<xsl:value-of select="altova:evaluate( $xpath, 10, 20, 'hi' )" />
Gibt "hi 20 10" zurück
```

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..

Im Folgenden ein weiteres Beispiel für die Verwendung:

```
<xsl:variable name="xpath" select="$p1" />
<xsl:value-of select="altova:evaluate( $xpath, //Name[1] )" />
Gibt den Wert des ersten Name Elements aus.

<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`)

Variablen können in der Erweiterungsfunktion `altova:evaluate()` wie im Beispiel unten gezeigt verwendet werden:

- 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

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.

altova:distinct-nodes()

Die `altova:distinct-nodes()` Funktion 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.

```
altova:distinct-nodes( $arg as node()* ) as node()*
```

altova:encode-for-rtf()

Die `altova:encode-for-rtf()` Funktion konvertiert den Input-String in Code für RTF.

```
altova:encode-for-rtf( $inputstr as xs:string?,
  $preserveallwhitespace as xs:boolean,
  $preservenewlines as xs:boolean) as xs:string
```

Whitespaces und neue Zeilen werden gemäß dem für die entsprechenden Parameter definierten Booleschen Wert beibehalten.

altova:xbrl-labels()

Die `altova:xbrl-labels()` Funktion 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.


```
altova:xbrl-labels( $name as xs:QName, $file as xs:string ) as node()*
```

altova:xbrl-footnotes()

Die `altova:footnotes()` Funktion erhält einen Node als Input-Argument und gibt die durch den Input-Node referenzierte Gruppe der XBRL-Fußnoten-Nodes zurück.

```
altova:footnotes( $arg as node() ) as node()*
```

altova:generate-auto-number(id als xs:string, start-with als xs:integer, increment als xs:integer, reset-on-change als xs:string)

Generiert eine Reihe von Zahlen mit der definierten ID. Die Anfangszahl (Ganzzahl) und die Inkrementierung werden definiert.

altova:reset-auto-number(id als xs:string)

Diese Funktion setzt die automatische Nummerierung der mit dem ID-Argument definierten automatischen Nummerierung zurück. Die Reihe wird auf den Anfangsganzzahlwert der Reihen zurückgesetzt (siehe `altova:generate-auto-number` oben).

altova:get-temp-folder als xs:string

Ruft den temp-Ordner ab.

2 Datentypen in XML-Schemas, die von DB generiert wurden

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.

- [MS Access](#)
- [MS SQL Server](#)
- [MySQL](#)
- [Oracle](#)
- [ODBC](#)
- [ADO](#)
- [Sybase](#)

2.1 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.

MS Access-Datentyp	XML-Schema-Datentyp
G U I D	xsd
char	xstring
varchar	xstring
m e m o	xstring
lt	xboolean
Num ber(single)	xfloat
Num ber(double)	xdouble
Decin al	xdecimal
Currency	xdecimal
Date/Tine	xdateTime
Num ber(Long Integer)	xinteger
Num ber(Integer)	xshort
Num ber(Byte)	xbyte
O L E O bject	xbase64Binary

2.2 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.

MS SQL Server-Datentyp	XML-Schema-Datentyp
uniqueidentifier	xsd
char	xstring
nchar	xstring
varchar	xstring
nvarchar	xstring
text	xstring
ntext	xstring
sysname	xstring
bit	xboolean
real	xfloat
float	xdouble
decimal	xdecimal
money	xdecimal
smallmoney	xdecimal
datetime	xdateTime
smalldatetime	xdateTime
binary	xsbase64Binary
varbinary	xsbase64Binary
image	xsbase64Binary
integer	xinteger
smallint	xshort
bigint	xlong
tinyint	xbyte

2.3 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.

My SQL-Datentyp	XML-Schema-Datentyp
char	xstring
varchar	xstring
text	xstring
tinytext	xstring
medium text	xstring
longtext	xstring
tinyint()	xboolean
float	xfloat
double	xdouble
decimal	xdecimal
datetime	xdateTime
blob	xbase64Binary
tinyblob	xbase64Binary
medium blob	xbase64Binary
longblob	xbase64Binary
smallint	xshort
bigint	xlong
tinyint	xbyte

2.4 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.

Oracle-Datentyp	XML-Schema-Datentyp
ROW ID	xsID
CHAR	xstring
NCHAR	xstring
VARCHAR2	xstring
NVARCHAR2	xstring
CLOB	xstring
NCLOB	xstring
NUMBER (with check constraint applied)*	xs:boolean
NUMBER	xdecimal
FLOAT	xdouble
DATE	xsdateTime
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 einem XML-Schema-Datentyp `xs:boolean` konvertiert. Dieser Mechanismus eignet sich zum Generieren eines Datentyps `xs:boolean` im generierten XML-Schema.

2.5 ODBC

Bei Generierung eines XML-Schemas anhand einer ODBC-Datenbank (DB) werden die ODBC-Datentypen wie unten in der Tabelle gezeigt in XML-Schema-Datentypen konvertiert.

ODBC-Datentyp	XML-Schema-Datentyp
SQL_GUID	xsd
SQL_CHAR	xstring
SQL_VARCHAR	xstring
SQL_LONGVARCHAR	xstring
SQL_BIT	xboolean
SQL_REAL	xfloat
SQL_DOUBLE	xdouble
SQL_DECIMAL	xdecimal
SQL_TIMESTAMP	xdateTime
SQL_DATE	xdate
SQL_BINARY	xbase64Binary
SQL_VARBINARY	xbase64Binary
SQL_LONGVARBINARY	xbase64Binary
SQL_INTEGER	xinteger
SQL_SMALLINT	xshort
SQL_BIGINT	xlong
SQL_TINYINT	xbyte

2.6 ADO

Bei Generierung eines XML-Schemas anhand einer ADO-Datenbank (DB) werden die ADO DB-Datentypen wie unten in der Tabelle gezeigt in XML-Schema-Datentypen konvertiert.

ADO-Datentyp	XML-Schema-Datentyp
adG U ID	xsd
adC har	xstring
adW C har	xstring
adVarC har	xstring
adW VarC har	xstring
adLongVarC har	xstring
adW LongVarC har	xstring
adVarW C har	xstring
adBoo lean	xboolean
adS inge	xfloat
adD ouble	xdouble
adN um eric	xdecimal
adC unency	xdecimal
adD B T im eS tam p	xdateTime
adD ate	xdate
adB inary	xbase64Binary
adVarB inary	xbase64Binary
adLongVarB inary	xbase64Binary
adI nteger	xinteger
adU nsignedI nt	xunsignedInt
adS m allI nt	xshort
adU nsignedS m allI nt	xunsignedShort
adB igI nt	xlong
adU nsignedB igI nt	xunsignedLong
adT inyI nt	xbyte
adU nsignedT inyI nt	xunsignedByte

2.7 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	xstring
nchar	xstring
varchar	xstring
nvarchar	xstring
text	xstring
sysname-varchar(30)	xstring
bit	xboolean
real	xfloat
float	xfloat
double	xdouble
decimal	xdecimal
money	xdecimal
smallmoney	xdecimal
datetime	xdatetime
smalldatetime	xdatetime
timestamp	xdatetime
binary<=255	xbase64Binary
varbinary<=255	xbase64Binary
image	xbase64Binary
integer	xinteger
smallint	xshort
tinyint	xbyte

3 Datentypen in DB, die von XML-Schema 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](#)
- [MS SQL Server](#)
- [MySQL](#)
- [Oracle](#)

3.1 MS Access

Bei Generierung einer MS Access-Datenbank (DB) anhand eines XML-Schemas werden XML-Schema-Datentypen wie unten in der Tabelle gezeigt in die MS Access Datentypen konvertiert.

XML-Schema-Datentyp	MS Access-Datentyp
xsD	G U I D
xsstring	If no facets varchar(255)
	Size = either length or maxLength
	If size <= 255 varchar(n)
	else memo
xsnormalizedString	Same as xsstring
xsToken	Same as xsstring
xs:Name	Same as xsstring
xs:NCName	Same as xsstring
xs:anyURI	Same as xsstring
xs:QName	Same as xsstring
xs:NOTATION	Same as xsstring
xsboolean	bit
xsfloat	Number(single)
xsdouble	Number(double)
xsdecimal	Decimal
xsduration	Date/Time
xsdateTime	Date/Time
xsTime	Date/Time
xsdate	Date/Time
xs:YearMonth	Date/Time
xs:Year	Date/Time
xs:MonthDay	Date/Time
xs:Day	Date/Time
xs:Month	Date/Time
xs:hexBinary	If no facets varbinary(255)
	Size = either length or maxLength
	If size <= 8000 varbinary

	~~else in age (OLE Object)~~
xsbase64Binary	Same as ~~xshexBinary~~
xsinteger	~~Number(Long Integer)~~
xsint	~~Number(Long Integer)~~
xsnegativeInteger	~~Number(Long Integer); value constant~~
xsgostiveInteger	~~Number(Long Integer); value constant~~
xsnonNegativeInteger	~~Number(Long Integer); value constant~~
xsnonPositiveInteger	~~Number(Long Integer); value constant~~
xsunsignedInt	~~Number(Long Integer)~~
xshort	~~-noequivalent-~~
xsunsignedShort	~~-noequivalent-~~
xslong	~~-noequivalent-~~
xsunsignedLong	~~-noequivalent-~~
xsbyte	~~Number(Byte)~~
xsunsignedByte	~~Number(Byte)~~

3.2 MS SQL Server

Bei Generierung eines XML-Schemas anhand einer MS SQL Server-Datenbank (DB) werden die MS SQL Server-Datentypen wie unten in der Tabelle gezeigt in XML-Schema-Datentypen konvertiert.

XML-Schema-Datentyp	MS SQL Server-Datentyp
D	uniqueidentifier
xs:string	if nvarchar
	{ if UNICODE nvarchar(255)
	else varchar(255)}
	else
	{ if UNICODE
	(Size=either length or maxLength)
	if Size <= 4000
	if FacetLength is Set then nChar
	else nVarChar
	if Size <= 1073741823 then nText
	else
	{ if NON-UNICODE
	(Size=either length or maxLength)
	if Size <= 8000
	if FacetLength is Set 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

xsdouble	float
xdecimal	decimal
xduration	datetime
xdateTime	datetime
xtime	datetime
xdate	datetime
xsgYearMonth	datetime
xsgYear	datetime
xsgMonthDay	datetime
xsgDay	datetime
xsgMonth	datetime
xshexBinary	if (facet binary(255))
	(Size = either length or max length
	if Size <= 8000
	if Facet length is set then binary
	else varbinary
	if Size <= 2147483647 then image
xbase64Binary	Same as xshexBinary
xinteger	int
xint	int
xnegativeInteger	int (constraint (-2147483648, -1))
xpositiveInteger	int (constraint (1, 2147483647))
xnonNegativeInteger	int (constraint (0, 2147483647))
xnonPositiveInteger	int (constraint (-2147483648, 0))
xunsignedInt	int (additional constraints)
xshort	smallint
xunsignedShort	smallint (additional constraints)
xlong	bigint
xunsignedLong	bigint (additional constraints)
xbyte	tinyint
xunsignedByte	tinyint (additional constraints)

3.3 MySQL

Bei Generierung eines XML-Schemas anhand einer MySQL Datenbank (DB) werden die MySQL DB-Datentypen wie unten in der Tabelle gezeigt in XML-Schema-Datentypen konvertiert.

XML-Schema-Datentyp	MySQL-Datentyp
xsD	varchar(255)
xsstring	If no facets then varchar(255)
	else if facet length is set and <= 255 then char
	else if facet maxLength is set and <= 255 then varchar
	else if maxLength is set and <= 65545 then text
	else if maxLength is set and <= 16777215 then medium text
	else if maxLength is set and <= 429496295 then longtext
xsnormalizedString	<i>Same as</i> xsstring
xsToken	<i>Same as</i> xsstring
xsName	<i>Same as</i> xsstring
xsNCName	<i>Same as</i> xsstring
xsanyURI	<i>Same as</i> xsstring
xsQName	<i>Same as</i> xsstring
xsNOTATION	<i>Same as</i> xsstring
xsboolean	tinyint(1)
xsfloat	float
xsdouble	double
xsdecimal	decimal
xsduration	timestamp
xsdateTime	datetime
xsTime	time
xsdate	date
xs:gYearMonth	timestamp(4)
xs:gYear	year(4)
xs:gMonthDay	timestamp(8); constraints to check month, day

xsgDay	tinyint unsigned(8); constraints to check day
xsgMonth	tinyint unsigned(8); constraints to check month
xshexBinary	if no facets then blob(255)
	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
xsbase64Binary	Same as xshexBinary
xsinteger	Integer
xsint	int
xsnegativeInteger	Integer(constrained to (-2 ³¹))
xspositiveInteger	Integer(constrained to (1, 2 ³¹))
xsnonNegativeInteger	Integer(constrained to (0, 2 ³¹))
xsnonPositiveInteger	Integer(constrained to (-2 ³¹ , 0))
xsunsignedInt	int(additional constraints)
xshort	Smallint
xsunsignedShort	Smallint(additional constraints)
xlong	Bigint
xsunsignedLong	Bigint(additional constraints)
xsbyte	Tinyint
xsunsignedByte	Tinyint(additional constraints)

3.4 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
<code>xsID</code>	<code>ROW ID</code>
<code>xs:string</code>	<code>if facets</code>
	<code>if UNICODE then NVARCHAR2 (255)</code>
	<code>else VARCHAR2 (255)</code>
	<code>else if UNICODE</code>
	<code>(Size=either length or maxLength)</code>
	<code>if Size <= 2000 then NCHAR</code>
	<code>if Size <= 4000 then NVARCHAR2</code>
	<code>if Size <= 4 G igits then NCLOB</code>
	<code>else if NON-UNICODE</code>
	<code>(Size=either length or maxLength)</code>
	<code>if Size <= 2000 then CHAR</code>
	<code>if Size <= 4000 then VARCHAR2</code>
	<code>if Size <= 4 G igits then CLOB</code>
<code>xs:namedString</code>	Same as <code>xs:string</code>
<code>xs:token</code>	Same as <code>xs:string</code>
<code>xs:Name</code>	Same as <code>xs:string</code>
<code>xs:NCName</code>	Same as <code>xs:string</code>
<code>xs:anyURI</code>	Same as <code>xs:string</code>
<code>xs:QName</code>	Same as <code>xs:string</code>
<code>xs:NOTATION</code>	Same as <code>xs:string</code>
<code>xs:boolean</code>	<code>NUMBER with constant Boolean</code>
<code>xs:float</code>	<code>FLOAT</code>
<code>xs:double</code>	<code>FLOAT</code>
<code>xs:decimal</code>	<code>NUMBER</code>
<code>xs:dateTime</code>	<code>TIMESTAMP</code>
<code>xs:dateTime</code>	<code>TIMESTAMP</code>
<code>xs:time</code>	<code>DATE</code>
<code>xs:date</code>	<code>DATE</code>

xsgYearMonth	INTERVAL YEAR TO MONTH
xsgYear	D A T E
xsgMonthDay	D A T E
xsgDay	D A T E
xsgMonth	D A T E
xshexBinary	if no facets then RAW (255)
	(Size = either length or maxLength)
	if Size <= 2000 then RAW (X)
	else Size <= 2 Gigabytes then LONG RAW (X)
	if Size <= 4 Gigabytes then BLOB (X)
xsbase64Binary	B L O B
xsinteger	N U M B E R
xsint	N U M B E R
xsnegativeInteger	NUMBER (constrained to { $-\infty, -2, -1$ })
xspositiveInteger	NUMBER (constrained to {1, 2, ...})
xsnonNegativeInteger	NUMBER (constrained to {0, 1, 2, ...})
xsnonPositiveInteger	NUMBER (constrained to { $-\infty, -2, -1, 0$ })
xsunsignedInt	NUMBER (additional constraints)
xshort	N U M B E R
xsunsignedShort	NUMBER (additional constraints)
xslong	N U M B E R
xsunsignedLong	NUMBER (additional constraints)
xsbyte	B L O B
xsunsignedByte	BLOB (additional constraints)

4 Technische Daten

Dieses Kapitel enthält nützliche Hintergrundinformationen zu den technischen Aspekten Ihrer Software. Es ist in die folgenden Abschnitte gegliedert:

- [OS- und Arbeitsspeichieranforderungen](#)
- [Altova XML Parser](#)
- [Altova XSLT- und XQuery-Prozessor](#)
- [Unicode-Unterstützung](#)
- [Internet-Verwendung](#)

4.1 OS- und Arbeitsspeichieranforderungen

Betriebssystem

Die Altova-Software-Applikationen sind:

- 32-Bit Windows-Applikationen für Windows XP Windows Server 2003 und 2008, Windows Vista und Windows 7 oder
- 64-Bit-Applikationen für Windows Vista und Windows 7

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 geparkt und analysiert werden kann und die Anzeige- und Bearbeitungsgeschwindigkeit während der normalen Arbeit verbessert wird.

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.

4.2 Altova XML Parser

Wenn Sie ein XML-Dokument öffnen, verwendet die Applikation den integrierten Validierungsparser (den Altova XML Parser), um das Dokument auf Wohlgeformtheit zu prüfen, es gegen ein Schema zu validieren (falls eines angegeben wurde) und Baumstrukturen und Infosets zu erstellen. Der Altova XML Parser dient auch dazu, beim Editieren von Dokumenten intelligente Eingabehilfen zur Verfügung zu stellen und etwaige Validierungsfehler dynamisch anzuzeigen.

Im integrierten Altova XML Parser ist die Final Recommendation der W3C XML Schema Spezifikation implementiert. Neue Entwicklungen, die von der XML Schema-Arbeitsgruppe empfohlen werden, werden ständig in den Altova Parser integriert, sodass Ihnen mit Altova-Produkten immer eine Entwicklungsumgebung auf dem neuesten Stand der Technik zur Verfügung steht.

4.3 Altova XSLT- und XQuery-Prozessor

Die Altova Produkte arbeiten mit dem Altova XSLT 1.0-Prozessor, dem Altova XSLT 2.0-Prozessor und dem Altova XQuery-Prozessor. Nähere Informationen zu diesen Prozessoren finden Sie im Anhang 1 zu dieser Dokumentation im Abschnitt über Prozessoren.

Diese drei Prozessoren stehen auch als eigenständige Applikationen in Form des AltovaXML-Pakets zum kostenlosen Download von der [Altova Website](#) zur Verfügung. Die Dokumentation zur Verwendung der Prozessoren wird mit dem AltovaXML-Paket mitgeliefert.

4.4 Unicode-Unterstützung

Unicode ist ein 16-Bit-Zeichensatzstandard (der auf 32-Bit erweiterbar ist), der vom [Unicode-Konsortium](#) definiert wurde und der eine eindeutige Zahl für jedes Zeichen zur Verfügung stellt, und zwar:

- plattformunabhängig
- programmunabhängig
- unabhängig von der Sprache

Im Prinzip arbeiten Computer nur mit Zahlen. Sie speichern Buchstaben und andere Zeichen unter einer Zahl, die sie diesen zuweisen. Vor der Entwicklung von Unicode gab es hunderte von verschiedenen Codiersystemen für die Zuweisung dieser Zahlen. Es gab keine einzige Codierung, die Platz für genügend Zeichen bot: So werden z.B. in der Europäischen Union mehrere verschiedene Codierungen benötigt, um alle ihre Sprachen abzudecken. Selbst für eine einzige Sprache wie z.B. Englisch reichte eine einzelne Codierung nicht aus, um alle gebräuchlichen Buchstaben, Satzzeichen und technischen Symbole abzudecken.

Diese Codierungssysteme standen früher miteinander in Konflikt, d.h. zwei Codierungen verwendeten dieselbe Zahl für zwei verschiedene Zeichen oder verschiedene Zahlen für dasselbe Zeichen. Jeder Computer (v.a. Server) muss viele verschiedene Codierungen unterstützen. Wenn aber Daten zwischen verschiedenen Codierungen oder Plattformen ausgetauscht werden, besteht immer das Risiko, dass Fehler entstehen.

Unicode ist die Lösung für diese Probleme!

Unicode enthält eine eindeutige Zahl für jedes Zeichen, egal für welche Plattform, welches Programm oder welche Sprache. Der Unicode-Standard wurde von branchenführenden Unternehmen wie Apple, HP, IBM, JustSystems, Microsoft, Oracle, SAP, Sun, Base und vielen anderen übernommen.

Unicode ist für moderne Standards wie XML, Java, ECMAScript (JavaScript), LDAP, CORBA 3.0, WML, usw. erforderlich und ist das offizielle Verfahren zum Implementieren von ISO/IEC 10646. Unicode wird von vielen Betriebssystemen, allen modernen Browsern und vielen anderen Produkten unterstützt. Die Entwicklung des Unicode-Standards und die Verfügbarkeit von Tools, die diesen unterstützen, gehören zu den wichtigsten globalen Software Technologie-Trends in letzter Zeit.

Die Integration von Unicode in Client-Server oder Multi-Tier-Applikationen und Websites bietet beträchtliche Kostenvorteile im Vergleich zur Verwendung herkömmlicher Zeichensätze. Dank Unicode kann ein einziges Software-Produkt oder eine einzige Website für mehrere Plattformen, Sprachen und Länder konzipiert werden, ohne dass es neu entwickelt werden muss. Daten können dadurch durch viele verschiedene Systeme übertragen werden, ohne dass dabei Fehler auftreten.

4.4.1 Windows XP

Die XML-Produkte von Altova bieten vollständige Unicode-Unterstützung. Um ein XML-Dokument von einem System mit nicht lateinischer Zeichensatzcodierung bearbeiten zu können, benötigen Sie allerdings auch 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. Aus diesem Grund kann es vorkommen, dass ein XML-Dokument "nicht druckbare" Zeichen enthält, da die gewählte Schriftart die erforderlichen Glyphen nicht enthält.

Es ist daher manchmal nützlich, wenn man eine Schriftart hat, die den gesamten Unicode-Bereich abdeckt - v.a. wenn Sie XML-Dokumente aus der ganzen Welt editieren.

Die größte uns bekannte Universalschriftart ist "Arial Unicode MS", eine Schriftart, die von Agfa Monotype für Microsoft entwickelt wurde. Diese Schriftart enthält mehr als 50.000 Glyphen und deckt das gesamte im Unicode 2.1.-Standard definierte Zeichenspektrum ab. Die Schriftart ist in Microsoft Office 2000 enthalten und beansprucht 23 MB.

Wir empfehlen Ihnen dringend, diese Schriftart auf Ihrem System zu installieren und mit der Applikation zu verwenden, wenn Sie öfter Dokumente bearbeiten, die in unterschiedlichen Schriftsystemen verfasst wurden. Diese Schriftart wird bei Verwendung der Standardinstallation von Microsoft Office nicht installiert, Sie können jedoch die Option "Benutzerdefinierte Installation" wählen, um sie zu installieren.

Im Ordner "Examples" finden Sie außerdem eine neue XHTML-Datei mit dem Namen `Unicode-UTF8.html`, die den Satz "Wenn die Welt miteinander spricht, spricht sie Unicode" in vielen verschiedenen Sprachen und Schriftsystemen enthält ("When the world wants to talk, it speaks Unicode") (世界的に話すなら、Unicode です) - Diese Zeile stammt aus der 10. Unicode-Konferenz von 1997 und ist ein schönes Beispiel für die Bedeutung von Unicode für den XML-Standard. Wenn Sie diese 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.

4.4.2 Rechts nach Links-Schreibrichtung

Beachten Sie bitte, dass selbst unter Windows NT 4.0 Texte aus Schriftsystemen mit einer Schreibrichtung von rechts nach links (z.B. Hebräisch oder Arabisch) nur in den Ländern korrekt wiedergegeben werden, in denen solche Schriftsysteme tatsächlich verwendet werden. Dies liegt daran, dass nur die hebräische und arabische Version von Windows NT die Wiedergabe und Bearbeitung von Text in Rechts-links-Schreibweise auf Betriebssystemebene unterstützen.

4.5 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.
- Falls Sie den URL-Modus des Dialogfelds "Öffnen" verwenden, um ein Dokument direkt über eine URL zu öffnen (**Datei | Öffnen | Zu URL wechseln**), wird dieses Dokument über eine HTTP-Verbindung (Port 80) aufgerufen. (*Diese Funktionalität steht in XMLSpy und Authentic Desktop zur Verfügung.*)
- 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 es ebenfalls über eine HTTP-Verbindung (Port 80) aufgerufen, sobald Sie das XML-Dokument validieren. Dies kann auch automatisch beim Öffnen eines Dokuments vorkommen, wenn Sie die Applikation so konfiguriert haben, dass Dateien beim Öffnen automatisch validiert werden (Register "Datei" des Dialogfelds **Extras | Optionen**).
- Wenn Sie den Befehl "Als Mail senden..." (**Datei | Als Mail senden**) 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 in diesem Handbuch und der Altova-Software-Lizenzvereinbarung beschriebenen Software-Aktivierung und beim Live-Update..

5 Lizenzinformationen

Dieser Anhang enthält die folgenden Informationen:

- Informationen über den [Vertrieb dieses Software-Produkts](#)
- Informationen im Zusammenhang mit dem [intellektuellen Eigentum](#) dieses Software-Produkts
- Die [Endbenutzer-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.

5.1 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 testen, bevor Sie sich zu einem Kauf entscheiden.
- Wenn Sie sich entschieden haben, die Software zu kaufen, können Sie Ihre [Bestellung online abgeben](#) und das Produkt steht Ihnen in Minutenschnelle im vollen Lizenzumfang zur Verfügung.
- Sie erhalten immer die neueste Version unserer Software
- Die Software enthält ein umfassendes Online-Hilfesystem. 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 weiterverwenden möchten, müssen Sie eine [Software-Produktlizenz](#) erwerben, die Sie in Form eines Keycodes erhalten, der in das Dialogfeld "Software-Aktivierung" eingegeben werden muss, um das Produkt zu aktivieren. Sie können Ihre Lizenz online in unserem Web-Shop unter [Altova Shop](#) 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, sollten 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.

Nähere Informationen finden Sie in der [Altova Software-Lizenzvereinbarung](#) am Ende dieses Handbuchs.

5.2 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 Computernamen 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 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.

Mehrfachlizenz

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 Website](http://www.iana.org/) (<http://www.iana.org/>)) 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 zwischen verschiedenen Gruppen in Ihrem Unternehmen natürlich blockieren, solange Sie mit anderen Mitteln sicherstellen können, dass Ihre Lizenzvereinbarung eingehalten wird.

Sie werden außerdem auch feststellen, dass Ihr Altova Produkt viele nützliche Funktionen enthält, die Ihre Internet-Verbindung verwenden, diese stehen jedoch nicht mit der Lizenzüberwachungsfunktion in Zusammenhang.

5.3 Rechte am geistigen Eigentum

Die Software und alle Kopien, deren Anfertigung Ihnen von Altova gestattet ist, sind das geistige Eigentum von Altova und dessen Lieferanten und stehen in deren Besitz. Struktur, Organisation und Code der Software stellen wertvolle Betriebsgeheimnisse und vertrauliche Informationen von Altova und dessen Lieferanten dar. Die Software ist durch gesetzliche Bestimmungen urheberrechtlich geschützt. Diese gesetzlichen Bestimmungen beinhalten (ohne darauf beschränkt zu sein) das Urheberrecht der Vereinigten Staaten, internationale Verträge und das in den Ländern, in denen die Software genutzt wird, geltende Recht. Altova behält sich das Eigentum an allen Patenten, Urheberrechten, Branchengeheimnissen, Warenzeichen und sonstigen geistigen Eigentumsrechten, die hinsichtlich der Software bestehen, vorbehält. Das Eigentumsrecht von Altova erstreckt sich auch auf alle Bilder, Fotografien, Animationen, Videos, Audioaufzeichnungen, Musikstücke, Texte und „Applets“, die Teil der Software sind, und alle dazugehörigen Unterlagen in gedruckter Form. Mitteilungen über geltend gemachte Verstöße gegen geistige Eigentumsrechte sind an den Copyright Agenten von Altova zu richten (nähere Angaben dazu finden Sie auf der Website von Altova).

Altova Software enthält Software-Komponenten Dritter, die ebenfalls urheberrechtlich geschützt sind. Unter anderem sind diese Urheberrechte ausführlich unter http://www.altova.com/legal_3rdparty.html aufgelistet.

Alle anderen Namen oder Warenzeichen sind das Eigentum ihrer jeweiligen Inhaber.

5.4 Altova Endbenutzer-Lizenzvereinbarung

DIES IST EIN RECHTSGÜLTIGES DOKUMENT -- BITTE BEWAHREN SIE ES SORGFÄLTIG AUF

ALTOVA® ENDBENUTZER-LIZENZVERTRAG

Lizenzgeber:
Altova GmbH
Rudolfsplatz 13a/9
A-1010 Wien
Österreich

Wichtig – Bitte sorgfältig lesen. Benutzerinformation:

Dieser Endbenutzer-Lizenzvertrag („Lizenzvertrag“) ist ein rechtsgültiger Vertrag, der zwischen Ihnen und Altova GmbH („Altova“) abgeschlossen wurde. Bevor Sie die von Altova bereitgestellte Software („Software“) sowie die dazugehörige Dokumentation (u.a. Unterlagen in gedruckter Form, Online-Dateien oder Dokumentation in elektronischer Form („Dokumentation“) verwenden, lesen Sie bitte dieses Dokument sorgfältig durch. Indem Sie auf die untenstehenden Schaltflächen „Ich akzeptiere“ und „Weiter“ klicken, oder indem Sie die Software installieren oder auf sonstige Weise nutzen, stimmen Sie zu, dass Sie die Bedingungen dieses Lizenzvertrags sowie die Datenschutzbestimmungen („Datenschutzbestimmungen“) von Altova, inklusive, jedoch nicht ausschließlich, die nachfolgend festgelegten Garantiausschlüsse, Haftungsbeschränkungen sowie Datennutzungs- und Kündigungsregelungen als rechtsverbindlich anerkennen, auch wenn Sie die Software nicht käuflich erwerben. Sie stimmen zu, dass diese Vereinbarung in demselben Umfang gültig und durchsetzbar ist, wie ein von Ihnen persönlich unterzeichneter Vertrag. Wenn Sie den Bedingungen dieses Lizenzvertrags nicht zustimmen, sind Sie nicht zur Verwendung der Software berechtigt und müssen alle heruntergeladenen Kopien, die sich in Ihrem Besitz oder unter Ihrer Kontrolle befinden, vernichten. Während des Installationsvorganges können Sie eine Kopie dieses Lizenzvertrages zum Zeitpunkt Ihrer Zustimmung ausdrucken. Darüber hinaus können Sie unsere Website unter <http://www.altova.com/de/eula> besuchen, um eine Kopie dieses Lizenzvertrags herunterzuladen und auszudrucken. Die Datenschutzbestimmungen können unter <http://www.altova.com/de/privacy> eingesehen werden.

1. SOFTWARELIZENZ

(a) Lizenzgewährung.

(i) Sofern Sie diesem Lizenzvertrag zugestimmt haben, gewährt Ihnen Altova eine nicht ausschließliche und nicht übertragbare (mit Ausnahme der nachfolgenden Bestimmungen) Lizenz, ohne das Recht zur Vergabe von Sublizenzen, zur Installation und Verwendung der Software auf einem kompatiblen Einzelplatzrechner oder einer Workstation, wobei die maximale Zahl an Computern, die im Lizenzumfang enthalten sind, nicht überschritten werden darf. Vorbehaltlich der Beschränkungen gemäß Abschnitt 1 (c), dürfen Sie eine Kopie der Software auf mehr als einem Ihrer kompatiblen Einzelplatzrechnern oder Workstations installieren und verwenden, wenn Sie eine Named-User-Lizenz (Lizenz zur benutzerdefinierten Nutzung) erworben haben. Vorbehaltlich der Beschränkungen gemäß den Abschnitten 1(d) und 1(e) darf die Software gleichzeitig im Netzwerk verwendet werden. Die maximale Zahl an Computern und/oder Benutzern sowie die Art der Lizenz, d.h. Einzelplatzlizenz, Named-User-Lizenz oder Parallellizenz, wird beim Kauf der Software festgelegt und genau angegeben. Einzelplatzlizenzen sind als fixe Lizenzen und nicht als Parallellizenzen zu verwenden, d.h. es ist nicht erlaubt, die Lizenz für einen Benutzer zu deinstallieren, um die Lizenz für einen anderen Benutzer zu installieren und diese Lizenz anschließend wieder zu deinstallieren und wieder für den ursprünglichen Benutzer zu

installieren. Einzelplatzlizenzen sollten statisch sein. Dessen ungeachtet sind permanente Uminstallationen einer Einzelplatzlizenz gestattet (wenn z.B. ein Mitarbeiter nicht mehr für das Unternehmen arbeitet und sein Rechner nicht mehr benötigt wird). Während des Evaluierungszeitraums (wie nachfolgend definiert) darf die Software nur von einem Benutzer und auf nur einem (1) Einzelplatzrechner oder einer Workstation installiert und genutzt werden. Wenn Sie die Lizenz im Rahmen eines Pakets von Altova-Softwareprodukten („Produktpaket“) erworben haben, von denen Sie nicht jedes Produkt einzeln installiert haben, gelten die Bestimmungen dieses Lizenzvertrags für die Verwendung der gesamten Software, die im Lieferumfang des Produktpakets enthalten ist.

(ii) Wenn Sie eine Lizenz für SchemaAgent erworben haben, dann gelten die Bestimmungen dieses Lizenzvertrags zur Nutzung der Software für den darin inkludierten SchemaAgent-Server („SchemaAgent-Server“). Dabei dürfen Sie den SchemaAgent-Server jedoch ausschließlich in Verbindung mit der Altova-Software und nur für die in der dazugehörigen Dokumentation beschriebenen Zwecke verwenden.

(iii) Wenn Sie eine Lizenz für Software erworben haben, die es dem Benutzer ermöglicht den Quellcode zu generieren, dann sind Sie durch die Ihnen im Rahmen dieser Lizenz erteilten Genehmigung zur Installation und Verwendung der Software auch berechtigt, den Quellcode zu generieren, und zwar entweder (i) anhand von Modulen der Altova Library, die in der Software inkludiert sind (der auf diese Weise generierte Code wird als „beschränkter Quellcode“ bezeichnet), oder (ii) anhand Ihrer eigenen Schemata oder Mappings (der anhand Ihres eigenen Quellmaterials generierte Code wird als „unbeschränkter Quellcode“ bezeichnet). Zusätzlich zu den durch diesen Lizenzvertrag eingeräumten Rechten gewährt Ihnen Altova eine nicht ausschließliche, nicht übertragbare und eingeschränkte Lizenz, den gesamten generierten Code (bestehend aus dem beschränkten und unbeschränkten Quellcode) in eine ausführbare Objektcodeform zu übersetzen und diese zu verwenden, zu kopieren, weiterzugeben oder zu lizenzieren. Sie sind nur dann befugt, den beschränkten Quellcode ab- oder weiterzugeben, weiterzulizenzieren, zu verkaufen oder an Dritte zu übertragen, wenn die betreffende Partei bereits in Form eines gesonderten Lizenzvertrags mit Altova zur Verwendung des beschränkten Quellcodes berechtigt ist. Ungeachtet anderslautender Bestimmungen in diesem Lizenzvertrag dürfen Sie die Altova-Library-Module oder den beschränkten Quellcode (oder sonstiges geistiges Eigentum von Altova enthalten in oder in Verbindung mit den Altova-Library-Modulen oder dem beschränkten Quellcode) nicht in solcher Art und Weise weitergeben, in andere Software einbetten, mit anderer Software verbinden oder auf andere Weise verwenden, die den beschränkten Quellcode den Bestimmungen des „Copylefts“, der freien Software oder einer Open Source Lizenz unterwerfen würde, welche die Offenlegung des beschränkten Quellcodes oder der Altova-Library-Module erforderlich machen würde. Altova behält sich alle übrigen Rechte an der Software vor. Bei den Bestandteilen von UModel, für die ein Reverse Engineering Ihres eigenen oder eines anderen Quellcodes, den Sie rechtmäßig erworben haben, zulässig ist, stellt eine solche Verwendung durch Sie keinen Verstoß gegen diesen Lizenzvertrag dar. Soweit dies nicht in Abschnitt 1(h) ausdrücklich gestattet wird, ist ein Reverse Engineering der Software explizit untersagt.

(iv) Falls beschränkter Quellcode in Form ausführbaren Objektcodes inkludiert wird, verpflichten Sie sich, folgende Informationen (1) in den Willkommensbildschirm, oder falls nicht vorhanden, in einen oder mehrere der durch den Endbenutzer jederzeit aufrufbaren Bildschirme und (2) - in elektronischer und/oder gedruckter Form in die Dokumentation zu inkludieren: "Teile dieses Programms wurden unter Verwendung von Altova® [Name der Altova Software, z.B. MapForce® 2011] entwickelt und enthalten Bibliotheken, die im Eigentum der Altova GmbH (Copyright © 2007-2011 Altova GmbH (www.altova.com)) stehen."

(b) Verwendung auf einem Server zur Installation und Verwendung von SchemaAgent. Sie sind berechtigt, ein (1) Exemplar der Software auf einem Dateiserver ausschließlich innerhalb Ihres internen Netzwerks zu installieren und von dort auf die maximale Zahl von betrieblich genutzten Computern innerhalb dieses Netzwerks, die in der Lizenz angegeben ist, herunterzuladen und zu installieren. Wenn Sie eine Lizenz für SchemaAgent

erworben haben, können Sie SchemaAgent auf jedem Server oder jeder Workstation installieren und in Verbindung mit Ihrer Software verwenden. Jede sonstige Netzwerknutzung ist unzulässig; dies umfasst insbesondere die direkte Verwendung oder die Nutzung der Software über Befehle, Daten oder Anweisungen von oder an einen Computer, der nicht Teil Ihres internen Netzwerks ist, die Verwendung für Internet- oder Webhosting-Dienste sowie die Verwendung durch einen Benutzer, der nicht unter einer gültigen Lizenz von Altova zur Verwendung dieses Exemplars der Software berechtigt ist.

(c) Benutzerdefinierte Nutzung (Named-Use). Wenn Sie eine Lizenz der „benutzerdefinierten Version“ der Software erworben haben, sind Sie berechtigt, die Software auf bis zu fünf (5) kompatiblen Einzelplatzrechnern oder Workstations zu installieren, auf denen Sie der primäre Nutzer sind. Dadurch können Sie von einem Computer zu einem anderen je nach Bedarf wechseln, vorausgesetzt, dass die Software von Ihnen als benutzerdefinierter Nutzer jeweils nur ein (1) Mal gleichzeitig verwendet wird. Wenn Sie mehrere benutzerdefinierte Lizenzen erworben haben, erhält jeder individuell definierte Nutzer einen gesonderten Lizenzschlüssel.

(d) Parallellizenz für die Verwendung im selben physischen Netzwerk oder am selben Unternehmensstandort. Wenn Sie eine Parallellizenz Software-Version erworben haben, können Sie die Software auf mehreren kompatiblen und betrieblich genutzten Computern installieren, jedoch höchstens zehnmal (10-mal) die maximale Zahl der Benutzer, vorausgesetzt, dass nur die maximale Zahl von Benutzern die Software wirklich gleichzeitig nutzen sowie weitere Voraussetzung, dass die Computer, auf welchen die Software installiert ist, in dasselbe physische Computer Netzwerk eingebunden sind. Die maximale Zahl der Parallelbenutzer wird beim Kauf der Software festgelegt. Sofern die Parallelbenutzer Version der Software an mehr als einem Standort oder in mehr als einem Netzwerk verwendet werden soll, sind für jedes physisch getrennte Netzwerk sowie für jeden physisch getrennten Standort separate Parallelbenutzer Lizenzen erforderlich, wobei die Einschränkungen hinsichtlich der maximalen Zahl der Benutzer und der Anzahl der Nutzer, die die Software benutzen, anwendbar sind. Für den Fall, dass ein Computer nicht in dasselbe physische Netzwerk eingebunden ist, ist für diesen eine lokal installierte Benutzerlizenz oder eine Parallellizenz für die Verwendung in einer virtuellen Umgebung erforderlich. Be- und Einschränkungen hinsichtlich der Nutzung von Parallelbenutzer Lizenzen auf Heimcomputern sind in Abschnitt 1(g) geregelt.

(e) Parallellizenz für die Verwendung in einer virtuellen Umgebung. Wenn Sie eine Parallellizenz erworben haben, sind Sie berechtigt, ein Exemplar der Software auf einem Terminalserver (Microsoft Terminal Server, Citrix Metaframe, usw.), einem Anwendungsvirtualisierungsserver (Microsoft App-V, Citrix XenApp, VMWare ThinApp, usw.) oder einer virtuellen Maschine in Ihrem Intranet zu installieren, damit einzelne Benutzer in Ihrer Organisation von einem anderen Computer über einen Terminalserver, einen Anwendungsvirtualisierungsserver oder eine virtuelle Maschine auf die Software zugreifen und diese nutzen können. Jedoch darf die Zahl der Benutzer, die über einen solchen Netzwerk- oder Terminalserver bzw. eine virtuelle Maschine zum selben Zeitpunkt auf die Software zugreifen, die maximale Zahl der im Lizenzumfang enthaltenen User nicht überschreiten. Außerdem darf die Gesamtanzahl der Benutzer, die die Software über den Terminal Server, den Anwendungsvirtualisierungsserver oder eine virtuelle Maschine verwenden, die sechs (6)-fache Anzahl der lizenzierten Benutzer nicht übersteigen. Folglich gelten die in Abschnitt 1(d) festgelegten Einschränkungen betreffend die Anzahl der Installationen und die Voraussetzung, dass die Verwendung über dasselbe physische Netzwerk erfolgen muss, nicht für Terminal Server-, Anwendungsvirtualisierungsserver-Umgebungen oder Umgebungen auf virtuellen Maschinen. Sie müssen in einer virtuellen Umgebung mit Hilfe geeigneter Maßnahmen sicherstellen, dass nicht mehr als die maximale Anzahl der in der Lizenz vorgesehenen Benutzer gleichzeitig auf die Software zugreifen. Altova gibt über die Performance der Altova-Software in einer Umgebung, in der Terminalserver, Anwendungsvirtualisierungsserver oder virtuelle Maschinen verwendet werden, keinerlei Zusicherungen oder Gewährleistungen ab. Die Anwendung der Altova-Software auf die vorstehend beschriebene Weise ist

ausdrücklich von der eingeschränkten Garantie, die durch Abschnitt 5 dieses Lizenzvertrags eingeräumt wird, ausgeschlossen. Ebenso wird kein technischer Support bei Problemen geleistet, die aufgrund der Anwendung in einer solchen Umgebung auftreten.

(f) Kopien für Sicherungs- und Archivierungszwecke. (iii) Sie dürfen eine (1) Kopie der Software für Sicherungs- und eine Kopie für Archivierungszwecke herstellen, sofern diese nicht auf einem Computer installiert oder verwendet werden. Außerdem müssen bei Kopien zu Sicherungs- und Archivierungszwecken alle im Original enthaltenen urheber- und patentrechtlichen Angaben sowie alle anderen Angaben hinsichtlich geistiger Eigentumsrechte von Ihnen reproduziert werden und in der jeweiligen Kopie der Software unverändert enthalten sein. Die Befugnis zum Anlegen einer Sicherungs- und Archivierungskopie darf nur an Dritte übertragen werden, wenn gleichzeitig eine Übertragung aller Rechte an der Software gemäß Abschnitt 3 erfolgt.

(g) Nutzung auf dem Privat-PC (privater und nicht-kommerzieller Gebrauch). Um sich mit der Software, ihren Features und Funktion besser vertraut zu machen, sind Sie als vorrangiger Benutzer des Computers, auf dem die Software installiert wurde, berechtigt, ein weiteres Exemplar der Software zum persönlichen und nicht kommerziellen Gebrauch („pnk“) auf einem (1) einzigen Ihrer Heimcomputer (z.B. Ihrem Laptop oder PC) zu installieren. Dieses pnk-Exemplar darf zu keinerlei kommerziellen oder umsatzgenerierenden Geschäftsaktivitäten verwendet werden. Dazu gehören insbesondere die Arbeit von zu Hause aus, jedwede Form von Telearbeit oder andere Arbeit, die im Zusammenhang mit der Verwendung der Software steht. Das pnk-Exemplar der Software darf nicht gleichzeitig auf auf einem Privat-PC und dem eigentlich dafür vorgesehenen Primärcomputer verwendet werden.

(h) Freischaltcodes, Upgrades und Updates. Bevor Sie die Software kaufen bzw. im Rahmen Ihrer Registrierung für den Evaluierungszeitraum von dreißig (30) Tagen erhalten Sie einen Evaluierungs-Freischaltcode. Wenn Sie sich in der Folge dazu entscheiden, die Software von Altova GmbH oder einem autorisierten Vertriebshändler zu kaufen, erhalten Sie einen uneingeschränkten Freischaltcode. Mit diesem können Sie die Software nach dem Evaluierungszeitraum aktivieren. Ohne ausdrückliche schriftliche Genehmigung von Altova dürfen Sie an Freischaltcodes keine Lizenzen einräumen und Freischaltcodes nicht reproduzieren oder vertreiben. Wenn die von Ihnen in Lizenz erworbene Software ein Upgrade oder Update ist, ersetzt das Upgrade oder Update das sie zuletzt heruntergeladen oder installiert haben, die zuvor lizenzierte Kopie der Software. Durch das betreffende Upgrade oder Update und den dazugehörigen Freischaltcode wird keine zweite Lizenz für die Software gewährt, und Sie dürfen die Kopie des Upgrades oder Updates nicht zusätzlich zu der Kopie der Software verwenden, die dadurch ersetzt wird und dessen Lizenz beendet wurde.

(i) Eigentumsrecht. Das Eigentumsrecht an der Software wird Ihnen nicht übertragen. Alle Exemplare der Software und alle von Ihnen angefertigte Kopien verbleiben – vorbehaltlich der Ihnen durch diesen Lizenzvertrag eingeräumten Rechte – im Eigentum von Altova. Dokumente, Dateien, Stylesheets, generierter Programmcode (dies beinhaltet auch den unbeschränkten Quellcode) und Schemata, die Sie unter Verwendung der Software anlegen oder erstellen, sind in Übereinstimmung mit der dazugehörigen Dokumentation und den Bestimmungen dieses Lizenzvertrags Ihr Eigentum außer diese wurden mit einer Testsoftware gemäß Abschnitt 4 dieser Vereinbarung erstellt. In diesem Fall wird Ihnen nur eine beschränkte Lizenz zur Nutzung sämtlicher Ergebnisse, welche einen generierten Programmcode (inklusive eines unbeschränkten Quellcodes) enthalten, wie Java, C++, C#, VB.NET oder XSLT und verbundene Projektdateien und Build Scripts sowie generierte XML, XML Schematas, Dokumentationen, UML Diagrammen und Datenbankstrukturen, für die 30 Tage dauernde Testperiode eingeräumt.

(j) Reverse Engineering. Sofern dies nicht ausdrücklich nach geltendem Recht der Europäischen Union gestattet ist, sind Sie nicht berechtigt, an der Software Reverse Engineering durchzuführen, die Software zu dekompileieren, zu disassemblieren, oder auf andere Weise zu versuchen, den Quellcode, zugrunde liegende Konzepte, Techniken für

Benutzerschnittstellen oder Algorithmen der Software direkt oder indirekt zu entschlüsseln. Davon ausgenommen ist lediglich das ausdrücklich nach geltendem Recht der Europäischen Union gestattete Maß an Dekompilierung, das erforderlich ist, um die Funktionsfähigkeit der Software mit anderen Softwareprogrammen zu erzielen. Dies ist Ihnen jedoch erst dann gestattet, wenn Sie zuerst bei Altova die dafür notwendigen Informationen angefordert haben und Altova Ihnen diese nicht zur Verfügung gestellt hat. Altova ist berechtigt, für die Bereitstellung solcher Informationen angemessene Bedingungen aufzustellen und eine angemessene Entschädigung zu verlangen. Die von Altova zur Verfügung gestellten Informationen bzw. alle von Ihnen im Sinne der hier festgelegten Bestimmungen rechtmäßig erworbenen Informationen dürfen nur zu den hierin angeführten Zwecken verwendet und keinesfalls an Dritte weitergegeben oder zur Entwicklung von Software genutzt werden, die der Software, die Gegenstand dieses Lizenzvertrags ist, in wesentlichen Aspekten ähnlich ist. Informationen im Sinne dieses Absatzes können von Benutzern aus der Europäischen Union bei der Kundendienstabteilung von Altova angefordert werden.

(k) Sonstige Beschränkungen. Sie sind nicht berechtigt, die Software (zur Gänze oder teilweise) zu verleihen, zu vermieten, per Leasing zur Verfügung zu stellen, weiterzulizenzieren, weiterzugeben oder auf sonstige Weise Dritten zu überlassen, es sei denn, dies ist durch die Bestimmungen in Abschnitt 3 oder an anderer Stelle in diesem Lizenzvertrag ausdrücklich gestattet. Sofern dies nicht ausdrücklich in diesem Vertrag gestattet ist, dürfen Sie die Software nicht kopieren. Alle Kopien, zu deren Anfertigung Sie berechtigt sind, müssen die im Original enthaltenen urheber- und patentrechtlichen Angaben sowie alle anderen Angaben hinsichtlich geistiger Eigentumsrechte unverändert enthalten. Die Software darf nicht verändert, adaptiert oder übersetzt werden. Sie dürfen weder direkt noch indirekt ein Pfand- oder Sicherungsrecht an der Software bestellen oder die Bestellung eines solchen zulassen. Es ist Ihnen nicht gestattet, wissentlich eine Handlung vorzunehmen, durch die die Software öffentlich zugänglich wird, oder die Software in einer Computerumgebung zu verwenden, die nicht in diesem Lizenzvertrag angegeben ist.

Sie haben die geltenden gesetzlichen Bestimmungen und die Anweisungen von Altova in Bezug auf die Benutzung der Software einzuhalten. Sie sind verpflichtet, Ihre Mitarbeiter und Vertreter, die Zugriff auf die Software haben, von den in diesem Lizenzvertrag enthaltenen Beschränkungen in Kenntnis zu setzen und diese auf sie zu überbinden.

(l) ES WIRD KEINE GARANTIE ODER GEWÄHRLEISTUNG FÜR DIE FEHLERFREIHEIT DER SOFTWARE ABGEGEBEN ODER GEWÄHRT UND VON ALTOVA KEINERLEI HAFTUNG DIESBEZÜGLICH ÜBERNOMMEN. UNABHÄNGIG VON IRGENDWELCHEN SUPPORTLEISTUNGEN NACH IRGEND EINEM TECHNISCHEN STANDARD IST DIE SOFTWARE KEINESFALLS FÜR DIE VERWENDUNG IN ODER IM ZUSAMMENHANG MIT KERNKRAFTANLAGEN, FLUGGERÄTENAVIGATION, KOMMUNIKATIONSSYSTEMEN ODER LUFTVERKEHRSKONTROLLEINRICHTUNGEN, MEDIZINISCHEN GERÄTEN ODER LEBENSERHALTUNGSSYSTEMEN, MEDIZINISCHEN ODER DIE GESUNDHEITSVORSORGE BETREFFENDEN ANWENDUNGEN, ODER SONSTIGEN ANWENDUNGEN KONZIPIERT, BEI DENEN FEHLER DER SOFTWARE ODER FEHLER BEI DER DATENVERARBEITUNG ZU TODESFÄLLEN, PERSONENSCHÄDEN ODER SCHWEREN SACH- ODER UMWELTSCHÄDEN FÜHREN KÖNNTEN. SIE ERKLÄREN SICH DAMIT EINVERSTANDEN, DASS SIE DIE ALLEINIGE VERANTWORTUNG FÜR DIE EIGNUNG UND ZWECKMÄSSIGKEIT DER SOFTWARE UND ALLE MIT DER SOFTWARE ERSTELLTEN ODER VERARBEITETEN DATEN FÜR DEN BEABSICHTIGTEN VERWENDUNGSZWECK TRAGEN UND SIE WERDEN ALTOVA UND DEREN GESCHÄFTSFÜHRER SOWIE ANGESTELLTE HINSICHTLICH ALLER ANSPRÜCHE, FORDERUNGEN UND KLAGEN DRITTER, DIE AUF DIE EIGNUNG UND ZWECKMÄSSIGKEIT DER VON IHNEN VERWENDETEN SOFTWARE ODER DIE VON DER SOFTWARE ERSTELLTEN DATEN GEGRÜNDET SIND, SCHAD- UND KLAGLOS HALTEN.

2. GEISTIGE EIGENTUMSRECHTE

Anerkennung der Rechte von Altova. Die Software und alle Kopien, deren Anfertigung Ihnen von Altova gestattet ist, sind das geistige Eigentum von Altova und dessen Lieferanten und stehen in deren Besitz. Struktur, Organisation und Code der Software stellen wertvolle Betriebsgeheimnisse und vertrauliche Informationen von Altova und dessen Lieferanten dar. Die Software ist durch gesetzliche Bestimmungen urheberrechtlich geschützt. Diese gesetzlichen Bestimmungen beinhalten (ohne darauf beschränkt zu sein) das Urheberrecht der Vereinigten Staaten, internationale Verträge und das in den Ländern, in denen die Software genutzt wird, geltende Recht. Sie anerkennen, dass sich Altova das Eigentum an allen Patenten, Urheberrechten, Branchengeheimnissen, Warenzeichen und sonstigen geistigen Eigentumsrechten, die hinsichtlich der Software bestehen, vorbehält. Das Eigentumsrecht von Altova erstreckt sich auch auf alle Bilder, Fotografien, Animationen, Videos, Audioaufzeichnungen, Musikstücke, Texte und „Applets“, die Teil der Software sind, und alle dazugehörigen Unterlagen in gedruckter Form. Sie dürfen keine Handlungen vornehmen, die sich nachteilig auf die geistigen Eigentumsrechte von Altova an der Software auswirken. Warenzeichen dürfen nur in Übereinstimmung mit den anerkannten Standards für die Verwendung von Warenzeichen (einschließlich der namentlichen Nennung der Warenzeicheninhaber) verwendet werden. Die Verwendung von Warenzeichen ist nur zur Kennzeichnung von Druckmaterialien, die mit der Software hergestellt wurden, gestattet. Es entstehen Ihnen daraus keinerlei Eigentumsrechte an dem betreffenden Warenzeichen. XMLSpy®, Authentic®, StyleVision®, MapForce®, UModel®, DatabaseSpy®, DiffDog®, SchemaAgent®, SemanticWorks®, MissionKit®, Markup Your Mind®, Nanonull™ und Altova® sind (in mehreren Ländern eingetragene) Warenzeichen von Altova GmbH. Unicode und das Logo von Unicode sind Warenzeichen von Unicode, Inc. Windows, Windows XP, Windows Vista und Windows 7 sind Warenzeichen von Microsoft. W3C, CSS, DOM, MathML, RDF, XHTML, XML und XSL sind (in mehreren Ländern eingetragene) Warenzeichen des World Wide Web Consortium (W3C). Marken des W3C sind von den Dachinstitutionen des Konsortiums (MIT, INRIA und Keio) eingetragen und stehen in deren Eigentum. Sofern dies nicht ausdrücklich hierin festgelegt ist, entstehen Ihnen aus diesem Lizenzvertrag keinerlei geistigen Eigentumsrechte an der Software. Mitteilungen über geltend gemachte Verstöße gegen geistige Eigentumsrechte sind an den Copyright Agent von Altova zu richten (nähere Angaben dazu finden Sie auf der Website von Altova).

3. ÜBERTRAGUNGSBESCHRÄNKUNG

Ungeachtet der vorstehenden Bestimmungen dürfen Sie alle Ihre Rechte zur Verwendung der Software an eine andere natürliche oder juristische Person unter der Voraussetzung übertragen, dass: (a) Sie den vorliegenden Lizenzvertrag, die Software und sonstige Soft- oder Hardware, die mit der Software, die Gegenstand dieses Lizenzvertrags ist, geliefert, verpackt oder auf dieser vorinstalliert ist, einschließlich aller Kopien, Updates und früherer Versionen sowie aller Exemplare von Fontsoftware, die in andere Formate konvertiert wurde, an die betreffende Person übertragen; (b) Sie keine Kopien, einschließlich Sicherungskopien und sonstiger, auf einem Computer gespeicherter Kopien, zurückbehalten; (c) der Empfänger von Altova einen persönlichen Freischaltcode von Altova erhält; und (d) der Empfänger die Bestimmungen dieses Lizenzvertrags sowie alle anderen Bestimmungen akzeptiert, zu denen Sie die Softwarelizenz rechtmäßig erworben haben. Ungeachtet der vorstehenden Ausführungen dürfen Sie keine Schulungs-, Vorab- oder Musterkopien der Software an Dritte übertragen.

4. ZUSÄTZLICHE BEDINGUNGEN FÜR VORAB- UND TESTVERSIONEN

Wenn es sich bei dem von Ihnen mit dieser Lizenz erworbenen Produkt um eine unverkäufliche Vorabversion oder um Beta-Software („Pre-Release-Software“) handelt, gelten die in diesem Abschnitt enthaltenen Bedingungen. Diese gelten außerdem für alle Test- und/oder Demoversionen von Altova-Software („Testsoftware“) und bleiben so lange gültig, bis Sie eine Lizenz erwerben. Wenn Bestimmungen dieses Abschnitts im Widerspruch zu anderen, in

diesem Lizenzvertrag enthaltenen Bestimmungen stehen, dann gehen die Bedingungen dieses Abschnitts hinsichtlich Pre-Release-Software und Testsoftware allen anderen Bestimmungen dieses Lizenzvertrags vor, soweit dies für die Auflösung des Widerspruchs erforderlich ist. Sie erkennen an, dass es sich bei der Pre-Release-Software um eine Vorabversion handelt, die nicht das endgültige Produkt von Altova darstellt, und in der Fehler, Funktionsstörungen oder andere Probleme auftreten können, die möglicherweise zu einem System- oder Hardwareabsturz oder -fehler bzw. zu Datenverlust führen. DIE PRE-RELEASE-SOFTWARE UND/ODER TESTSOFTWARE WIRD IHNEN DAHER **OHNE GEWÄHR FÜR DEREN VERWENDUNG ODER LEISTUNGSMERKMALE** ZUR VERFÜGUNG GESTELLT, UND ALTOVA ÜBERNIMMT DIESBEZÜGLICH KEINERLEI AUSDRÜCKLICHEN ODER STILLSCHWEIGENDEN GEWÄHRLEISTUNGS- ODER HAFTUNGSVERPFLICHTUNGEN. SOFERN UND SOWEIT FÜR PRE-RELEASE- UND/ODER TESTSOFTWARE KEIN HAFTUNGSSAUSSCHLUSS, SONDERN NUR EINE HAFTUNGSBESCHRÄNKUNG GESETZLICH ZULÄSSIG IST, BESCHRÄNKT SICH DIE HAFTUNG VON ALTOVA UND DESSEN LIEFERANTEN AUF INSGESAMT FÜNFZIG US-DOLLAR (50 USD). Wenn die Testversion mit einer Time-out-Funktion ausgestattet ist, kann sie nach Ablauf des Evaluierungszeitraums nicht mehr verwendet werden. Mit dem Ablauf dieses Zeitraums endet Ihre Lizenz automatisch, wenn Sie sie nicht verlängern. Ihre Lizenz zur Nutzung sämtlicher mit der Testsoftware erstellten Ergebnisse, welche einen generierten Programmcode (inklusive den unbeschränkten Quellcode) wie Java, C++, C#, VB.NET oder XSLT und verbundene Projektdateien und Build Scripts sowie generierte XML, XML Schematas, Dokumentationen, UML Diagramme und Datenbankstrukturen enthalten, erlischt automatisch nach Ablauf der vorgesehenen Testperiode. Die Lizenz zum Gebrauch dieser Ergebnisse lebt nach dem Kauf einer Lizenz für die Software, die Sie getestet und mit der Sie das Ergebnis erstellt haben, wieder auf. Der Zugriff auf Dateien, die mit der Testsoftware erstellt wurden, erfolgt auf Ihr alleiniges Risiko. Sie bestätigen, dass Altova Ihnen weder zugesichert noch garantiert hat, die Pre-Release-Software zu einem späteren Zeitpunkt anderen Personen anzukündigen oder zur Verfügung zu stellen. Sie bestätigen ferner, dass Altova sich Ihnen gegenüber weder ausdrücklich noch stillschweigend verpflichtet hat, die Pre-Release-Software öffentlich anzukündigen oder einzuführen, und dass Altova kein Produkt anbieten muss, das der Pre-Release-Software ähnlich oder mit ihr kompatibel ist. Sie bestätigen folglich, dass jegliche Forschungs- oder Entwicklungsarbeit, die Sie in Bezug auf die Pre-Release-Software oder damit verbundene Produkte ausführen, in Ihr alleiniges Risiko fällt. Wenn Altova Sie dazu auffordert, werden Sie für die Dauer dieses Lizenzvertrags Altova Rückmeldung bezüglich Ihrer Tests und der Pre-Release-Software geben, einschließlich entsprechender Berichte über Fehler und Funktionsstörungen. Wenn Sie die Pre-Release-Software im Rahmen einer gesonderten schriftlichen Vereinbarung erhalten haben, unterliegt Ihre Verwendung der Software den darin enthaltenen Bedingungen. Es ist Ihnen nicht gestattet, die Pre-Release-Software unterzulizenzieren, per Leasing zur Verfügung zu stellen, zu verleihen, zu vermieten, weiterzugeben oder auf sonstige Weise Dritten zu überlassen. Sie verpflichten sich, bei Erhalt einer späteren unveröffentlichten Version der Pre-Release-Software bzw. bei Erhalt einer kommerziellen Vollversion von Altova (entweder als Einzelprodukt oder als Teil eines Gesamtprodukts) die zuvor von Altova erhaltene Pre-Release-Software zurückzugeben oder zu vernichten und den Bestimmungen des Lizenzvertrags für die betreffende spätere Version der Pre-Release-Software Folge zu leisten.

5. EINGESCHRÄNKTE GARANTIE UND HAFTUNGSBESCHRÄNKUNG

(a) Eingeschränkte Garantie und Garantieansprüche der Kunden. Altova garantiert der Person/Rechtspersönlichkeit, die ursprünglich die Lizenz für die Verwendung der Software gemäß den Bestimmungen dieses Lizenzvertrags erworben hat, dass (i) die Software über einen Zeitraum von neunzig (90) Tagen nach Erhalt im Wesentlichen in Übereinstimmung mit der dazugehörigen Dokumentation funktioniert, und (ii) die von Altova zur Verfügung gestellten Supportleistungen im Wesentlichen auf die in Abschnitt 6 dieses Vertrags niedergelegte Weise erfolgen. In einigen Ländern sind Beschränkungen über die Dauer einer stillschweigenden Garantie nicht zulässig, so dass die obigen Beschränkungen und Ausschlüsse eventuell für Sie nicht zutreffen. Im größtmöglichen rechtlich zulässigen Maß sind stillschweigende Garantien in

Bezug auf die Software (sofern solche existieren) auf neunzig (90) Tage beschränkt. Die gesamte Haftung von Altova und dessen Lieferanten, sowie Ihre einzigen Garantieansprüche, sind nach dem Ermessen von Altova auf eine der beiden folgenden Optionen beschränkt: (i) Erstattung des Kaufpreises (wenn zutreffend), oder (ii) Reparatur oder Austausch der Software, die unter die eingeschränkte Garantie von Altova fällt und die unter Vorlage einer Kopie des Kaufbelegs bei Altova reklamiert wird. Diese eingeschränkte Garantie gilt nicht, wenn die Funktionalität der Software durch ein Versehen, durch Missbrauch, falsche Anwendung, Trojaner, Viren oder einen sonstigen schädlichen externen Code beeinträchtigt wurde. Die Garantie für jede Ersatzsoftware erstreckt sich auf die Restdauer des ursprünglichen Garantiezeitraums oder auf dreißig (30) Tage, je nachdem, welcher Zeitraum länger ist. Die eingeschränkte Garantie gilt nicht für Test- und/oder Pre-Release-Software.

(b) Keine weiteren Garantien, Haftungsausschluss. MIT AUSNAHME DER VORSTEHEND ANGEFÜHRTEN EINGESCHRÄNKTEN GARANTIE UND DEN DIESBEZÜGLICHEN GARANTIEANSPRÜCHEN BESTEHEN SEITENS ALTOVA ODER DESSEN LIEFERANTEN KEINERLEI WEITEREN GARANTIEVERPFLICHTUNGEN. ALTOVA UND DESSEN LIEFERANTEN ÜBERNEHMEN KEINE GEWÄHR FÜR DIE ANWENDUNG ODER DIE ERGEBNISSE AUS DER NUTZUNG DER SOFTWARE. MIT AUSNAHME DER VORSTEHEND ANGEFÜHRTEN EINGESCHRÄNKTEN GARANTIE SOWIE IM HINBLICK AUF ALLE ANDEREN GEWÄHRLEISTUNGEN, BEDINGUNGEN, ZUSICHERUNGEN ODER ANSPRÜCHE, DIE NACH DER FÜR IHR LAND GELTENDEN RECHTSORDNUNG NICHT AUSGESCHLOSSEN ODER EINGESCHRÄNKT WERDEN KÖNNEN, SCHLIESSEN ALTOVA UND DESSEN LIEFERANTEN ALLE ANDEREN AUSDRÜCKLICHEN ODER STILLSCHWEIGENDEN GEWÄHRLEISTUNGEN, BEDINGUNGEN, ZUSICHERUNGEN UND ANSPRÜCHE AUS, DIE SICH AUS DEM GESETZ, DER RECHTSPRAXIS, EINEM GEWOHNHEITSRECHT, EINEM HANDELSBRAUCH ODER AUS SONSTIGEN GRÜNDEN ERGEBEN. ALTOVA UND DESSEN LIEFERANTEN SCHLIESSEN IM GRÖSSTMÖGLICHEN RECHTLICH ZULÄSSIGEN UMFANG ALLE ANDEREN AUSDRÜCKLICHEN UND STILLSCHWEIGENDEN GARANTIEEN UND GEWÄHRLEISTUNGEN AUS. DIES BEINHÄLTET UNTER ANDEREM GARANTIEEN IM HINBLICK AUF MARKTGÄNGIGKEIT, EIGNUNG FÜR EINEN BESTIMMTEN ZWECK, ZUFRIEDENSTELLENDEN QUALITÄT, INFORMATIONSGEHALT ODER -RICHTIGKEIT, UNGESTÖRTE NUTZUNG, EIGENTUMSRECHT UND NICHTVERLETZUNG VON RECHTEN DRITTER UND DIE ERBRINGUNG ODER NICHTERBRINGUNG VON SUPPORTLEISTUNGEN IN BEZUG AUF DIE SOFTWARE. MIT DIESER EINGESCHRÄNKTEN GARANTIE WERDEN IHNEN BESTIMMTE RECHTE EINGERÄUMT. UNTER UMSTÄNDEN BESITZEN SIE NOCH ANDERE RECHTE; DIE JA NACH LAND/RECHTSORDNUNG UNTERSCHIEDLICH SEIN KÖNNEN.

(c) Haftungseinschränkung. SOWEIT DIE ANWENDBAREN GESETZE DIES ZULASSEN, HAFTEN ALTOVA ODER DESSEN LIEFERANTEN, AUCH WENN EINE IM RAHMEN DER GARANTIE DURCHFÜHRTE ABHILFEMASSNAHME IHREN WESENTLICHEN ZWECK NICHT ERFÜLLT, IN KEINEM FALL FÜR KONKRETE, ZUFÄLLIG ENTSTANDENE, MITTELBARE, UNMITTELBARE ODER FOLGESCHÄDEN JEDLICHER ART (INSBESONDERE SCHÄDEN AUS ENTGANGENEM GEWINN, GESCHÄFTSUNTERBRECHUNGEN, VERLUST VON GESCHÄFTSINFORMATIONEN ODER ANDEREN FINANZIELLEN VERLUSTEN), DIE DURCH DIE NUTZUNG ODER DIE UNMÖGLICHKEIT DER NUTZUNG DER SOFTWARE ODER DIE NICHTERBRINGUNG VON SUPPORTLEISTUNGEN ENTSTANDEN SIND, SELBST WENN ALTOVA AUF DIE MÖGLICHKEIT EINES SOLCHEN SCHADENS HINGEWIESEN WURDE. IN JEDEM FALL IST DIE GESAMTHAFTUNG VON ALTOVA GEMÄSS DIESEM LIZENZVERTRAG AUF DIE HÖHE DES FÜR DAS SOFTWAREPRODUKT ENTRICHTETEN BETRAGES BESCHRÄNKT. Da es in einigen Ländern und Rechtsordnungen nicht zulässig ist, die Haftung für Schäden auszuschließen oder zu beschränken, gilt die obige Einschränkung unter Umständen nicht für Sie. In solchen Ländern und Rechtsordnungen gilt die Beschränkung der Haftung durch Altova im größtmöglichen rechtlich zulässigen Umfang, und der Ausschluss bzw. die Beschränkung der in diesem Lizenzvertrag beinhalteten Garantien und Haftungen hat keine Auswirkung auf

die Konsumentenschutzrechte von Personen, die Waren auf andere Weise als im Zuge ihrer geschäftlichen Tätigkeit erwerben. Der vorstehende Haftungsschluss/die vorstehende Haftungsbeschränkung sind wesentliche Bestandteile des zwischen Ihnen und Altova abgeschlossenen Lizenzvertrags.

(d) Ansprüche in Zusammenhang mit Urheberrechtsverletzungen. Altova wird Sie gegenüber allen Forderungen, Prozessen oder Verfahren schad- und klaglos halten bzw. alle Ansprüche, Prozesse oder Verfahren beilegen, die Dritte mit dem Argument gegen Sie erheben, dass der Inhalt der Software gegen ein Urheberrecht verstößt oder ein geistiges oder sonstiges Eigentumsrecht verletzt, das durch das Recht der Vereinigten Staaten oder der Europäischen Union geschützt ist (insgesamt als „Ansprüche“ bezeichnet). Dies erfolgt – soweit nicht ausdrücklich etwas Anderes festgelegt ist – jedoch nur insoweit, als der betreffende Anspruch sich direkt aus der Verwendung der Software ergibt und nach Maßgabe der in Abschnitt 5 dieses Vertrags festgelegten Beschränkungen. Altova ist von jedem Anspruch innerhalb von zehn (10) Geschäftstagen, nachdem Sie erstmals davon benachrichtigt wurden, in Kenntnis zu setzen. Außerdem haben Sie mit Altova in angemessenem Umfang bei der Abwehr solcher Ansprüche zu kooperieren und Altova dabei zu unterstützen, ohne dass Sie dafür Kosten geltend machen können. Das Recht auf Entscheidungen in Bezug auf solche Ansprüche liegt allein bei Altova (dies beinhaltet auch, ohne darauf beschränkt zu sein, die Auswahl der Rechtsberater und das Recht, für Sie einen Vergleich zu den von Altova für zweckmäßig erachteten Bedingungen einzugehen). Sie können auf eigene Kosten einen Rechtsberater hinzuziehen und an den Verfahrens- oder Vergleichsverhandlungen teilnehmen. Altova kommt bis zu einer Höhe von insgesamt maximal dem Kaufpreis der Software für die Schäden, Kosten und Anwaltsgebühren auf, zu deren Bezahlung Sie in Zusammenhang mit solchen Ansprüchen verpflichtet werden (oder die Sie aufgrund eines Vergleichs zu entrichten haben), soweit diese nicht von einer Versicherung oder dritten Partei übernommen werden. Ist oder wird die Software Gegenstand von aufgrund einer Urheberrechtsverletzung vorgebrachten Ansprüchen, oder wird ihre Verwendung untersagt, oder ist es nach Ansicht des Rechtsberaters von Altova wahrscheinlich, dass ein solcher Umstand eintritt, so wird Altova versuchen, eine Beilegung herbeizuführen, indem alle wirtschaftlich vertretbaren Anstrengungen unternommen werden, um die Software zu modifizieren oder eine Lizenz für die weitere Verwendung der Software zu erwerben. Wenn es nach Ansicht des Rechtsberaters von Altova nicht möglich ist, den bevorstehenden oder bereits vorgebrachten Anspruch bzw. die Verfügung, mit der die Verwendung der Software untersagt wurde, durch angemessene Abänderung oder den Erwerb einer Lizenz beizulegen, so kann Altova diesen Lizenzvertrag ohne negative Konsequenzen für Altova beenden und Ihnen anteilig alle bereits an Altova entrichteten Gebühren zurückerstatten.

MIT AUSNAHME DER VORSTEHEND ANGEFÜHRTEN BESTIMMUNGEN OBLIEGEN ALTOVA KEINERLEI HAFTUNGSVERPFLICHTUNGEN FÜR ANSPRÜCHE IN ZUSAMMENHANG MIT URHEBERRECHTSVERLETZUNGEN. Diese Haftungsverpflichtung gilt nicht für Urheberrechtsverletzungen, die ausschließlich auf vom Kunden eingebrachte Elemente zurückzuführen sind.

6. SUPPORT UND WARTUNG

Zusätzlich zu der Version des Softwareprodukts, für das Sie eine Lizenz erhalten haben, bietet Altova eine Reihe von optionalen „Support- & Wartungspaketen“ („*Software & Maintenance Package*“, „SMP“) an, die Sie zusätzlich zu Ihrer Lizenz für die Software erwerben können. Der für das jeweilige SMP geltende Supportzeitraum (wie nachfolgend definiert) wird beim Kauf des SMP festgelegt. Die Support- und Wartungsservices und die Upgrades, die Ihnen zur Verfügung stehen, hängen davon ab, ob Sie ein SMP erwerben bzw. für welche Version des SMP Sie sich entscheiden.

(a) Wenn Sie kein SMP erwerben, erhalten Sie beim Kauf lediglich die Software, danach jedoch keinerlei Wartungsreleases oder Updates. Obwohl die Möglichkeit besteht, dass Altova Ihnen in Einzelfällen Wartungsreleases als kostenlose Zusatzleistung zur Verfügung stellt, sind in diesen Releases keine neuen Produktfeatures, die über das beim Kauf der Software bestehende Maß hinausgehen, beinhaltet. In jedem Fall erhalten Sie dreißig (30) Tage lang ab

Kauf der Software (der „Supportzeitraum“ im Sinne dieses Absatzes 6(a)) kostenlosen technischen Support von Altova. Außerdem kann Altova in Einzelfällen auch während des dreißig (30) Tage dauernden Evaluierungszeitraums technischen Support als kostenlose Zusatzleistung zur Verfügung stellen. Technischen Support erhalten Sie ausschließlich über ein webbasiertes Supportformular, wobei es keine garantierte Reaktionszeit gibt.

(b) Mit dem Erwerb eines SMP haben Sie während des dafür geltenden Supportzeitraums **Anspruch auf die von Ihnen erworbene Produktversion** sowie auf alle Wartungsreleases und Updates für diese Produktversion, die während des für Sie geltenden Supportzeitraums freigegeben werden. Während des Supportzeitraums Ihres SMP erhalten Sie auch Upgrades auf die entsprechende Produktversion der Software, mit der auf eine höhere Version der Software, für die Sie die Lizenz erworben haben, gewechselt wird und die während Ihres Supportzeitraums freigegeben werden. Die einzelnen Upgradeeditionen, auf die Sie innerhalb Ihres Supportzeitraums Anspruch haben, sind in dem von Ihnen erworbenen SMP im Detail angeführt. Software, die als gesondertes Produkt eingeführt wird, ist nicht im SMP enthalten. Wartungsreleases, Updates und Upgrades können neue Features enthalten, dies muss aber nicht der Fall sein. Darüber hinaus erhalten Sie während des Supportzeitraums bevorzugten technischen Support von Altova, und zwar ausschließlich über ein webbasiertes Supportformular. Altova wird alle wirtschaftlich vertretbaren Anstrengungen unternehmen, um alle Anfragen per E-Mail innerhalb von achtundvierzig (48) Stunden während der Geschäftszeiten (Montag bis Freitag, 8.00 bis 22.00 Uhr UTC, ausgenommen Feiertage in Österreich und den USA) zu beantworten. Außerdem wird Altova sich in angemessenem Umfang darum bemühen, Workarounds für Fehler, die bei der Software aufgetreten sind, zur Verfügung zu stellen.

Während des Supportzeitraums können Sie Altova Fehler oder Defekte in der Software melden. Wenn nach Altovas Einschätzung ein reproduzierbarer schwerwiegender Fehler vorliegt, der die Verwendung und Funktionalität der Software erheblich beeinträchtigt, wird Altova wirtschaftlich vertretbaren Anstrengungen unternehmen um Korrekturen oder provisorische Lösungen anzubieten, die in zukünftigen Updates oder Wartungsreleases enthalten sind. Diese Updates oder Wartungsreleases werden von Altova von Zeit zu Zeit zur Verfügung gestellt.

Es steht im Ermessen von Altova, einen schriftlichen Nachweis über von Ihnen festgestellte Fehler oder Funktionsstörungen zu verlangen oder Beispieldateien anzufordern, aus denen das aufgetretene Softwareproblem hervorgeht. In einem solchen Fall müssen Sie die angeforderten Nachweise oder Dateien, aus denen ausreichend detailliert hervorgeht, in welchen Aspekten Fehler bei der Software auftreten, so schnell wie möglich per E-Mail, Fax oder Expresspost mit Zustellung am nächsten Tag an Altova übermitteln. Bei der Diagnose oder Analyse von Fehlern haben Sie in zumutbarem Rahmen mit Altova zu kooperieren. Fehlerbehebungen können in Wartungsreleases, Updates oder neuen Hauptversionen der Software enthalten sein. Altova ist nicht verpflichtet, unwesentliche Fehler, d.h. Fehler, die die Benutzung der Software nach dem Ermessen von Altova nicht wesentlich beeinträchtigen, zu beheben. Sie erhalten von Altova unter keinen Umständen Beratung, Unterstützung oder Hilfestellung, die allgemeiner Natur ist und nicht in unmittelbarem Zusammenhang mit der Software steht.

Bei der Aktualisierung der Software kann es unter Umständen vorkommen, dass vor der Installation auch Software aktualisiert werden muss, die nicht unter diesen Lizenzvertrag fällt. Für Updates von Betriebssystem- und Anwendungssoftware, die nicht ausdrücklich Gegenstand dieses Lizenzvertrags ist, sind ausschließlich Sie verantwortlich. Solche Updates sind nicht im Umfang dieser Lizenz enthalten und werden nicht von Altova zur Verfügung gestellt. Die Erfüllung der in diesem Abschnitt 6 festgelegten Verpflichtungen durch Altova versteht sich unter der Bedingung, dass Sie die Software ordnungsgemäß verwenden und diesen Lizenzvertrag ausnahmslos jederzeit einhalten. Altova ist nicht verpflichtet, technischen Support zu leisten, wenn nach Ansicht von Altova die Fehlfunktion der Software auf einen der folgenden Gründe zurückzuführen ist: (i) Fehler, die durch die Verlegung der Software auf einen anderen Rechner oder Speicherort hervorgerufen werden; (ii) Änderungen, Modifikationen oder Versuche, die Software abzuwandeln, die von Altova nicht schriftlich

genehmigt wurden; (iii) äußere Einflüsse auf die Software, wie z.B. Naturkatastrophen, Stromausfälle, Stromschwankungen oder Computerausfälle; (iv) Ihr Versäumnis die Software auf dem von Altova festgelegten Release Level zu halten; oder (v) Nutzung der Software ohne vorherige Genehmigung durch Altova zusammen mit einer anderen Software. Sie alleine sind dafür verantwortlich: (i) allen Betriebs- und Fehlerbehebungsanleitungen von Altova Folge zu leisten, Altova unverzüglich von Fehlern oder Defekten an der Software zu informieren und Altova eine genaue Beschreibung dieser Fehler und/oder Defekte zu liefern; (ii) für den Schutz Ihrer vertraulichen Informationen zu sorgen; (iii) Datensicherungssysteme und -abläufe für die Wiederherstellung verlorener oder geänderter Dateien, Daten oder Programme einzurichten und anzuwenden.

7. AKTIVIERUNG DER SOFTWARE, UPDATES UND LIZENZÜBERWACHUNG

(a) Lizenzüberwachung. Altova verfügt über ein integriertes Lizenzüberwachungsmodul, das Ihnen hilft, unbeabsichtigte Verletzungen dieses Lizenzvertrags zu vermeiden. Dabei kann Altova zur Lizenzüberwachung von verschiedenen installierten Versionen der Software Ihr internes Netzwerk verwenden.

(b) Aktivierung der Software. Die Software von Altova kann Ihr internes Netzwerk und Ihre Internetverbindung verwenden, um Angaben über Ihre Lizenz im Zuge der Installation, Registrierung, Benutzung oder Aktualisierung an einen Altova-Lizenzserver zu übertragen und diese zu verifizieren, um auf diese Weise eine nicht lizenzierte oder illegale Verwendung der Software zu verhindern und den Kundenservice von Altova weiter zu verbessern. Die Aktivierung erfolgt über einen Austausch von lizenzbezogenen Daten zwischen Ihrem Computer und dem Altova-Lizenzserver. Sie stimmen dieser Vorgangsweise von Altova zu und verpflichten sich, allen diesbezüglichen Vorgaben Folge zu leisten. Sie erklären sich damit einverstanden, dass die Verwendung von Freischaltcodes, welche nicht von Altova erstellt werden oder wurden und nicht rechtmäßig von Altova oder einem dazu berechtigten Wiederverkäufer erworben wurden, zum Zweck der Aktivierung oder Nutzung der Software die gewerblichen Schutzrechte von Altova sowie die Bestimmungen dieses Lizenzvertrages verletzt. Sie erklären sich weiters damit einverstanden, dass Versuche mit dem Zweck der Umgehung oder Deaktivierung der urheberrechtlichen Schutzmaßnahmen oder Lizenzmanagementsystemen von Altova die gewerblichen Schutzrechte von Altova sowie die Bestimmungen dieses Lizenzvertrages verletzen. Altova behält sich ausdrücklich das Recht vor, alle rechtlich verfügbaren sowie angemessenen Mittel zu ergreifen, um derartige Praktiken zu verhindern und entgangenen Gewinn, Schäden und Kosten zurückerstattet zu bekommen.

(c) LiveUpdate. Altova stellt Ihnen ein neues LiveUpdate Benachrichtigungsservice zur Verfügung, welches kostenlos ist. Altova kann Ihr internes Netzwerk und Ihre Internetverbindung verwenden, um Angaben über Ihre Lizenz an einen LiveUpdate-Server von Altova zu übertragen, um Ihre Lizenz in gewissen Zeitabständen zu verifizieren und festzustellen, ob ein Update für Sie verfügbar ist.

(d) Verwendung von Daten. Der gesamte Wortlaut der Datenschutzbestimmungen von Altova kann unter <http://www.altova.com/de/privacy> eingesehen werden. Die Datenschutzbestimmungen sind durch Bezugnahme Teil dieses Lizenzvertrags. Durch Ihre Zustimmung zu den Bestimmungen dieses Lizenzvertrags bzw. durch Benutzung der Software erklären Sie sich damit einverstanden, dass Altova für die in diesem Lizenzvertrag und/oder in den Datenschutzbestimmungen (in der jeweils geltenden Fassung) genannten Zwecke Daten erhebt, verarbeitet und weitergibt. Altova behält sich das Recht vor, diese Bestimmung des Lizenzvertrags und/oder der Datenschutzbestimmungen jederzeit zu ändern. Wir legen Ihnen nahe, die auf der Website von Altova veröffentlichten Datenschutzbestimmungen von Zeit zu Zeit erneut durchzulesen.

(e) Hinweis für Benutzer in Europa. Bitte beachten Sie, dass die in Absatz 7(d) beschriebenen Informationen von Altova, Inc., einem Unternehmen mit Sitz in Beverly, Massachusetts, USA, oder seinen Tochterunternehmen, Zweigniederlassungen oder weltweit ansässigen autorisierten Partnern zum Zweck der Datenverarbeitung, Analyse und Überprüfung nach außerhalb der EU transferiert werden können. Sie werden darauf hingewiesen, dass in den USA ein Datenschutzmodell zur Anwendung kommt, das teils auf Gesetzen, teils auf Regierungsverordnungen und zum Teil auf Selbstregulierung beruht. Des Weiteren werden Sie darauf hingewiesen, dass der Rat der Europäischen Union befunden hat, dass dieses amerikanische Modell den in Artikel 25 der Datenrichtlinie der Europäischen Union (Richtlinie 95/46/EC, 1995 O.J. (L 281) 31) festgelegten Datenschutzbestimmungen nicht "in ausreichendem Ausmaß" Rechnung trägt. Laut Artikel 26 der Datenrichtlinie der Europäischen Union, dürfen persönliche Daten dann von der Europäischen Union in ein Drittland übertragen werden, wenn die jeweilige Person ihre Zustimmung zur Übertragung derartiger Informationen eindeutig gegeben hat, unabhängig davon, in welchem Ausmaß diese Daten in anderen Ländern geschützt sind. Durch Ihre Zustimmung zu dieser Software-Lizenzvereinbarung gestatten Sie die Übertragung aller derartiger Informationen an die USA sowie deren Verarbeitung gemäß dieser Software-Lizenzvereinbarung und Altovas Datenschutzbestimmungen.

8. DAUER UND BEENDIGUNG

Dieser Lizenzvertrag wird folgendermaßen beendet: (a) indem Sie Altova eine schriftliche Benachrichtigung von der Vertragsbeendigung übermitteln; (b) durch Altova in Form einer diesbezüglichen schriftlichen Benachrichtigung, wenn Sie gegen diesen Lizenzvertrag verstoßen und diesen Verstoß nicht innerhalb von 10 (zehn) Tagen, nachdem Altova Ihnen dies mitgeteilt hat, berichtigen; oder (c) auf Verlangen eines autorisierten Vertriebshändlers von Altova, wenn Sie eine Lizenzzahlung nicht entrichten oder sonstige Zahlungsverpflichtungen nicht einhalten. Der Lizenzvertrag einer früheren Version der Software, für die Sie ein Upgrade oder ein Update durchgeführt haben, wird mit Ihrer Zustimmung zu dem Lizenzvertrag, der für das Upgrade oder Update gilt, automatisch beendet. Bei Beendigung des Lizenzvertrags dürfen Sie die gesamte Software, die Gegenstand dieser Lizenz ist, nicht mehr verwenden, müssen alle Kopien, die in Ihrem Besitz oder Einflussbereich stehen, vernichten, und müssen in zumutbarem Rahmen alle von Altova geforderten Maßnahmen ergreifen, um sicherzustellen, dass keine Kopien der Software in Ihrem Besitz oder Einflussbereich verbleiben. Die in den Abschnitten 1(h), 1(i), 1(j), 1(k), 2, 5(b), 5(c), 5(d), 7(d), 7(e), 9, 10 und 11 niedergelegten Bestimmungen bleiben auch nach Beendigung dieses Lizenzvertrags weiterhin aufrecht.

9. EINGESCHRÄNKTE RECHTE UND EXPORTBESCHRÄNKUNGEN

Die Entwicklung der Software wurde ausschließlich privat finanziert. Bei der Software handelt es sich um kommerzielle Computersoftware, die mit **eingeschränkten Rechten** ausgestattet ist. Die Verwendung, Vervielfältigung oder Weitergabe der Software durch die US-Regierung, eine Behörde oder einen Kooperationspartner der US-Regierung unterliegt den Beschränkungen im Rahmen dieses Vertrags sowie den Beschränkungen nach FAR 12.211 und 12.212 (48 C.F.R. §12.211 und 12.212) bzw. DFARS 227.7202 (48 C.F.R. §227-7202). Dabei wird eine Lizenz für kommerzielle Computersoftware und kommerzielle Computerdokumentation an die US-Regierung als Endnutzer ausschließlich in Form einer Lizenz für kommerziell genutzte Güter erteilt, weshalb es sich bei den damit verbundenen Rechten um dieselben Rechte handelt, die allen anderen Endnutzern im Rahmen dieses Lizenzvertrags gewährt werden. Beim Hersteller handelt es sich um Altova GmbH, Rudolfsplatz, 13a/9, A-1010 Wien, Österreich/EU. Die Software und/oder Dokumentation darf ausschließlich unter Einhaltung aller anwendbaren Exportbestimmungen der Vereinigten Staaten von Amerika sowie des Landes, in dem Sie die Software erhalten haben, verwendet, exportiert oder wiedereingeführt werden. Insbesondere darf die Software und/oder Dokumentation nicht (i) in ein Land exportiert oder wiedereingeführt werden, über das die Vereinigten Staaten ein Embargo verhängt haben, oder einem Staatsangehörigen oder Bewohner eines solchen Landes überlassen werden; oder (ii) einer Person überlassen werden, die auf der Liste der Specially Designated Nationals des U.S.

Treasury Department oder dem Table of Denial Orders des U.S. Department of Commerce verzeichnet sind. Indem Sie die Software benutzen, erklären Sie, dass Sie weder in einem dieser Länder ansässig sind noch seiner Kontrolle unterliegen, noch ein Staatsangehöriger oder Bewohner eines dieser Länder sind, noch auf einer der vorstehend erwähnten Listen genannt werden.

10. SOFTWARE DRITTER

Die dieser Lizenz unterliegende Software kann Software Dritter enthalten, für die ergänzende Vermerke und/oder Nutzungsbedingungen gelten. Diese Vermerke und ergänzenden Nutzungsbedingungen für die Software Dritter können über unsere Website unter http://www.altova.com/legal_3rdparty.html eingesehen werden und sind durch Bezugnahme Teil dieses Lizenzvertrags. Indem Sie Ihre Zustimmung zu den Bedingungen dieses Vertrags erteilen, stimmen Sie auch automatisch allen ergänzenden Bestimmungen, die möglicherweise darin enthalten sind, zu.

11. ALLGEMEINE BESTIMMUNGEN

Wenn Sie sich in der Europäischen Union befinden und die Software in der Europäischen Union und nicht in den Vereinigten Staaten verwenden, unterliegt dieser Lizenzvertrag dem Recht der Republik Österreich (unter Ausschluss von dessen Verweisungsnormen und der UN-Kaufrechtskonvention). Sie erklären sich ausdrücklich damit einverstanden, dass alle Streitigkeiten oder Konflikte mit Altova, die in Zusammenhang mit Ihrer Verwendung der Software stehen, in die alleinige Zuständigkeit des Handelsgerichts Wien fallen. Sie erklären sich weiters ausdrücklich damit einverstanden, dass Sie bezüglich solcher Streitigkeiten oder Konflikte der persönlichen Zuständigkeit des Handelsgerichts Wien unterstellt sind.

Wenn Sie sich in den Vereinigten Staaten befinden und die Software in den Vereinigten Staaten verwenden, unterliegt dieser Lizenzvertrag dem Recht des Commonwealth of Massachusetts, USA (unter Ausschluss von dessen Verweisungsnormen und der UN-Kaufrechtskonvention). Sie erklären sich ausdrücklich damit einverstanden, dass alle Streitigkeiten oder Konflikte mit Altova, die in Zusammenhang mit Ihrer Verwendung der Software stehen, in die alleinige Zuständigkeit der einzel- und bundesstaatlichen Gerichte im Bundesstaat Massachusetts fallen. Sie erklären sich weiters ausdrücklich damit einverstanden, dass Sie bezüglich solcher Streitigkeiten oder Konflikte der persönlichen Zuständigkeit der einzel- und bundesstaatlichen Gerichte im Bundesstaat Massachusetts unterstellt sind.

Wenn Sie sich nicht in der Europäischen Union oder den Vereinigten Staaten befinden und die Software nicht in den Vereinigten Staaten verwenden, unterliegt dieser Lizenzvertrag dem Recht der Republik Österreich (unter Ausschluss von dessen Verweisungsnormen und der UN-Kaufrechtskonvention). Sie erklären sich ausdrücklich damit einverstanden, dass alle Streitigkeiten oder Konflikte mit Altova, die in Zusammenhang mit Ihrer Verwendung der Software stehen, in die alleinige Zuständigkeit des Handelsgerichts Wien fallen. Sie erklären sich weiters ausdrücklich damit einverstanden, dass Sie bezüglich solcher Streitigkeiten oder Konflikte der persönlichen Zuständigkeit des Handelsgerichts Wien unterstellt sind. Die Anwendung von Verweisungsnormen einer Rechtsordnung sowie des UN-Abkommens zum internationalen Warenkauf (CISG) auf diesen Lizenzvertrag wird ausdrücklich ausgeschlossen.

Dieser Lizenzvertrag enthält die gesamte Vereinbarung zwischen den Parteien in Bezug auf den Vertragsgegenstand und tritt an die Stelle aller diesbezüglichen früheren mündlichen oder schriftlichen Vereinbarungen zwischen den Parteien. Benachrichtigungen oder sonstige Mitteilungen im Rahmen dieses Lizenzvertrags müssen schriftlich erfolgen und gelten als ordnungsgemäß übermittelt, wenn sie per Einschreiben mit Rückschein oder per Kurierdienst mit Zustellung am nächsten Tag an die auf der Website von Altova angegebene Adresse (wenn Altova der Empfänger ist) bzw. an die in den Aufzeichnungen von Altova für Sie eingetragene Adresse (wenn Sie der Empfänger sind) oder aber an eine andere, von den Vertragspartnern festgelegte und auf die vorstehend beschriebene Weise bekannt gegebene Adresse geschickt

werden. Dieser Lizenzvertrag ist für die Vertragspartner verbindlich und geht auf ihre Erben, persönlichen und rechtlichen Vertreter, verbundenen Unternehmen, Rechtsnachfolger und zulässigen Abtretungsempfänger über. Die Nichtdurchsetzung oder Nichtausübung von Rechten oder Rechtsmitteln unter diesem Lizenzvertrag durch die Vertragspartner stellt keinen Verzicht auf ein solches Recht oder Rechtsmittel dar und beeinträchtigt in keiner Weise das Recht des betreffenden Vertragspartners, ein solches Recht oder Rechtsmittel sowie alle anderen Rechte oder Rechtsmittel aus diesem Lizenzvertrag später durchzusetzen bzw. auszuüben. Eine Änderung dieses Lizenzvertrags ist nur Form einer schriftlich niedergelegten Vereinbarung möglich, die von beiden Vertragspartnern unterzeichnet wird. Wenn eine Zuwiderhandlung gegen die Bestimmungen dieses Lizenzvertrags durch einen der Vertragspartner erfolgt ist oder droht, so kann der jeweils andere Vertragspartner alle ihm zustehenden Rechte und Rechtsmittel geltend machen. Jeder Vertragspartner ist ordnungsgemäß befugt und ermächtigt, in diesen Lizenzvertrag einzutreten und die daraus erwachsenden Verpflichtungen zu erfüllen. Sollte eine Bestimmung dieses Lizenzvertrags für unwirksam, rechtswidrig oder undurchführbar erklärt werden, so wird dadurch die Wirksamkeit der übrigen Bestimmungen nicht berührt, und dieser Lizenzvertrag bleibt im größtmöglichen rechtlich zulässigen Umfang wirksam und gültig. Die Vertragspartner haben die vorstehenden Vertragsbedingungen zur Kenntnis genommen und erklären sich ausdrücklich damit einverstanden.

Letzte Aktualisierung: 2011-10-01

Index

■

.docx, 327**.NET,**

Integration von XMLSPY mit, 395

und XSLT Debugger, 400

Unterschiede zur XMLSPY Standalone Version, 398

.NET Erweiterungsfunktionen,

Datentypkonvertierungen, .NET in XPath/XQuery, 1157

Datentypkonvertierungen, XPath/XQuery in .NET, 1156

für XSLT und XQuery, 1151

Instanzmethoden, Instanzfelder, 1155

Konstruktoren, 1154

statische Methoden, statische Felder, 1154

Übersicht, 1151

.pptx, 327**.xlsx, 327**

A

Abfrage,

siehe Datenbank abfragen, 580

siehe Datenbankabfrage (Fenster), 580

siehe XQuery, 580

Abfragefenster,

im Fenster Datenbankabfrage, 587

Abfragen,

für DB-Anzeige in der Authentic-Ansicht, 301

Abgeleitete Types,

Base Type modifizieren, 133

Ableiten eines Schematyps, 143**ActiveX Controls,**

Unterstützung, 737

ADO,

Konvertierung von Datentypen in XML-Schema

Generierung von DB, 1170

ADO-Verbindungen, 338, 569**Aktive Konfiguration,**

für globale Ressourcen, 652

Aktualisierung,

Statusaktualisierungen im Hintergrund, 390

Aktuelle Execution Nodes einblenden,

XSLT Debugger, 550

Alias,

siehe Globale Ressourcen, 351

Alles markieren,

Befehl, 440

Altova Erweiterungen,

Diagrammfunktionen (siehe Diagrammfunktionen), 1159

Altova Erweiterungsfunktionen,

allgemeine Funktionen, 1160

Diagrammfunktionen (siehe Diagrammfunktionen), 1160

Altova Globale Ressourcen,

siehe Globale Ressourcen, 351

Altova Skripting-Projekte, 697**Altova Support, 29****Altova XML Parser,**

Info, 1183

Altova XSLT 1.0-Prozessor,

Einschränkungen und implementierungsspezifisches

Verhalten, 1130

Altova XSLT 2.0-Prozessor,

allgemeine Informationen, 1132

Informationen, 1132

Altova-Produkte, 29**Altova-Prozessoren,**

in Altova-Produkten, 1184

Anbieter,

Versionskontrolle, 376

Änderungen verfolgen, 667**Angedockte Fenster, 11****Angedocktes Fenster, 682, 683****Anhängen, 486**

Zeile (in Authentic-Ansicht), 562

Anmerkungen zur Programmierung,

Skripting-Projekt, 723

Anpassen,

Kontextmenü, 660

Kontextmenü 'Anpassen', 664

Makros, 662

Menü, 660

Symbolleiste/Menü Befehle, 653

Anpassung, 26**Ansicht,**

ausklappen, 629

Browser-Ansicht, 629

einklappen, 630

Enhanced Grid-Ansicht, 628

erweitern, 630

Ansicht,

- Gehe zu Datei, 631
- Gehe zu Zeile/Zeichen, 631
- Menü, 628
- Optimale Breite, 630
- Schema Design-Ansicht, 629
- Text-Ansicht, 628

Ansicht konfigurieren,

- Dialogfeld in der Content Model-Ansicht, 528

Ansicht wechseln,

- zur Authentic-Ansicht, 152

Anzeige-Einstellungen, 675**Anzeigen, 682, 683****API,**

- aufrufen, 1107
- Dokumentation, 752
- JAVA, 1046
- JAVA Classpath, 1046

Application,

- ActiveDocument, 801
- AddMacroMenuItem, 801
- Application, 801
- ClearMacroMenu, 802
- CurrentProject, 802
- Dialogs, 803
- Documents, 803
- GetDatabaseImportElementList, 804
- GetDatabaseSettings, 804
- GetDatabaseTables, 805
- GetExportSettings, 805
- GetTextImportElementList, 806
- GetTextImportExportSettings, 806
- ImportFromDatabase, 807
- ImportFromSchema, 808
- ImportFromText, 808
- ImportFromWord, 809
- NewProject, 810
- OnBeforeOpenDocument, 799
- OnBeforeOpenProject, 799
- OnDocumentOpened, 800
- OnProjectOpened, 800
- OpenProject, 810
- Parent, 811
- Quit, 811
- ReloadSettings, 812
- RunMacro, 812
- ScriptingEnvironment, 812
- ShowApplication, 813

- ShowForm, 813
- URLDelete, 814
- URLMakeDirectory, 814
- WarningNumber, 815
- WarningText, 815

Applikationsebene,

- Integration von XMLSpy, 1080

Applikations-Events, 714**Arabisch, 1186****Arbeitsspeicher-Anforderungen, 1182****Archiv, 327****Archiv-Ansicht,**

- und EPUB-Dateien, 329

Arial Unicode MS, 1185**ASPRJ-Dateien, 697****ATL,**

- Plug-In Beispieldateien, 741

Atomisierung von Nodes,

- in XPath 2.0- und XQuery 1.0-Auswertung, 1138

ATTLIST Deklaration,

- als Child in Grid-Ansicht hinzufügen, 498
- in Grid-Ansicht anhängen, 492
- in Grid-Ansicht einfügen, 485
- Konvertieren in (Grid-Ansicht), 500

Attribut,

- in Grid-Ansicht anhängen, 487
- Konvertieren in (Grid-Ansicht), 499

Attribute,

- als Child in Grid-Ansicht hinzufügen, 494
- in der Content Model-Ansicht ein- und ausblenden, 57
- in Grid-Ansicht einfügen, 480
- in Schemadefinitionen, 57

Attribute von Schemakomponenten,

- in der Schema-Ansicht definieren, 117

AttributeFormDefault,

- Einstellungen in der Schema Design-Ansicht, 521

Attribut-Eingabehilfe,

- in der Authentic-Ansicht, 157

Attribut-Vorschau, 671**Attributwerte,**

- in der Authentic-Ansicht eingeben, 284

Ausgabefenster,

- Anzeige ein-/aus, 683

Ausgabeformatierung, 667**Aussteigen,**

- XSLT Debugger, 550

Auswerten,

- XPath, 17

Authentic,

- CDATA-Abschnitte, 282
- Daten eingeben, 282
- Daten in einer XML-Datenbank bearbeiten, 557
- Dateneingabe-Steuerelemente, 282
- DB-Daten bearbeiten, 555
- Eingabehilfen, 276
- Elemente anwenden, 279
- Elemente löschen, 279
- Entities, 282
- Kontextmenü, 276
- Markup-Tags anzeigen, 276
- Menü, 554
- Nodes einfügen, 279
- Nodes entfernen, 279
- Nodes hinzufügen, 279
- Sonderzeichen, 282
- Tabellen, 279

Authentic Skripterstellung,

- Sicherheitseinstellungen, 563
- vertrauenswürdige Pfade, 563

Authentic XML, 272**Authentic-Ansicht,**

- als XML/Text einfügen, 161
- Attributwerte eingeben, 284
- Bearbeiten von XML, 175
- Benutzeroberfläche, 151
- Dokumentanzeige, 155
- Eingabehilfen, 157
- Entities einfügen, 285
- Hauptfenster, 155
- Kontextmenüs, 161
- Markup-Anzeige, 155
- neue XML-Datei öffnen, 554
- SPS-Tabellen, 293
- Symbolleistenschaltflächen, 152
- Tabellen (SPS und XML), 293
- Text formatieren, 152
- Tutorial, 274
- Übersicht über die GUI, 151
- Verwendung von XML-Tabellen, 294
- Verwendung wichtiger Features, 288
- Voraussetzungen für das Tutorial, 274
- Vorlage, 275
- wechseln zur, 152, 629
- XML-Dokument drucken, 286
- XML-Dokument öffnen, 275
- XML-Tabellen, 294

- XML-Tabellensymbole, 298

Authentic-Ansicht-Events, 714**AuthenticDataTransfer,**

- dropEffect, 817
- getData, 818
- ownDrag, 818
- type, 818

Authentic-Menü,

- Anzeige von Markup-Symbolen, 152
- Bearbeiten dynamischer Tabellen, 152

AuthenticRange,

- AppendRow, 823
- Application, 823
- CanPerformAction, 823
- CanPerformActionWith, 824
- Close, 824
- CollapsToBegin, 825
- CollapsToEnd, 825
- Copy, 825
- Cut, 825
- Delete, 826
- DeleteRow, 826
- DuplicateRow, 827
- ExpandTo, 828
- FirstTextPosition, 828
- FirstXMLData, 829
- FirstXMLDataOffset, 830
- GetElementAttributeNames, 831
- GetElementAttributeValue, 831
- GetElementHierarchy, 831
- GetEntityNames, 832
- Goto, 833
- GotoNext, 833
- GotoNextCursorPosition, 834
- GotoPrevious, 834
- GotoPreviousCursorPosition, 835
- HasElementAttribute, 835
- InsertEntity, 836
- InsertRow, 836
- IsCopyEnabled, 837
- IsCutEnabled, 837
- IsDeleteEnabled, 837
- IsEmpty, 838
- IsEqual, 838
- IsFirstRow, 838
- IsInDynamicTable, 838
- IsLastRow, 839
- IsPasteEnabled, 839

AuthenticRange,

- IsTextStateApplied, 839
- LastTextPosition, 840
- LastXMLData, 841
- LastXMLDataOffset, 841
- MoveBegin, 842
- MoveEnd, 843
- MoveRowDown, 843
- MoveRowUp, 843
- Parent, 844
- Paste, 844
- PerformAction, 844
- Select, 845
- SelectNext, 846
- SelectPrevious, 846
- SetElementAttributeValue, 847
- SetFromRange, 848
- Text, 849

AuthenticView, 866

- Application, 859
- AsXMLString, 859
- DocumentBegin, 861
- DocumentEnd, 861
- Event, 862
- Goto, 863
- IsRedoEnabled, 864
- IsUndoEnabled, 864
- MarkupVisibility, 864
- OnBeforeCopy, 850
- OnBeforeCut, 851
- OnBeforeDelete, 851
- OnBeforeDrop, 852
- OnBeforePaste, 853
- OnDragOver, 853
- OnKeyboardEvent, 855
- OnMouseEvent, 856
- OnSelectionChanged, 856
- Parent, 864
- Print, 865
- Redo, 865
- Selection, 865
- Undo, 866
- WholeDocument, 867
- XMLDataRoot, 867

Auto-Komplettierung,

- Text-Ansicht Ein/Aus, 670

Autokomplettierung in SQL-Scripts, 587**Automatische Validierung, 668****Automatische Validierung ausschalten, 668**

B

Base Type,

- modifizieren, 133

Bearbeiten,

- Makro Schaltfläche, 664
- Menü, 431

Bearbeiten in der Textansicht, 102**Bearbeitet mit XMLSPY, 667****Bearbeitungsansichten, 96****Beenden,**

- Debuggersitzung, 549

Befehl,

- hinzufügen zu Symbolleiste/Menü, 653
- Kontextmenü, 660
- löschen aus Menü, 660
- Menü zurücksetzen, 660

Befehle,

- Auflistung in Tastaturzuordnung, 686

Befehlszeile, 689**Beispieldateien,**

- Tutorial, 30

Beispiele,

- Pfad installierter Dateien, 28

Benutzerdefiniert,

- Wörterbuch, 635

Benutzerhandbuch, 3, 6**Benutzeroberfläche,**

- Beschreibung, 11
- mittels Plug-In konfigurieren, 738

Beschleunigen,

- Statusaktualisierungen im Hintergrund beschleunigen, 376

Beschreibung der Benutzeroberfläche, 11**Bestellen der Altova-Software, 687****Betriebssysteme für Altova-Produkte, 1182****Bibliotheksmodule,**

- in XQuery-Dokument, 1135

Big-endian, 677**Breakpoint,**

- Dialogfeld, 551

Breakpoint Aus - XSLT Debugger, 551**Breakpoint Ein - XSLT Debugger, 551****Breakpoints,**

- im XSLT/XQuery Debugger, 263

Breakpoints dialog, 251**Browser, 671**

Ansicht, 629

Menü, 633

Browser-Ansicht,

Beschreibung, 164

Inhalt aktualisieren, 633

Laden einer Seite abbrechen, 633

neues Fenster, 633

Schriftgröße, 633

weiter, 633

zurück, 633

Browser-Fenster,

im Fenster "Datenbankabfrage", 584

C

C#,

integration of XMLSpy, 1085

Integration von XMLSpy, 1085

Call Stack-Fenster,

in XSLT/XQuery Debugger, 260

CDATA,

als Child in Grid-Ansicht hinzufügen, 494

in Grid-Ansicht anhängen, 488

in Grid-Ansicht einfügen, 481

Konvertieren in (Grid-Ansicht), 500

CDATA-Abschnitte,

in der Authentic-Ansicht einfügen, 288

Child anhängen,

in Grid-Ansicht, 493

Class,

JAVA, 1046

Class ID,

in XMLSpy Integration, 1080

ClassPath statement, 1046**CodeGeneratorDlg,**

Application, 868

CPPSettings_DOMType, 869

CPPSettings_LibraryType, 870

CPPSettings_UseMFC, 870

CSharpSettings_ProjectType, 871

OutputPath, 871

OutputPathDialogAction, 871

OutputResultDialogAction, 871

Parent, 872

ProgrammingLanguage, 872

PropertySheetDialogAction, 872

TemplateFileName, 873

Codepage, 673**Collapse,**

unselected, 630

Collations,

in XPath 2.0, 1138

in XQuery-Dokument, 1135

COM-API,

Dokumentation, 752

Comment,

als Child in Grid-Ansicht hinzufügen, 494

in Grid-Ansicht anhängen, 488

in Grid-Ansicht einfügen, 481

Konvertieren in (Grid-Ansicht), 500

Comparisons,

of files and directories, 373

ComplexType,

Definition erweitern, 48

ComplexTypes,

in Schema-Definitionen, 48

Compositor,

in Schema View, 121

Connecting to data source,

via a global resource, 348, 579

Constraints, 138**Content Model,**

Attribute ein- und ausblenden, 57

Diagramm speichern, 523

einfaches Content Model erstellen, 36

Content Model View, 121**Content Model-Ansicht, 32**

konfigurieren, 528

Content models,

of schema components, 121

Copy command, 432**Copyright-Informationen, 1188****count() function in XPath 2.0,**

siehe fn:count(), 1138

count()-Funktion,

in XPath 1.0, 1130

CPU,

Belastung - Statusaktualisierungen im Hintergrund beschleunigen, 376

CR&LF, 667**CSS, 316**

Autokomplettierung, 319

CSS, 316

- Dokumentübersicht, 319
- Eigenschaften, 319
- Info-Fenster, 319
- Syntaxfärbung, 319

CSS-Info-Fenster, 319**CSV-Datei,**

- als XML importieren, 606

CustomCatalog, 506**Cut command, 432****CVS, 376**

D

Darstellung, 671**Database,**

- connecting via global resource, 348, 579

Database/Table View,

- how to use, 82

DatabaseConnection,

- ADOConnection, 874
- AsAttributes, 874
- CreateMissingTables, 875
- CreateNew, 875
- DatabaseKind, 875
- ExcludeKeys, 876
- File, 876
- IncludeEmptyElements, 877
- NumberDateTimeFormat, 877
- ODBCConnection, 877
- SQLSelect, 879
- TextFieldLen, 879

Datei,

- als Mail senden, 427
- Druckoptionen, 428
- Kodierung, 421
- Neu laden, 421
- Öffnen, 416
- Optionen beim Öffnen, 667
- Register, 667
- schließen, 422
- speichern, 422
- Standard-Encoding, 677

Datei (Menü), 412**Datei abrufen,**

- Versionskontrolle, 455

Dateiarten, 668**Dateien,**

- Vergleichsoptionen, 649
- Zu Versionskontrolle hinzufügen, 460
- zuletzt verwendete, 430
- zwei vergleichen, 644

Dateien speichern,

- Kodierung, 421

Dateien vergleichen, 644

- Optionen, 649

Dateierweiterungen,

- anpassen, 506
- Erweiterungen als Dateityp definieren, 197
- für XQuery-Dateien, 238

Dateipfade,

- in XML-Dokument einfügen, 170

Dateivergleiche, 374**Datenbank,**

- ADO-Verbindungen, 338, 569
- bestehende Verbindungen, 337, 568
- Datensätze bearbeiten, 592
- DB auf Basis von Schema erzeugen, 619
- ODBC-Verbindungen, 342, 573
- Oracle XML DB, 604
- Struktur als XML-Schema importieren, 614
- Verbindung herstellen, 335, 564
- Verbindungsassistent, 336, 566
- XML-Daten exportieren, 624

Datenbank abfragen (Befehl), 580**Datenbankabfrage,**

- Abfrage erstellen, 591
- Browser-Fenster im Fenster Datenbankabfrage, 584
- Ergebnisse, 592
- Ergebnisse bearbeiten, 592
- Meldungsregister, 592
- Verbindung zur DB für Abfrage herstellen, 582

Datenbankabfrage (Fenster), 580**Datenbanken,**

- in Authentic-Ansicht bearbeiten, 555
- siehe auch DB, 300
- und globale Ressourcen, 363
- Unterstützung in XMLSpy, 350

Datenbanken in XMLSpy, 333**Datensätze bearbeiten, 592****Datentypen,**

- für DB, die von XML-Schemas generiert wurden, 1172
- in XPath 2.0 und XQuery 1.0, 1138
- Konvertierung von DB in XML-Schema, 1164

Datentypen,

siehe auch XML-Schema-Datentypen, 1165

Datumswahl,

in der Authentic-Ansicht verwenden, 308

DB,

Abfragen erstellen, 301

Abfragen in der Authentic-Ansicht, 300

Anzeige in der Authentic-Ansicht filtern, 301

Bearbeiten in der Authentic-Ansicht, 300

Datentypen von XML-Schema, 1172

Datentyp-Konvertierung in XML-Schema, 1164

in der Authentic-Ansicht bearbeiten, 306

Navigieren in Tabellen in der Authentic-Ansicht, 301

Parameters in DB-Abfragen, 301

DB XML,

XML Schemas zuweisen für IBM DB2, 598

XML-Schemas verwalten, für IBM DB2, 595

db2-fn:sqlquery, 244**db2-fn:xmlcolumn, 244****Debuggen von Makros, 721****Debugger,**

aktuelle Execution Nodes einblenden, 550

aussteigen, 550

Breakpoint Ein/Aus, 551

Breakpoint einfügen/löschen, 550

Debug-Fenster, 552

Dialogfeld Breakpoints, 551

einsteigen, 549

Einstellungen, 552

enable/disable tracepoint, 551

insert/remove tracepoint, 550

Sitzung beenden, 549

überspringen, 550

XSLT Debugger erneut starten, 549

XSLT Debugger starten, 549

XSLT Debugger stoppen, 549

Debugger -Fenster,

Anordnung, 262

deep-equal() function in XPath 2.0,

siehe fn:deep-equal(), 1138

Default Functions Namespace,

für XPath 2.0- und XQuery 1.0-Ausdrücke, 1138

in XSLT 2.0 Stylesheets, 1132

Delete,

Application.URLDelete, 814

Delete command, 432**Details-Eingabehilfe, 36****Diagramm anzeigen, 532****Diagramme,**

Beispiel (einfach), 184

mehrere Diagrammregister, 181

neu laden, 181

Übersicht, 181

Dialogs,

Application, 880

CodeGeneratorDlg, 880

DTDSchemaGeneratorDlg, 881

FileSelectionDlg, 880

GenerateSampleXMLDlg, 881

Parent, 881

SchemaDocumentationDlg, 881

Diffdog,

für den Vergleich konfigurieren, 390

directories,

creating with Application.URLMakeDirectory, 814

Directory comparisons, 373**Disable,**

tracepoint - XSLT debugger, 551

DocEditEvent (obsolete),

altKey (obsolete), 987

altLeft (obsolete), 988

button (obsolete), 989

cancelBubble (obsolete), 990

clientX (obsolete), 991

clientY (obsolete), 992

ctrlKey (obsolete), 993

ctrlLeft (obsolete), 994

dataTransfer (obsolete), 995

fromElement (obsolete), 996

keyCode (obsolete), 996

propertyName (obsolete), 997

repeat (obsolete), 997

returnValue (obsolete), 997

shiftKey (obsolete), 998

shiftLeft (obsolete), 999

srcElement (obsolete), 1000

type (obsolete), 1001

DocEditView (obsolete),

ApplyTextState (obsolete), 1007

CurrentSelection (obsolete), 1008

EditClear (obsolete), 1008

EditCopy (obsolete), 1009

EditCut (obsolete), 1009

EditPaste (obsolete), 1010

EditRedo (obsolete), 1010

EditSelectAll (obsolete), 1011

DocEditView (obsolete),

EditUndo (obsolete), 1011
 event (obsolete), 1012
 GetAllowedElements (obsolete), 1012
 GetNextVisible (obsolete), 1015
 GetPreviousVisible (obsolete), 1016
 IsEditClearEnabled (obsolete), 1017
 IsEditCopyEnabled (obsolete), 1017
 IsEditCutEnabled (obsolete), 1018
 IsEditPasteEnabled (obsolete), 1018
 IsEditRedoEnabled (obsolete), 1019
 IsEditUndoEnabled (obsolete), 1019
 IsRowAppendEnabled (obsolete), 1020
 IsRowDeleteEnabled (obsolete), 1020
 IsRowDuplicateEnabled (obsolete), 1021
 IsRowInsertEnabled (obsolete), 1021
 IsRowMoveDownEnabled (obsolete), 1023
 IsRowMoveUpEnabled (obsolete), 1023
 IsTextStateApplied (obsolete), 1024
 IsTextStateEnabled (obsolete), 1024
 LoadXML (obsolete), 1025
 RowAppend (obsolete), 1027
 RowDelete (obsolete), 1027
 RowDuplicate (obsolete), 1028
 RowInsert (obsolete), 1028
 RowMoveDown (obsolete), 1029
 RowMoveUp (obsolete), 1029
 SaveXML (obsolete), 1030
 SelectionMoveTabOrder (obsolete), 1031
 SelectionSet (obsolete), 1032
 XMLRoot (obsolete), 1033

DOCTYPE Deklaration,

als Child in Grid-Ansicht hinzufügen, 497
 in Grid-Ansicht anhängen, 491
 in Grid-Ansicht einfügen, 484
 Konvertieren in (Grid-Ansicht), 500

Document,

Application, 887
 AssignDTD, 888
 AssignSchema, 888
 AssignXSL, 888
 AssignXSLFO, 889
 AuthenticView, 889
 Close, 890
 ConvertDTDOrSchema, 890
 CreateChild, 892
 CreateSchemaDiagram, 892
 CurrentViewMode, 893

DataRoot, 893
 DocEditView, 893
 Encoding, 894
 EndChanges, 894
 ExecuteXQuery, 894
 ExportToDatabase, 895
 ExportToText, 895
 FullName, 896
 GenerateDTDOrSchema, 897
 GenerateProgramCode, 898
 GenerateSampleXML, 898
 GenerateSchemaDocumentation, 898
 GetExportElementList, 900
 GetPathName, 901
 GridView, 901
 IsModified, 902
 IsValid, 902
 IsWellFormed, 903
 Name, 904
 OnBeforeCloseDocument, 885
 OnBeforeSaveDocument, 885
 OnBeforeValidate, 886
 OnCloseDocument, 887
 OnViewActivation, 887
 Path, 904
 RootElement, 904
 Save, 905
 SaveAs, 905
 Saved, 905
 SaveInString, 906
 SaveToURL, 906
 SetActiveDocument, 906
 SetEncoding, 907
 SetExternalIsValid, 908
 SetPathName, 908
 StartChanges, 908
 SwitchViewMode, 909
 Title, 909
 TransformXSL, 910
 TransformXSLFO, 910
 UpdateViews, 911
 UpdateXMLData, 911
 XQuery, 894

Documents,

Count, 912
 Item, 912
 NewAuthenticFile, 912
 NewFile, 913

Documents,

- NewFileFromText, 913
- OpenAuthenticFile, 914
- OpenFile, 914
- OpenURL, 914
- OpenURLDialog, 915

Dokument,

- Oracle XML DB durchsuchen, 604
- Rechtschreibprüfung, 635

Dokumentation,

- von XML-Schema, 523

Dokumente im Hauptfenster, 12**Dokumentebene,**

- Beispiele für die Integration von XMLSpy, 1084
- Integration des XMLSpyControl, 1083
- Integration von XMLSpy, 1083, 1084

Dokument-Events, 714**Dokumentieren,**

- Schema, 62

Drag & Drop,

- Elemente hinzufügen, 41

Drucken,

- von der Authentic-Ansicht, 286

Druckereinrichtung, 429**DTD, 667, 668**

- Attlist-Deklaration, 485, 492, 498, 500
- Element-Deklaration, 485, 492, 497, 500
- Entities inkludieren, 511
- Entity-Deklaration, 485, 492, 498, 501
- Entwurf anhand von XML-Datei erzeugen, 518
- Gehe von XML-Dokument zu Definition, 512
- Gehe von XML-Dokument zu DTD, 511
- generieren anhand von XML-Schema (Enterprise und Professional Edition), 192
- in UML konvertieren, 517
- Konvertieren in XML-Schema, 514
- Menübefehle zu, 510
- Notation Deklaration, 501
- Notation-Deklaration, 485, 492, 498
- Referenz von XML zu externer DTD, 484
- von XML-Dokument generieren, 512
- zu XML-Dokument zuweisen, 510

DTD/Schema,

- Menü, 510

DTDs, 189

- in der Grid-Ansicht bearbeiten (Enterprise und Professional Edition), 190
- in der Textansicht bearbeiten, 190

- in XML-Schemas konvertieren (Enterprise und Professional Edition), 190

- XML-Dokument generieren, 190

DTDSchemaGeneratorDlg,

- Application, 916
- AttributeTypeDefinition, 916
- DTDSchemaFormat, 917
- FrequentElements, 917
- GlobalAttributes, 917
- MaxEnumLength, 917
- MergeAllEqualNamed, 917
- OnlyStringEnums, 918
- OutputPath, 918
- OutputPathDialogAction, 918
- Parent, 918
- ResolveEntities, 919
- TypeDetection, 919
- ValueList, 919

Durchsuchen,

- Oracle XML DB, 604

Dynamische (SPS) Tabellen in der Authentic-Ansicht,

- Verwendung, 293

Dynamische Tabellen,

- bearbeiten, 152

E

Eclipse,

- und XMLSpy, 402

Eclipse Plattform,

- und XMLSpy Integrationspaket, 403
- XMLSpy Debugger-Perspektive, 410

Eclipse-Plattform,

- XMLSpy Perspektive, 407

Eigenschaften und Events (Fenster), 698**Ein- und auskommentieren,**

- in XML-Dokumenten in der Textansicht, 170

Einfügen,

- als Text, 288
- als XML, 288
- Breakpoint - XSLT Debugger, 550
- in Grid-Ansicht, 479
- Zeile (in Authentic-Ansicht), 562

Einfügen als,

- Text, 161
- XML, 161

Eingabehilfe, 683

Details, 36

Eingabehilfen, 16, 31

aktualisieren, 509
Anzeige ein/aus, 683
für XML-Dokumente, 177
für XQuery, 239
in der Schema-Ansicht, 134
in Grid-Ansicht, 80

Eingabehilfen aktualisieren,

Befehl, 509

Eingabehilfen in der Textansicht, 105**Eingabemethoden, 1185****Eingabetaste,**

drücken, 313

Einrücken,

in Text-Ansicht, 439

Einrücklinien, 631**Einrückung in der Textansicht, 98****Einsteigen,**

XSLT Debugger, 549

Einstellung für automatisches Makro, 717**Einstellungen, 26, 667**

Skripting, 679
XSLT Debugger, 552

Einstellungen für den Dateivergleich, 649**Einstellungen für Textansicht (Dialogfeld), 631****Element,**

als Child in Grid-Ansicht hinzufügen, 494
in Grid-Ansicht anhängen, 487
in Grid-Ansicht einfügen, 480
Konvertieren in (Grid-Ansicht), 499

ELEMENT Deklaration,

als Child in Grid-Ansicht hinzufügen, 497
Konvertieren in (Grid-Ansicht), 500

ELEMENT-Deklaration,

in Grid-Ansicht anhängen, 492

ELEMENT-Deklaration in DTD,

in Grid-Ansicht einfügen, 485

Elemente,

Inhalt einschränken, 44
optional machen, 44

Element-Eingabehilfe,

in der Authentic-Ansicht, 157

ElementFormDefault,

Einstellungen in der Schema Design-Ansicht, 521

ElementList,

Count, 920

Item, 920

RemoveElement, 920

ElementListItem,

ElementKind, 921
FieldCount, 921
Name, 921
RecordCount, 921

Elementtyp,

in XML-Dokument festlegen, 69

E-Mail,

Dateien senden, 427

Enable tracepoint - XSLT debugger, 551**Encoding, 421**

in XQuery-Dokument, 1135
Standard, 677

Endbenutzer-Lizenzvereinbarung, 1188, 1192**Enhanced Grid-Ansicht, 628**

siehe Grid-Ansicht, 71

Entities,

in der Authentic-Ansicht definieren, 288, 309
in der Authentic-Ansicht einfügen, 288
in der Authentic-Ansicht Entities einfügen, 285
in XML Schema-basierten XML-Dokumenten, 168

Entities-Eingabehilfe,

in der Authentic-Ansicht, 157

ENTITY Deklaration,

als Child in Grid-Ansicht hinzufügen, 498
in Grid-Ansicht anhängen, 492
Konvertieren in (Grid-Ansicht), 501

Entity-Deklaration,

in Grid-Ansicht einfügen, 485

Enumerationen,

für Attribute definieren, 57
in XMLSpyControl, 1124

Enumerations,

SPYAttributeTypeDefinition, 1035
SPYAuthenticActions, 1035
SPYAuthenticDocumentPosition, 1035
SPYAuthenticElementActions, 1036
SPYAuthenticElementKind, 1036
SPYAuthenticMarkupVisibility, 1036
SPYDatabaseKind, 1037
SPYDialogAction, 1037
SPYDOMType, 1037
SPYDTDSchemaFormat, 1037
SPYEncodingByteOrder, 1038
SPYExportNamespace, 1038
SPYFrequentElements, 1038

Enumerations,

- SPYKeyEvent, 1039
- SPYLibType, 1039
- SPYLoading, 1039
- SPYMouseEvent, 1039
- SPYNumberDateTimeFormat, 1040
- SPYProgrammingLanguage, 1040
- SPYProjectItemTypes, 1041
- SPYProjectType, 1041
- SPYSampleXMLGenerationOptimization, 1041
- SPYSampleXMLGenerationSchemaOrDTDAssignment, 1041
- SPYSchemaDefKind, 1042
- SPYSchemaDocumentationFormat, 1042
- SPYTextDelimiters, 1043
- SPYTextEnclosing, 1043
- SPYTypeDetection, 1043
- SPYURLTapes, 1043
- SPYViewModes, 1044
- SPYVirtualKeyMask, 1044
- SPYXMLDataKind, 1045

EPUB-Dateien, 329**Erneut starten,**

- XSLT Debugger, 549

Ersetzen, 22

- Text, 440
- Text in Dokument, 444
- Text in mehreren Dateien, 445

Erstellen,

- Neues XML Dokument, 67

Erweitern,

- alles, 630

Erweiterte Validierung, 210

- in SchemaAgent, 537

Erweiterungsfunktionen für XSLT und XQuery, 1143**Erweiterungsfunktionen in .NET für XSLT und XQuery,**

- siehe .NET Erweiterungsfunktionen, 1151

Erweiterungsfunktionen in Java für XSLT und XQuery,

- siehe Java-Erweiterungsfunktionen, 1143

Erweiterungsfunktionen in MSXSL Scripts, 1157**Erzeugen,**

- DB auf Basis von Schema, 619

Evaluierungs-Keycode,

- für Ihre Altova-Software, 687

Evaluierungszeitraum,

- bei Altova Software-Produkten, 1191
- für Altova-Software-Produkte, 1189
- von Altova Software-Produkten, 1188

Event, 799, 800, 850, 851, 852, 853, 855, 856, 885, 886, 887, 940, 941, 942

Event Handler,

- im Skripting-Projekt, 714
- Übersicht, 702

Events, 714

- und Event Handler, 704

Excel 2007, 327**Explorer, 668****ExportSettings,**

- CreateKeys, 922
- ElementList, 922
- EntitiesToText, 922
- ExportAllElements, 923
- FromAttributes, 923
- FromSingleSubElements, 923
- FromTextValues, 923
- IndependentPrimaryKey, 924
- Namespace, 924
- SubLevelLimit, 924

Extern geparste Entitäten, 668**ExternalID Deklaration,**

- als Child in Grid-Ansicht hinzufügen, 497
- in Grid-Ansicht anhängen, 491
- in Grid-Ansicht einfügen, 484
- Konvertieren in (Grid-Ansicht), 500

Externe Funktionen,

- in XQuery-Dokument, 1135

Externer XSL Prozessor, 677**Extras,**

- Menü, 635

F

Farbe, 673, 674**Farben,**

- Register, 675
- Tabelle, 675

Favoriten, 685**Fenster,**

- Anzeige verwalten, 11
- automatisch ausblenden, 11
- Eingabehilfe, 683
- frei schwebend, andocken, als Register, 11
- horizontal nebeneinander, 682
- Info, 683

Fenster,

- Menü, 682
- öffnen, 684
- Projekt, 682, 683
- überlappend, 682
- Übersicht, 31
- vertikal anordnen, 682

Fenster "Trace", 266**Fenster Ablaufverfolgung,**

- im XSLT/XQuery Debugger, 262

Fenster automatisch ausblenden, 11**File comparisons, 373****FileSelectionDlg,**

- Application, 925
- DialogAction, 925
- FullName, 925
- Parent, 926

Finden,

- siehe Suchen, 444

fn:base-uri in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

fn:collection in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

fn:count() in XPath 2.0,

- und Whitespace, 1138

fn:current-date in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

fn:current-dateTime in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

fn:current-time in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

fn:data in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

fn:deep-equal() in XPath 2.0,

- und Whitespace, 1138

fn:id in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

fn:idref in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

fn:index-of in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

fn:in-scope-prefixes in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

fn:last() in XPath 2.0,

- und Whitespace, 1138

fn:lower-case in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

fn:normalize-unicode in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

fn:position() in XPath 2.0,

- und Whitespace, 1138

fn:resolve-uri in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

fn:static-base-uri in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

fn:upper-case in XPath 2.0,

- Unterstützung in Altova-Prozessoren, 1139

Formulare,

- aufrufen, 704
- benennen, 708
- Eigenschaften, 708
- in Skripting-Projekten, 708
- neues erstellen, 708
- Übersicht, 702
- und Event-Behandlung, 711
- und Formularobjekte, 709
- und vordefinierte Befehle, 724

Formularobjekteigenschaften, 709**Formularobjektpalette, 698****Frei schwebende Fenster, 11****Freigegeben,**

- aus Versionskontrolle, 461

functions,

- XPath 2.0 and XQuery 1.0, 1138

Funktionen,

- siehe unter XSLT 2.0-Funktionen, 1134

Funktionstasten, 655

G

Gehe zu Datei, 631**Gehe zu Zeile/Zeichen, 631****Gemischte Markup-Symbole in (in Authentic)-Ansicht, 561****Generate Sample XML, 1035, 1041****GenerateSampleXMLDlg,**

- Application, 936
- FillWithSampleData, 938
- NonMandatoryAttributes, 937
- NonMandatoryElements, 937
- Parent, 937
- RepeatCount, 937
- TakeFirstChoice, 937

Generieren,

Generieren,

DB-Struktur auf Basis von Schema, 619

Global, 1185**Globale,**

Einstellungen, 667

Globale Deklarationen, 706

Übersicht, 702

Globale Elemente,

in XML-Schemas verwenden, 55

Globale Elemente anzeigen, 531**Globale Komponenten,**

in der Schema-Übersicht erstellen, 117

Globale Ressourcen, 351

definieren, 352, 651

Konfigurationen ändern, 364

Konfigurationen kopieren, 359

Symbolleiste aktivieren, 654

Typ Datei definieren, 354

Typ Datenbank definieren, 357

Typ Ordner definieren, 356

verwenden, 360, 363, 364

vom Typ Datei und Ordner verwenden, 360

Globale Schemakomponenten,

suchen und umbenennen, 223

Globales Skripting-Projekt,

von XMLSpy, 697

Grafikformate,

in der Authentic-Ansicht, 312

Grammatik, 668**Graue Leiste, 629, 630****Grid View,**

and Table View, 82

Grid-Ansicht, 107, 628, 630

Attribut / Element anhängen, 80

bearbeiten, 108, 503, 504

Dateneingaben, 71

Eingabehilfen, 113

Eingabehilfen verwenden, 80

Tabellen, 109

und Datenimport/-export, 109

XML-Dokumente bearbeiten (Enterprise und Professional Edition), 172

zu Tabellenansicht wechseln, 501

Grid-Ansicht-Events, 714**Grid-Schriftarten, 673****GridView,**

CurrentFocus, 943

Deselect, 943

IsVisible, 943

OnBeforeDrag, 940

OnBeforeDrop, 941

OnBeforeStartEditing, 941

OnEditingFinished, 942

OnFocusChanged, 942

Select, 943

SetFocus, 943

Größe, 674**Große Markup-Symbole anzeigen, 152, 155****Große Markup-Symbole in (in Authentic)-Ansicht, 561**

H

Hauptfenster, 12, 31**Hebräisch, 1186****Hilfe,**

Index, 685

Inhaltsverzeichnis, 685

Menü, 685

Suchen, 686

Tastaturzuordnungen, 686

Hilfesystem, 685**Hin- und Herschalten, 683****Hintergrund,**

Statusaktualisierungen, 390

Statusaktualisierungen - Intervall steigern, 376

Hintergrundinformationen, 1181**Hin- und Herschalten, 682****Hinzufügen,**

Dateien zu Projekt, 466

Ordner zu Projekt, 467

Hiragana, 1185**Höher stufen,**

Befehl, 503

HTML, 316

Integration von XMLSpy, 1089

HTML-Beispiel,

für XMLSpyControl Integration, 1080, 1081, 1082

HTML-Dokumente,

bearbeiten, 317

Info-Fenster, 317

HTML-Info-Fenster, 317

I

IBM DB2,

- Schemaverwaltung und -zuweisung, 595
- Verwalten von XML-Schemas, 595
- XML Schemas einer XML-Datei zuweisen, 598

Identity Constraint,

- in der Content Model-Ansicht ein- und ausblenden, 57

Identity Constraints, 138

- in der Schema-Ansicht definieren, 117

Implementierungsspezifisches Verhalten,

- von XSLT 2.0-Funktionen, 1134

Implizite Zeitzone,

- und XPath 2.0-Funktionen, 1138

Import,

- Datenbankdaten basierend auf XML-Schema, 618
- MS Word Dokument als XML, 614

Import/Export von Daten,

- und Grid-Ansicht, 109

Importieren,

- Datenbankstruktur als XML-Schema, 614
- Textdatei als XML, 606

In Dateien suchen (Befehl), 445

In Dateien suchen (Fenster), 22

In Element einbinden,

- Befehl, 504

In Schemas suchen, 212

- Begriff ersetzen, 213
- Begriff suchen, 213
- Bereich definieren, 220
- Ergebnisse, 222
- Fenster, 222
- globale Komponenten, 223
- Suche auf Komponenten einschränken, 215
- Suche durch Eigenschaft und Eigenschaftswert einschränken, 217
- Suchen- und Ersetzen-Befehl ausführen, 221
- Umbenennen von globalen Komponenten, 223

In Schemas suchen (Fenster), 23

Index,

- Hilfe, 685

Info,

- Fenster, 31

Infofenster, 31

Info-Fenster, 15, 683

- für CSS-Dokumente, 319

- für HTML-Dokumente, 317

Info-Fenster in XSLT/XQuery Debugger, 261

Info-Fenster, XSLT-Register,

- Beschreibung, 234
- siehe auch XSL-Übersicht, 234
- und Erstellen von ZIP-Ordern, 234
- und Projekte, 234
- und XSLT-Dokumente, 234

Informationsfenster,

- anordnen, 262

Inhaltsverzeichnis, 685

Insert,

- tracepoint - XSLT debugger, 550

Installation,

- Pfad von Beispieldateien, 28

Installieren,

- Versionskontrollsysteme, 383

Instanzen,

- Anzahl, 36

Integrating,

- XMLSpy in Applikationen, 1079

Intelligentes Editieren, 670

Internet, 688

Internet-Verwendung,

- in Altova-Produkten, 1187

ISO/IEC 10646, 1185

J

JAVA,

- API, 1046
- ClassPath, 1046

Java Erweiterungsfunktionen,

- Datentypkonvertierungen, Java in Xpath/XQuery, 1151
- Instanzmethoden, Instanzfelder, 1149
- Konstrukturen, 1148
- statische Methoden, statische Felder, 1149

Java extension functions,

- datatype conversions, XPath/XQuery to Java, 1150

Java-Erweiterungsfunktionen,

- benutzerdefinierte JAR-Dateien, 1147
- benutzerdefinierte Klassendateien, 1145
- für XSLT und XQuery, 1143
- Übersicht, 1143

JRE,

JRE,

für XMLSpy Plugin für Eclipse, 403

JSON, 316

bearbeiten in der Textansicht, 324

von und in XML konvertieren, 626

K

Kanji, 1185**Kapitel, 685****Katakana, 1185****Katalog,**

Oasis XML, 506

Kataloge, 197**Keycodes,**

für Ihre Altova-Software, 687

Klappleiste, 631**Klappleiste in der Textansicht, 100****Kleine Markup-Symbole anzeigen, 152, 155****Kleine Markup-Symbole in (in Authentic)-Ansicht, 561****Kodierung,**

von Dateien, 421

Kommentare in SQL-Scripts, 587**Komponenten-Definitionen,**

wiederverwenden, 48

Komponentennavigator, 134**Kompositor,**

für Sequences, 36

Konfigurationen,

einer globalen Ressource, 352, 652

Konfigurationen in Globalen Ressourcen, 364**Konfigurieren,**

XMLSPY-Benutzeroberfläche, 738

Kontextfenster,

im XSLT/XQuery Debugger, 258

Kontextmenü,

anpassen, 664

Befehle, 660

Kontextmenüs,

in der Authentic-Ansicht, 161

Konvertieren,

Datenbankschema in XML-Schema, 614

Menü, 606

MS Word Daten in XML, 614

Oracle XML DB, 604

Schema in DB-Struktur, 619

Textdatei in XML, 606

Konvertieren in,

in Grid-Ansicht, 499

Kopieren,

als strukturierten Text, 433

als XML-Text, 432

Zeile (in Authentic-Ansicht), 562

Kürzel, 655

anzeigen in Tooltip, 664

zuweisen/löschen, 655

L

last() function in XPath 2.0,

siehe fn:last(), 1138

last()-Funktion,

in XPath 1.0, 1130

Leere Elemente, 668**Leere Zeilen,**

in XML-Dokumenten in der Textansicht, 170

Lesezeichen, 685

einfügen und entfernen, 447

navigieren, 447

Lesezeichen in der Textansicht, 100**Lesezeichen in SQL-Scripts, 587****Lesezeichenleiste, 631****Links,**

in der Authentic-Ansicht verwenden, 288

Little-endian, 677**Lizenz, 1192**

für Ihre Altova-Software, 687

Informationen, 1188

Lizenzüberwachung,

bei Altova-Produkten, 1190

loading, 914**Löschen,**

Befehl aus Symbolleiste, 653

Befehl im Kontextmenü, 660

Breakpoint - XSLT Debugger, 550

Kürzel zu Befehl, 655

Symbol aus Symbolleiste, 653

Symbolleiste, 654

Zeile (in Authentic-Ansicht), 562

M

MainCatalog, 506

Makro,

- Hinzufügen zu Menü/Symbolleiste, 662
- Schaltfläche bearbeiten, 664

Makros,

- ausführen, 719
- Ausführung, 704
- debuggen, 721
- Funktionen für Makros in globalen Deklarationen, 706
- im Skript-Editor als automatisches Makro definieren, 717
- mit dem Skript-Editor bearbeiten, 717
- mit dem Skript-Editor erstellen, 717
- Übersicht, 702
- Verwendung im Skripting-Projekt, 717

Makros anzeigen,

- Skripting, 644

MapForce, 518

Markup,

- in der Authentic-Ansicht, 152

Markup-Code (in Authentic)-Ansicht,

- ausblenden, 561
- kleine/große/gemischte einblenden, 561

Markup-Symbole,

- in der Authentic-Ansicht, 155

Markup-Symbole (in Authentic)-Ansicht ausblenden, 561

Markup-Symbole ausblenden, 152, 155

Maximale Zellenbreite, 671

Meldungen (Fenster), 16

Meldungsfenster,

- in XSLT/XQuery Debugger, 260

Menü,

- anpassen, 660
- Ansicht, 628
- Bearbeiten, 431
- Befehl hinzufügen/löschen, 653
- Befehle löschen, 660
- Browser, 633
- Extras, 635
- Fenster, 682
- Hilfe, 685
- Konvertieren, 606
- Makro hinzufügen, 662

- Projekt, 449

- Standard/XMLSPY, 660

Menüleiste, 24

Menüs, 26

Microsoft Office 2007, 327

Microsoft® SharePoint® Server, 471

MIME, 668

Mixed Markup-Symbole anzeigen, 152, 155

MS Access,

- Datentypen von XML-Schema, 1173
- Konvertierung von Datentypen in XML-Schema
- Generierung von DB, 1165

MS SQL Server,

- Datentypen von XML-Schema, 1175
- Konvertierung von Datentypen in XML-Schema
- Generierung von DB, 1166
- Schema Extensions, 533, 534
- Schema-Einstellungen, 535

MS Visual Source Safe, 376

MSXML, 677

msxsl:Script, 1157

Multi-user, 667

MySQL,

- Datentypen von XML-Schema, 1177
- Konvertierung von Datentypen in XML-Schema
- Generierung von DB, 1167

N

Nach oben/nach unten,

- Zeile verschieben (Authentic-Ansicht), 562

Named Schema-Beziehungen,

- MS SQL Server Schema-Einstellungen, 534

Named Templates, 231

Named templates aufrufen, 231

Namespace Präfix,

- in Grid-Ansicht einfügen, 509

Namespaces,

- Einstellungen in der Schema Design-Ansicht, 521
- im XSLT 2.0 Stylesheet, 1132
- in Schemas, 35
- in XQuery-Dokument, 1135

Navigationsverlauf, 149

Navigieren,

- Kürzel in Schema, 60

Nebeneinander, 671

- horizontal, 682

Nebeneinander, 671

vertikal, 682

Neu laden, 667

geänderte Dateien, 421

Neue Datei,

erstellen, 412

Neue Funktionen, 8**Neues XML-Dokument,**

erstellen, 67

Nicht lateinischen Schriftsysteme, 1185**Nicht-XML Dateien, 668****Node,**

aktuellen Execution Node einblenden, 550

NOTATION Deklaration,

als Child in Grid-Ansicht hinzufügen, 498

in Grid-Ansicht anhängen, 492

in Grid-Ansicht einfügen, 485

Konvertieren in (Grid-Ansicht), 501

O

OASIS,

XML-Katalog, 506

Objektsuche,

im Fenster "Datenbankabfrage", 584

ODBC,

Konvertierung von Datentypen in XML-Schema
Generierung von DB, 1169

ODBC-Verbindungen, 342, 573**Office Open XML, 327****Öffnen, 416****OOXML,**

siehe Office Open XML, 327

Optimale Breite, 630, 671**Optionale Elemente,**

erstellen, 44

Optionen beim Öffnen,

Datei, 667

Oracle,

Datentypen von XML-Schema, 1179

Konvertierung von Datentypen in XML-Schema
Generierung von DB, 1168

Schema-Extensions, 532

Oracle XML Datenbank, 604**Oracle XML DB,**

Datenbank durchsuchen, 604

Ordner abrufen,

Versionskontrolle, 456

OrgChart.sps,

im Tutorial der Authentic-Ansicht, 274

OrgChart.xml,

im Tutorial der Authentic-Ansicht, 274

OrgChart.xsd,

im Tutorial der Authentic-Ansicht, 274

P

Parameter,

in DB-Abfragen, 301

Übergabe an Stylesheet über Schnittstelle, 544

Parent, 904**Parser,**

in Altova-Produkte integrierter, 1183

XSLT, 677

Paste command, 432**PDF,**

Transformieren in PDF in XMLSpy, 228

Plattformen für Altova-Produkte, 1182**Plug-in, 663**

ATL-Beispieldateien, 741

Konfiguration der Benutzeroberfläche, 738

Registrierung, 736

XMLSPY, 735

Position,

Zeichen, 631

Zeile, 631

position() function,

in XPath 1.0, 1130

position() function in XPath 2.0,

siehe fn:position(), 1138

PowerPoint 2007, 327**Pretty-Print,**

Text-Ansicht, 439

Processing Instruction,

als Child in Grid-Ansicht hinzufügen, 495

in Grid-Ansicht anhängen, 488

in Grid-Ansicht einfügen, 481

Konvertieren in (Grid-Ansicht), 500

Programmeinstellungen, 667**Projekt,**

aktive Dateien hinzufügen, 467

Dateien zu Projekt hinzufügen, 466

Eigenschaften, 475

Projekt,

- externe Ordner hinzufügen, 467
- externen Webordner hinzufügen, 471
- Hinzufügen aktiver und verwandter Dateien, 467
- Hinzufügen zu Versionskontrolle, 460
- Menü, 449
- Neu, 451
- Neu laden, 452
- öffnen, 452
- Ordner hinzufügen, 467
- schließen, 452
- Speichern, 452
- URL zu Projekt hinzufügen, 466
- zuletzt verwendete, 477

Projekte,

- Batchverarbeitung mit Projekten, 371
- benennen, 367
- Eigenschaften, 367
- erstellen und bearbeiten, 367
- globale Ressourcen hinzufügen, 466
- Pfad installierter Dateien, 28
- speichern, 367
- Übersicht, 366, 449
- verwenden, 371
- Vorteile, 371

Projekte in XMLSpy,

- erstellen, 93
- Vorteile, 93

Projektfenster, 13, 31

- Anzeige ein/aus, 683

Projekt-Fenster, 682, 683**Projektverwaltung in XMLSpy, 93****Prozessor-Informationen, 1129****PUBLIC,**

- Identifizier - Katalog, 506

PVCS Version Manager, 376

Q

QName Serialisierung,

- Rückgabe durch XPath 2.0 -Funktionen, 1139

R

Rechtschreibprüfung, 635

- benutzerdefiniertes Wörterbuch, 635

Rechtliches, 1188**Rechtschreiboptionen, 639****Referenz, 411****Referenz für Programmierer, 692****Regionen in SQL-Scripts, 587****Register, 667****Registrierung,**

- Plug-In, 736

Registrierung der Altova-Software, 687**Registry,**

- Einstellungen, 667

Regular Expression,

- in Suchstring, 440

Regular Expressions, 445**Remove,**

- tracepoint - XSLT debugger, 550

Ressourcen,

- Statusaktualisierungen im Hintergrund beschleunigen, 376

RichEdit, 561**RichEdit 3.0, 674****Rückgängig, 432****Rückwärtskompatibilität,**

- des XSLT 2.0-Prozessors, 1132

S

save, 906**schema, 808**

- DB auf Basis von Schema erzeugen, 619
- Design-Ansicht, 629
- Dokumentation, 62
- Dokumentation Schriftart, 673
- einer DB-XML zuweisen, 598
- Einstellungen, 667
- Einstellungen für Oracle, 533
- in UML konvertieren, 517
- siehe auch XML-Schema, 32, 510
- Verwaltung und Zuweisung in IBM DB2-Datenbanken, 595

Schema Design,

Schema Design,

Menü, 521

Schema Design-Ansicht,

Diagramm anzeigen, 532

Globale Elemente anzeigen, 531

Zoom-Funktion, 531

Schema editing,

annotations, 121

comments, 121

content model, 121

processing instructions, 121

Schema Schriftarten, 673**Schema View, 116****SchemaAgent,**

arbeiten mit, 204, 205, 209, 210

erweiterte Validierung, 537

Schemas anzeigen, 536

Schemas von XMLSpy aus öffnen, 209

Verbindung trennen, 536

Verbindung zu Server herstellen von XMLSpy, 535

SchemaAgent in XMLSpy, 201**SchemaAgent Server,**

Verbindung herstellen, 202

Schema-Ansicht,

Eingabehilfen, 134

konfigurieren, 42

vorwärts und rückwärts navigieren, 149

Schema-Ansicht, suchen in,

siehe In Schemas suchen, 212

Schemabearbeitung,

Annotationen, 117

Kommentare, 117

Processing Instructions, 117

SchemaDocumentationDlg,

AllDetails, 945

Application, 945

IncludeAll, 947

IncludeAttributeGroups, 947

IncludeComplexTypes, 947

IncludeGlobalElements, 948

IncludeGroups, 948

IncludeIndex, 949

IncludeLocalElements, 949

IncludeRedefines, 949

IncludeSimpleTypes, 950

OptionsDialogAction, 951

OutputFile, 951

OutputFileDialogAction, 951

OutputFormat, 952

Parent, 952

ShowAnnotations, 952

ShowAttributes, 952

ShowChildren, 953

ShowConstraints, 954

ShowDiagram, 953

ShowEnumerations, 953

ShowNamespace, 954

ShowPatterns, 954

ShowProgressBar, 955

ShowProperties, 955

ShowResult, 955

ShowSingleFacets, 955

ShowSourceCode, 956

ShowType, 956

ShowUsedBy, 956

Schemafähigkeit,

des XPath 2.0- und XQuery-Prozessors, 1138

Schemamodule, 193, 537, 538**Schemas,**

für IBM DB2 verwalten, 595

im Speicher, 520

Schemas, suchen in,

siehe In Schemas suchen, 212

Schema-Übersicht, 32, 117**Schema-Validierung eines XML-Dokuments,**

für XQuery, 1135

Schließen, 422

Projekt, 452

Schriftart, 1185

Schema, 673

Schema Dokumentation, 673

Schriftarten in der Textansicht, 98**Schriftgröße,**

in Browser-Ansicht, 633

Schriftsystem, 1185**Schriftsysteme, 1186****Scripts in XSLT/XQuery,**

siehe Erweiterungsfunktionen, 1143

Sequence-Kompositor,

verwenden, 36

SharePoint® Server, 471**SimpleTypes,**

in Schema-Definitionen, 48

Skript-Editor,

Beschreibung der Benutzeroberfläche, 698

Hauptfenster, 698

Skripting,

- Makros anzeigen, 644
- Zu Skripting-Umgebung wechseln, 643

Skripting-Projekt,

- Anmerkungen zur Programmierung, 723
- Application Event Handler, 714
- Event Handler, 702
- Formulare, 702, 708
- Globale Deklarationen, 702, 706
- Makros, 702
- Makros in, 717
- Schritte bei der Erstellung, 704
- und Events, 714

Skripting-Projekte,

- für XMLSpy, 697
- für XMLSpy Projekt, 697

Skripting-Projektstrukturfenster, 698**Skripting-Sprache, 704****Skripting-Umgebung, 694**

- Übersicht über die Verwendung, 696

Skripts, 679**Skriptsprache, 679****Smart Restrictions, 143****Software-Produktlizenz, 1192****Source control, 680****Speicher, 376**

- Speichern von Schemainformationen, 520

Speichern,

- Projekt, 452

Splashscreen, 671**SPP Datei-Speicherpfade, 449****Sprache,**

- Skripting-Sprache - wechseln, 704

SPS,

- neuer XML-Datei zuweisen, 412

SPS-Datei,

- zu XML-Datei zuweisen, 556

SPS-Tabellen,

- dynamischen Tabellen bearbeiten, 152

SPS-Tabellen in der Authentic-Ansicht,

- Verwendung, 293

SpyProject,

- CloseProject, 957
- ProjectFile, 958
- RootItems, 958
- SaveProject, 958
- SaveProjectAs, 958

SpyProjectItem,

- ChildItems, 959
- FileExtensions, 959
- ItemType, 959
- Name, 960
- Open, 960
- ParentItem, 960
- Path, 960
- ValidateWith, 960
- XMLForXSLTransformation, 960
- XSLForXMLTransformation, 961
- XSLTransformationFileExtension, 961
- XSLTransformationFolder, 961

SpyProjectItems,

- AddFile, 961
- AddFolder, 962
- AddURL, 962
- Count, 962
- Item, 963
- RemoveItem, 963

SQL Editor,

- Abfrage erstellen, 591
- Beschreibung, 587
- im Fenster Datenbankabfrage, 587

SQL Server,

- XML-Schemas verwalten, 600

Standard,

- Encoding, 677
- Menü, 660

Standardansicht,

- Einstellung im Hauptfenster, 668

Standardmäßiger Editor, 668**Star Team, 376****Start,**

- XSLT Debugger, 549

Startgruppe,

- hinzufügen (Kontextmenü), 664

Statische (SPS) Tabellen in der Authentic-Ansicht,

- Verwendung, 293

Status,

- Hintergrundaktualisierungen, 390

Statusleiste, 24**Stil, 673, 674****Stoppen,**

- XSLT Debugger, 549

Strukturierter Text, 670**Studio,**

- XMLSPY Integration in Visual Studio .NET, 395

Stylesheet PI, 547, 548

StyleVision, 518

zur Bearbeitung eines StyleVision Power Stylesheet, 556

StyleVision Power Stylesheet,

in %STYLEVISION%> bearbeiten, 556

zu XML-Datei zuweisen, 556

StyleVision Power Stylesheet zu XML-Datei zuweisen, 556**Suchen,**

Hilfe, 686

Text in Dokument, 440

und Text in Dokument ersetzen, 444

Suchen in Schemas,

siehe In Schemas suchen, 212

Support Center, 688**Support-Optionen, 29****Sybase,**

Konvertierung von Datentypen in XML-Schema

Generierung von DB, 1171

Symbol,

Hinzufügen zu Symbolleiste/Menü, 653

Symbole,

große anzeigen, 664

Symbolleiste, 24

Befehle anpassen, 653

Ein/Aus, 654

große Symbole anzeigen, 664

Makro hinzufügen, 662

Neue erstellen, 654

Symbolleiste & Menübefehle zurücksetzen, 654

Syntaxfärbung, 668, 671

für XQuery, 239

Syntaxüberprüfung von JSON-Dokumenten, 324

T

Tabelle,

automatisch erzeugen, 668

Befehl in Grid-Ansicht, 501

Farben, 675

Tabelle Ansicht, 670**Tabellen,**

dynamische (SPS)-Tabellen (SPS) bearbeiten, 152

in der Authentic-Ansicht, 279

in der Grid-Ansicht, 109

Tabellen in der Authentic-Ansicht,

Arbeiten mit (statischen und dynamischen) SPS Tabellen,

293

Arbeiten mit XML-Tabellen, 294

Symbole zum Bearbeiten von XML-Tabellen, 298

Verwendung, 293

Tabellenansicht,

in Grid-Ansicht wechseln, 501

Spalten sortieren, 503

Zeile anhängen, 502

Zeile einfügen, 502

Table View,

how to use, 82

Tabulatorgröße,

einstellen, 631

und Pretty-Print, 631

Tastaturzuordnungen, 686**Tatstaturkürzel, 655****Technische Informationen, 1181****Technischer Kundendienst, 688****Templates-Fenster,**

in XSLT/XQuery Debugger, 261

terminate, 811**Text,**

als Child in Grid-Ansicht hinzufügen, 494

in der Authentic-Ansicht bearbeiten, 288

in der Authentic-Ansicht formatieren, 288

in Dokument suchen, 440

in Grid-Ansicht anhängen, 487

in Grid-Ansicht einfügen, 480

Konvertieren in (Grid-Ansicht), 500

Pretty-Print, 439

Schriftart, 674

suchen und ersetzen, 444

Textansicht, 97

Bearbeiten in, 71

Eingabehilfen, 105

Einrückung, 98, 100

Formatieren von Text, 98

in der Textansicht, 98

intelligente Bearbeitungsfunktionen, 102

Klappleiste, 100

Lesezeichen, 100

Schriftarteigenschaften, 98

spezielle Bearbeitungsfunktionen für XML-Dokumente, 170

und Kommentare in XML-Dokumenten, 170

und leere Zeilen in XML-Dokumenten, 170

Zeilennummerierung, 100

Zeilenumbruch, 98

Text-Ansicht, 628

Text-Ansicht, 628

- Auto-Kompletierung Ein/Aus, 670
- Schema Schriftarten, 673

Textansicht-Events, 714**Textdatei,**

- als XML importieren, 606
- Export von XML-Daten, 621

TextImportExportSettings,

- DestinationFolder, 964
- EnclosingCharacter, 964
- Encoding, 964
- EncodingByteOrder, 964
- FieldDelimiter, 964
- FileExtension, 965
- HeaderRow, 965
- ImportFile, 965

Themen,

- Ansicht im Inhaltsverzeichnis, 685

Tiefer stufen,

- Befehl, 504

Tooltip,

- anzeigen, 664
- Kürzel anzeigen, 664

Tracepoints,

- in XSLT/XQuery Debugger verwenden, 266

Transformation,

- siehe XSLT-Transformation, 542

Tutorial,

- Beispieldateien, 30
- Ziele, 30

Tutorials,

- Pfad installierter Dateien, 28

type,

- Erweiterung im XML-Dokument, 69

UCS-4, 1185**UML,**

- Schemas konvertieren, 517

Unicode,

- Konsortium, 1185
- Standard, 1185
- Unterstützung, 1185

Unicode-Zeichen, 1185**Universalschriftart, 1185****Unnamed Element Relationships,**

- MS SQL Server Schema-Einstellungen, 535

Unselected, 630**Urheberrecht, 688****URL, 814, 906, 914, 915**

- per E-Mail senden, 427

UTF-16, 677**UTF-8, 1185**

V

Validator,

- in Altova-Produkten, 1183

Validieren, 506

- XML-Dokument, 76

Validieren von XML-Dokumenten, 168**Validierung, 26**

- DTD zu XML-Dokument zuweisen, 510
- von verknüpften Schemas mit Hilfe von SchemaAgent, 210
- XML-Schema zu XML-Dokument zuweisen, 511

Validierungsmeldungen, 16**Variablenfenster,**

- in XSLT/XQuery Debugger, 259

Verbergen, 682, 683**Verbindung mit SchemaAgent Server trennen, 536****Verbindung zu einer Datenbank herstellen, 335, 564****Verbindung zu SchemaAgent Server herstellen, 202****Verbindung zu SchemaAgent Server von XMLSpy, 535****Verbindung zur Datenquelle, 342, 573****Verbindung zur Datenquelle herstellen, 336, 338, 566, 569****Verbindungsassistent, 336, 566****Vergleiche,**

- von Dateien, 374
- von Verzeichnissen, 375

Vergleichen,

- Diffdog konfigurieren, 390

U

Überlappend,

- Fenster, 682

Übernehmen, 667**Überprüfen,**

- Rechtschreibung, 635

Übersicht, 31**Überspringen,**

- XSLT Debugger, 550

UCS-2, 677, 1185

Vergrößern/Verkleinern,

in Schema Design-Ansicht, 531

Versionskontrolle, 376

aktivieren /deaktivieren, 454

aktuellste Version holen, 455

auschecken, 457

Auschecken rückgängig, 459

ausgliedern, 461

Diffdog Vergleichseditor, 390

Eigenschaften, 465

Einchecken, 458

Hinzufügen in Versionskontrolle, 460

Installationsverfahren, 383

Projekt öffnen, 453

Status aktualisieren, 465

Unterschiede anzeigen, 463

unterstützte Anbieter, 452

Verlauf anzeigen, 462

Versionskontrollsystem wechseln, 465

Versionskontrollmanager, 465**Versionsnummer, 688****Vertrauenswürdige Pfade für Authentic Skripts, 563****Vertrieb,**

von Altova Produken, 1191

von Altova Software-Produkten, 1188

von Altova-Software-Produkten, 1189

Verzeichnisse,

zwei vergleichen, 646

Verzeichnisse vergleichen, 646**Verzeichnisvergleiche, 375****View,**

Collapse, 630

Visual Studio .Net,

bekannte Probleme mit XMLSpy, 401

und XMLSPY, 395

und XMLSPY Unterschiede, 398

und XSLT Debugger, 400

Volltextsuche, 686**Von rechts nach links, 1186****Vorlagen,**

Ordner, 30

von XML-Dokumenten in Authentic-Ansicht, 554

Vorlagendateien,

für neue Dokumente, 412

Vorwärts,

in der Schema-Ansicht, 149

VS .NET,

und XMLSpy Integrationspaket, 396

W

Web Server, 688**Wechseln,**

Zu Skript-Umgebung, 643

Whitespace im XML-Dokument,

Behandlung durch den Altova XSLT 2.0-Prozessor, 1132

Whitespace Nodes im XML-Dokument,

und Behandlung durch den XSLT 1.0-Prozessor, 1130

Whitespace-Behandlung,

und XPath 2.0-Funktionen, 1138

Whitespace-Markierungen, 631**Wiedergabe, 1186****Wiederholende Elemente, 670****Windows,**

Unterstützung für Altova-Produkte, 1182

Windows 2000, 1185**Windows NT, 1185****Wohlgeformtheit von XML-Dokumenten, 168****Wohlgeformtheitsprüfung, 505**

bei XML-Dokument, 76

Wohlgeformtheitsprüfung von JSON-Dokumenten, 324**Word 2007, 327****Word-Dokument,**

als XML importieren, 614

Wörterbuch,

aktuelles ändern, 635

benutzerdefiniertes Wörterbuch hinzufügen, 635

Rechtschreibprüfung, 635

X

XInclude, 436, 481, 488, 495

appending in Grid View, 488

in der Grid-Ansicht als Child anhängen, 495

in die Grid-Ansicht einfügen, 481

in Grid-Ansicht einfügen, 436

in Textansicht einfügen, 436

in XML-Dokument einfügen, 170

XML, 1185

Dokument, 67

Menü, 478

Neues Dokument, 67

XML, 1185

- Oasis-Katalog, 506
- Rechtschreibprüfung, 635
- Symbole, 67

XML Diff,

- Dateien vergleichen, 649
- Dateivergleich, 644
- Verzeichnisse vergleichen, 646

XML Import,

- Basierend auf Schema, 618

XML Parser,

- Info, 1183

XML Schema,

- generieren anhand einer DTD (Enterprise und Professional Edition), 190
- MS SQL Server Schema-Einstellungen, 535

xml:base, 147

- and XInclude, 436, 481, 488, 495

xml:id, 147**xml:lang, 147****xml:space, 147****XML; Kopieren als strukturierten Text, 433****XMLData,**

- AppendChild, 975
- EraseAllChildren, 976
- EraseCurrentChild, 977
- GetChild, 978
- GetChildKind, 978
- GetCurrentChild, 979
- GetFirstChild, 979
- GetNextChild, 980
- HasChildren, 981
- HasChildrenKind, 981
- InsertChild, 982
- IsSameNode, 983
- Kind, 983
- MayHaveChildren, 983
- Name, 984
- Parent, 984
- TextValue, 984

XML-Datei,

- von DTD oder XML-Schema erzeugen, 518

XML-Datei Global Resources, 352**XML-Daten,**

- in Datenbank exportieren, 624
- in Textdatei exportieren, 621

XML-Daten in Datenbank exportieren, 624**XML-Daten in Textdateien exportieren, 621****XML-Datenbank,**

- neue Datenzeile in die Authentic-Ansicht laden, 557

XML-DB,

- neue XML-Datenzeile laden, 301

XML-Dokument,

- Bearbeiten in der Authentic-Ansicht, 175
- generieren anhand einer DTD, 190
- generieren anhand von XML-Schema (Enterprise und Professional Edition), 192
- in der Authentic-Ansicht öffnen, 275
- in Textansicht bearbeiten, 71
- Oracle XML DB durchsuchen, 604
- to XSLT Stylesheet zuweisen, 548

XML-Dokumente, 165

- automatische Validierung, 166
- Automatisieren von XQuery-Ausführungen, 179
- Automatisieren von XSLT-Transformationen, 179
- Bearbeitungsfunktionen in der Textansicht, 170
- Dateipfade einfügen, 170
- in der Grid-Ansicht bearbeiten (Enterprise und Professional Edition), 172
- Kodierung, 187
- öffnen, 166
- Schemas generieren, 187
- Schemas zuweisen (einschl. DTDs), 168
- speichern, 166
- Standardansichten, 166
- Suchen und Ersetzen in XML-Dokumenten, 187
- Text importieren und exportieren, 187
- transformieren mit XSLT, 179
- und Kommentieren in der Textansicht, 170
- und leere Zeilen in der Textansicht, 170
- und XPath-Ausdruck eines Node, 170
- und XQuery, 179
- validieren, 168
- Wohlgeformtheit prüfen, 168
- XInclude einfügen, 170
- XPath-Ausdrücke auswerten, 187

XML-Konformität, 668**XML-Prolog,**

- als Child in Grid-Ansicht hinzufügen, 494
- in Grid-Ansicht anhängen, 488
- in Grid-Ansicht einfügen, 481
- Konvertieren in (Grid-Ansicht), 500

XML-Schema,

- Ansicht konfigurieren, 42
- Content Model-Ansicht konfigurieren, 528
- Content Model-Diagramm, 523

XML-Schema,

- Dokumentation generieren, 523
- einer DB-XML zuweisen, 598
- Einstellungen in der Schema Design-Ansicht, 521
- Erstellen eines einfachen Schemas, 32
- für IBM DB2 verwalten, 595
- Gehe von XML-Dokument zu Definition, 512
- Gehe von XML-Dokument zu Schema, 511
- in DTD konvertieren, 514
- Komponenten hinzufügen, 36
- Menübefehle zu, 510
- MS SQL Server Extensions, 533
- MS SQL Server Schema-Einstellungen, 534
- Namespace-Einstellungen in der Schema Design-Ansicht, 521
- Namespaces definieren, 35
- Navigation in der Design-Ansicht, 60
- neues erstellen, 32
- Oracle Extensions, 532
- Oracle Schema-Einstellungen, 533
- siehe auch Schema, 510
- Tutorial, 32
- Verwaltung und Zuweisung in IBM DB2-Datenbanken, 595
- von XML-Dokument generieren, 512
- während der Bearbeitung des XML-Dokuments ändern, 86
- XML-Datei erzeugen, 518
- zu XML-Dokument zuweisen, 511

XML-Schema-Datentypen,

- bei der Generierung von XML-Schema anhand einer ADO DB, 1170
- bei der Generierung von XML-Schema anhand einer MS Access DB, 1165
- bei der Generierung von XML-Schema anhand einer MS SQL Server DB, 1166
- bei der Generierung von XML-Schema anhand einer MySQL DB, 1167
- bei der Generierung von XML-Schema anhand einer ODBC DB, 1169
- bei der Generierung von XML-Schema anhand einer Oracle DB, 1168
- bei der Generierung von XML-Schema anhand einer Sybase DB, 1171
- Konvertierung in MS Access-Datentypen, 1173
- Konvertierung in MS SQL Server-Datentypen, 1175
- Konvertierung in MySQL-Datentypen, 1177
- Konvertierung in Oracle-Datentypen, 1179

XML-Schemas, 189

- andere Schemas inkludieren, 193
- in der Grid-Ansicht bearbeiten (Enterprise und Professional Edition), 192

- in der Schema-Ansicht bearbeiten (Enterprise und Professional Edition), 192
- in der Textansicht bearbeiten, 192
- in DTD konvertieren (Enterprise und Professional Edition), 192
- in Module aufteilen, 537
- inkludierte Schemas auf eine Ebene bringen (nivellieren), 538
- inkludierte Schemas nivellieren, 193
- plus DTDs, 168
- und globale Ressourcen, 168
- XML-Dokument generieren, 192

XML-Signatur, 558**XML-Signaturen, 311****XMLSpy, 411, 688**

- allgemeine Einführung, 10
- Features, 29
- Hilfe, 29
- Integration, 1079
- Plug-In Registrierung, 736
- und Eclipse, 402

XMLSpy API,

- aufrufen, 1107
- Dokumentation, 752

XMLSpy Befehlstabelle, 1094**XMLSpy Debugger-Perspektive, 410****XMLSpy Enterprise Edition,**

- Benutzerhandbuch, 3

XMLSpy Integration,

- Beispiel, 1080, 1081, 1082

XMLSpy Integrationspaket, 396, 403**XMLSpy Perspektive in Eclipse, 407****XMLSPY Plug-In, 735****XMLSpy Plugin für Eclipse,**

- installieren, 403

XMLSpy Plugin für VS .NET,

- installieren, 396

XMLSpyCommand,

- in XMLSpyControl, 1108

XMLSpyCommands,

- in XMLSpyControl, 1109

XMLSpyControl, 1110

- Beispiel für die Integration auf Applikationsebene, 1080
- Beispiel für Integration auf Applikationsebene, 1080, 1081, 1082
- Beispiele für die Integration auf Dokumentebene, 1084
- Dokumentation, 1079
- Integration auf Applikationsebene, 1080
- Integration auf Dokumentebene, 1083, 1084

XMLSpyControl, 1110

- Integration mittels C#, 1085
- Integration mittels HTML, 1089
- integration using C#, 1085
- object reference, 1108

XMLSpyControlDocument, 1116**XMLSpyControlPlaceholder, 1122****XMLSpyDocumentEditor,**

- MarkUpView, 1025

XMLSpyLib, 752

- Application, 797
- AuthenticDataTransfer, 817
- AuthenticRange, 821
- AuthenticSelection (obsolete), 1002
- AuthenticView, 849
- CodeGeneratorDlg, 868
- DatabaseConnection, 873
- Dialogs, 879
- DocEditEvent (obsolete), 986
- DocEditView (obsolete), 1005
- Document, 883
- Documents, 911
- DTDSchemaGeneratorDlg, 916
- ElementList, 919
- ElementListItem, 920
- ExportSettings, 921
- FileSelectionDlg, 924
- GenerateSampleXMLDlg, 936
- GridView, 940
- ProjectItem, 959
- SchemaDocumentationDlg, 944
- SpyProject, 957
- SpyProjectItems, 961
- TextImportExportSettings, 963
- XMLData, 974

XML-Tabellen in der Authentic-Ansicht,

- Editiersymbole, 298
- Verwendung, 294

XML-Text, 670

- kopieren, 432

XML-Text ein- und auskommentieren, 448**XML-Vorlagendatei,**

- in der Authentic-Ansicht, 275

XPath,

- auswerten, 17
- Generieren eines Node in einem XML-Dokument, 170

XPath (Fenster), 17**XPath 1.0,**

- in XPath Evaluator, 504

XPath 2.0,

- in XPath Evaluator, 504

XPath 2.0 Functions,

- implementation information, 1138

XPath 2.0-Funktionen,

- allgemeine Informationen, 1138
- spezifische Funktionen siehe unter fn:, 1138

XPath des ausgewählten Node im XML-Dokument,

- in die Zwischenablage kopieren, 435

XPath Evaluator,

- Verwendung, 504

XPath kopieren, 435**XPath zum ausgewählten Node, 151****XPath-Funktionen - Unterstützung,**

- für einzelne Funktionen siehe unter fn:, 1139

XPaths,

- für tracepoints definieren, 266

XPath-Watch-Fenster,

- in XSLT/XQuery Debugger, 259

XPointer,

- Generieren eines Node in einem XML-Dokument, 170

XPointer des ausgewählten Node im XML-Dokument,

- in die Zwischenablage kopieren, 435

XPointer kopieren, 435**XPointers, 436, 481, 488, 495****XQuery,**

- Ausführung, 242
- Datei öffnen, 238
- DB-Unterstützung, 244
- Dokumentvalidierung, 242
- Eingabehilfen, 239
- Erweiterungsfunktionen, 1143
- Funktionen für IBM DB2, 244
- in Textansicht bearbeiten, 237
- intelligente Editierfeatures, 240
- Prozessor-Informationen, 1129
- Syntaxfärbung, 239
- Variablen an XQuery-Dokument übergeben, 544
- zur Abfrage von XML-Datenbanken, 244

XQuery 1.0 Functions,

- implementation information, 1138

XQuery 1.0-Funktionen,

- allgemeine Informationen, 1138
- spezifische Funktionen siehe unter fn:, 1138

XQuery 1.0-Prozessor,

- Informationen, 1135

XQuery Debugger,

XQuery Debugger,

siehe XSLT/XQuery Debugger, 248

XQuery-Ausführung, 547**XQuery-Dateien,**

Dateierweiterungen in XMLSpy festlegen, 238

XQuery-Prozessor,

in Altova-Produkten, 1184

xs:QName,

siehe auch QName, 1139

xsi:type,

Verwendung, 69

XSL,

siehe XSLT, 548

XSL/Query Menü, 540**xsl:call-template, 231****XSL:FO,**

Transformation, 542

und XSLT-Transformationen, 228

xsl:param, 231**xsl:preserve-space, 1130****xsl:strip-space, 1130****xsl:with-param, 231****XSLT, 631**

Autokomplettierung in der Textansicht, 226

Dokumente, 226

Eingabehilfen, 226

Erweiterungsfunktionen, 1143

Funktionalität in XMLSpy, 226

in XMLSpy ändern, 91

Prozessor, 677

Prozessor-Informationen, 1129

Transformationen in XMLSpy, 228

und Batch-Transformationen, 228

Validierung, 226

XSLT 1.0-Prozessor,

Einschränkungen und implementierungsspezifisches Verhalten, 1130

XSLT 2.0 Stylesheet,

Namespace-Deklarationen, 1132

XSLT 2.0-Funktionen,

implementierungsspezifisches Verhalten, 1134

spezifische Funktionen siehe unter fn:, 1134

XSLT 2.0-Prozessor,

allgemeine Informationen, 1132

Informationen, 1132

XSLT Debugger,

aktuelle Execution Nodes einblenden, 550

aussteigen, 550

Breakpoint einfügen/löschen, 550

Breakpoints Dialogfeld, 551

Breakpoints Ein/Aus, 551

Debug-Fenster, 552

Debugger erneut starten, 549

Debugger starten, 549

Debugger stoppen, 549

Debuggersitzung beenden, 549

einsteigen, 549

Einstellungen, 552

enable/disable tracepoint, 551

in Visual Studio .NET, 400

insert/remove tracepoint, 550

überspringen, 550

XSLT Stylesheet,

öffnen, 548

zu XML-Dokument zuweisen, 547, 548

XSLT Stylesheet für FO,

zu XML-Dokument zuweisen, 548

XSLT/XQuery debuggen,

verwendete Dateien, 248

XSLT/XQuery Debugger,

Beschreibung der Benutzeroberfläche, 249

Beschreibung der Funktionsweise, 249

Einstellungen, 254

Fenster "Call Stack", 260

Fenster "Info", 261

Fenster "Meldungen", 260

Fenster "Templates", 261

Fenster "Variablen", 259

Fenster "XPath-Watch", 259

Fenster Ablaufverfolgung, 262

Funktionen und Verwendung, 248

Informationsfenster, 257

Kontextfenster, 258

Sitzung starten, 255

toolbar icons, 251

Tracepoints Verwendung, 266

Verwendung von Breakpoints, 263

XSLT-Dokumente,

mit XSL-Übersicht bearbeiten und verwalten, 230

und Info-Fenster, 234

XSLT-Dokumentstruktur, 21**XSLT-Funktionen,**

im Fenster XSL-Übersicht, 231

XSLT-Parameter,

über Schnittstelle an Stylesheet übergeben, 544

XSLT-Prozessoren,

XSLT-Prozessoren,

in Altova-Produkten, 1184

XSL-Transformation,

siehe XSLT, 89

XSLT-Transformation, 541

in FO, 542

in PDF, 542

in XMLSpy, 90

Tutorial, 89

XSLT-Datei zuweisen, 89

XSLT-Vorlagen,

im Fenster XSL-Übersicht, 231

XSL-Übersicht, 230**XSL-Übersicht (Fenster), 21, 231****ZIP-Dateien, 327****Zoomen in der Textansicht, 100****Zuletzt verwendete Dateien,**

Liste, 430

Zurück,

in der Schema-Ansicht, 149

Zurücksetzen,

Kürzel, 655

Menübefehle, 660

Symbolleiste & Menübefehle, 654

Zuweisen,

Kürzel zu Befehl, 655

Z

Zeichen,

Position, 631

Zeichen-Entities,

in der HTML-Ausgabe von XSLT-Transformationen, 1130

Zeichennormalisierung,

in XQuery-Dokument, 1135

Zeichensatz,

Kodierung, 677

Zeile,

anhängen (in Authentic-Ansicht), 562

einfügen (in Authentic-Ansicht), 562

Gehe zu, 631

kopieren (in Authentic-Ansicht), 562

löschen (in Authentic-Ansicht), 562

nach oben/nach unten verschieben, 562

Zeile in Grid-Ansicht,

anhängen, 502

einfügen, 502

Zeilenendezeichen, 631**Zeilenlänge,**

Zeilenumbruch in Textansicht, 630

Zeilenleiste, 631**Zeilennummerierung in der Textansicht, 100****Zeilenschaltung,**

siehe Eingabetaste, 313

Zeilenumbruch, 667

aktivieren/deaktivieren, 630

Ein/Aus, 630

Zeilenumbruch in der Textansicht, 98